Algoritmo Evolutivo para Seleção de Atributos

Larissa Viana da Silva

Universidade Federal de Ouro Preto, MG

Abstract: Evolutionary Algorithms (EA) are those whose principles are based on the behavior of nature, the most common type being the Genetic Algorithm (GA). GA's are widely applied to optimization problems, such as feature selection. In this context, the following work seeks to apply the GA in the selection of features for insertion in Decision Tree and Random Forest classification models.

Resumo: Algoritmos Evolutivos (AE) são aqueles cujos princípios se baseiam no comportamento da natureza, sendo seu tipo mais comum o Algoritmo Genético (AG). Os AG's são amplamente aplicados em problemas de otimização, como a seleção de atributos (Feature Selection). Neste contexto, o seguinte trabalho busca aplicar o AG na seleção de atributos para a inserção em modelos de classificação Arvore de Decisão (Decision Tree) e Floresta Aleatória (Random Forest).

Keywords: Feature Selection, Evolutionary Algorithm, Genetic Algorithm, Decision Tree, Random Forest.

Palavras-chaves: Seleção de Atributos, Algoritmo Evolutivo, Algoritmo Genético, Arvore de Decisão, Floresta Aleatória.

1 Introdução

A seleção de atributos (*feature selection*) é definido como o processo de busca por um subconjunto de atributos, dentre todos disponíveis, de acordo com determinado critério, que apresente desempenho preditivo equivalente ao conjunto completo (Abd-Alsabour, 2014). Ou seja, é analisar dentre todos os atributos à disposição, aqueles que podem ser removidos sem afetar o conceito original dos dados, criando-se um subconjunto com os atributos que possuem correlação. E isto é realizado removendo-se atributos redundantes ou irrelevantes, sem perder a ideia original dos dados

É tido como uma das técnicas de processamento de dados mais importantes, sendo comumente aplicado à algoritmos de classificação, em que a presença de atributos aleatórios ou ruidosos podem influenciar negativamente no seu processo de aprendizado. Além disso, os atributos redundantes ou não correlacionados, aumentam a complexidade do algoritmo de classificação sem conceder qualquer conhecimento útil (Rong et al., 2019).

A busca pelo subconjunto de atributos ideal é classificado como um problema NP-difícil. Diz-se que um problema pertence a classe NP-difícil se for pelo menos tão difícil como um NP-completo (um problema que pode ser resolvido por um algoritmo e a solução pode ser verificada em tempo polinomial), mas as soluções não podem ser necessariamente verificada dentro do tempo polinomial (Freitas, 2020).

Segundo Abd-Alsabour (2014), são 2^n estados possíveis no espaço de busca (onde n é o numero de atributos do conjunto de dados). E cada estado especifica um subconjunto de atributos factível. Portanto, o problema envolve vasculhar todo o espaço de busca até identificar o melhor.

Ainda de acordo com Abd-Alsabour (2014), para um valor de n grande, a analise de todos os estados se torna computacionalmente custoso. Entretanto, se for previamente conhecido quais d atributos, dentre o conjunto completo, devem ser escolhidos, então só será necessário vasculhar os subconjuntos de tamanho d.

Uma heurística possível para a seleção de atributos é o Algoritmo Evolutivo (AE). Esta abordagem se inspira na natureza através dos mecanismo de evolução natural, onde uma população inicial de indivíduos lutam pela sobrevivência e reprodução, de acordo com sua aptidão (Freitas, 2020). O tipo mais comum de AE, é o Algoritmo Genético (AG), sendo amplamente utilizado como método de otimização de funções (Abd-Alsabour, 2014).

O seguinte trabalho propõe a aplicação de um Algoritmo Genético na seleção de atributos de modelos regressivos de classificação. Os modelos utilizados foram Árvore de Decisão (*Decision Tree*) e Floresta Aleatória (*Random Forest*). É feita a analise comparativa dos resultados de acurácia obtida pelos modelos com o conjunto completo e com os resultados obtidos utilizando o AG.

2 Problema

"Um sistema de justiça criminal eficaz que garanta investigações rigorosas, processo oportuno e julgamento justo de criminosos suspeitos de homicídio é um pré-requisito para

defender o estado de direito, bem como para obter justiça para as vítimas de homicídio." (UNODC, 2013)

A citação acima pertence a edição de 2013 do livro "Estudo Global de Homicídios", e retrata a importância do processo de investigação e resolubilidade dos casos de homicídios para a manutenção do estado de direito e como garantidor de justiça. Entretanto, a nível global, a taxa de resolução de homicídios é ligeiramente superior a 60%, com taxa mais elevadas em países europeus e asiáticos, e menores nos continentes americanos (Sturup et al., 2015).

Um sistema capaz de estabelecer características acerca do criminoso, como a identificação do gênero, pode contribuir com as investigações e promover a elucidação dos casos. E ainda, saber quais atributos acerca do crime são mais relevantes pra essa determinação, tornam o modelo de identificação menos complexo e mais intuitivo.

Desta maneira, este trabalho propôs a utilização de modelos de regressão para a classificação do gênero do criminoso (homem, mulher ou não identificado). E a utilização de um algoritmo genético para otimização dos atributos.

O conjunto de dados utilizado é composto por casos de assassinatos do Relatório Suplementar de Homicídios do FBI (USA) de 1976 a 2014, com informações sobre mais de 22,000 homicídios. O dados incluem a idade, raça, sexo, etnia das vítimas e perpetradores, além da relação entre a vítima e o agressor e a arma utilizada. Os dados podem ser encontrados em: `www.kaggle.com/murderaccountability/homicide-reports`

3 Desenvolvimento

A seguir são descritos os passos da implementação dos modelos de regressão e do algoritmo genético:

3.1 Tratamento dos Dados

O conjunto de dados original possui 23 atributos, mas nem todos são relevantes para o problema. Assim, alguns atributos foram removidos, restando-se 17. Para visualizar o conjunto de dados completo acesse: `github.com/lrssv/ComputacaoEvolutiva/Data_treat`

Tabela 1. Amostra do conjunto de dados.

City	Year	Month	Incident	Crime Type	...
Anchorage	1980	January	1	Murder	
Anchorage	1980	March	1	Murder	...
Anchorage	1980	March	2	Murder	...
Anchorage	1980	April	1	Murder	...
...	...	...	...	...	...

Como pode ser observado na tabela 1, as informações disponibilizadas no conjunto de dados estão no formato categórico. No entanto, em sua forma original, os modelos não são capazes de reconhece-los. Assim, os dados passaram por uma transformação chamada *Target Encoder (TE)*. TE é o processo de substituição de um valor categórico pela média da variável de destino. Desta forma, o dados passam a ser representados no formato numérico, mantendo preservada a informação contida neles. O resultado da aplicação da TE pode ser observado na tabela 2

Tabela 2. Amostra do conjunto de dados com TE.

City	Year	Month	Incident	Crime Type	...
0.903846	1980	0.793266	1	0.772907	...
0.903846	1980	0.791463	1	0.772907	...
0.903846	1980	0.791463	2	0.772907	...
0.903846	1980	0.791873	1	0.772907	...
...	...	...	...	...	...

3.2 Modelos

Os modelos utilizados pertencem a biblioteca de algoritmos para aprendizado de máquina $Scikit-learn$. Foram implementados o modelos Classificador Arvore de Decisão (Decision Tree Classifier) e Classificador Floresta Aleatória (Random Forest Classifier), variando-se a profundidade máxima de cada modelo. A escolha dos parâmetros foi feita inicialmente de forma aleatória, com o intuito de verificar a eficácia do método de seleção de atributos proposto. Para as demais configurações dos modelos utilizou-se o valor padrão estabelecido pela biblioteca.

Os dados foram divididos em dois conjuntos: y os dados referente ao sexo do criminoso. E x os demais dados. Em seguida, os conjuntos foram separados em treino (80%) e teste (20%).

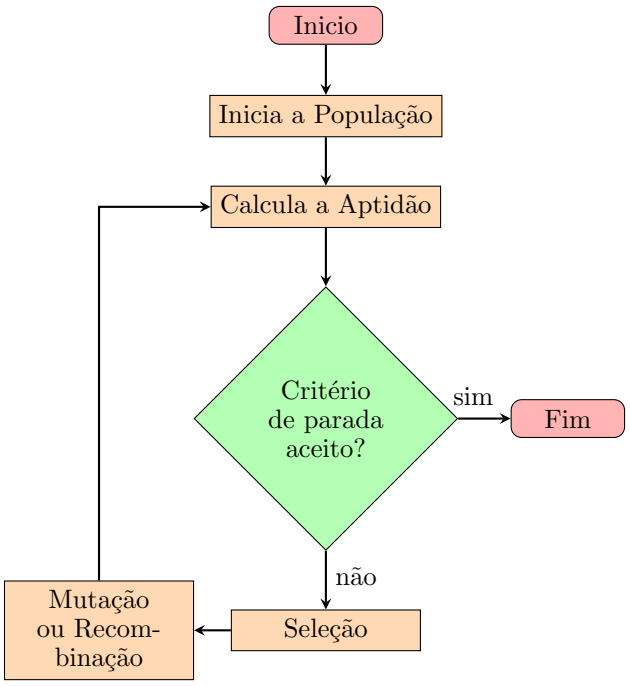
Os parâmetros estipulados e a acurácia de cada modelo podem ser vistos na tabela 3.

Tabela 3. Configuração e acurácia dos modelos.

Modelo	Profundidade Máxima	Acurácia
Arvore de Decisão 1	14	95,34%
Arvore de Decisão 2	10	95,42%
Arvore de Decisão 3	3	95,12%
Floresta Aleatória 1	14	95,52%
Floresta Aleatória 2	10	95,45%

3.3 Algoritmo Genético

O Algoritmo Genético em questão propõe:



Inicia-se com uma população de tamanho n, onde cada individuo da população representa uma solução do problema dado. Inicialmente os indivíduos são gerados aleatoriamente, e chamados de "Primeira Geração". Em seguida, é estipulada a aptidão de cada individuo, de acordo com critérios estabelecidos. Caso o critério de parada seja satisfeito, o algoritmo é finalizado. Caso contrário, os indivíduos passam por mutação ou recombinação (crossover), de acordo com a probabilidade de crossover e a intensidade de mutação, gerando novos indivíduos que são chamados de "filhos". Por fim, os melhores pais e filhos são selecionados para fazer parte da próxima geração, que irá constituir a nova população. Dessa forma, o processo se repete até que o critério de parada seja aceito.

3.4 Parte 1

A seguir são descritas as etapas de construção do algoritmo genético na primeira etapa desse estudo. Embora o problema seja multi-objetivo, ou seja, desejamos minimizar o erro e minimizar a quantidade de atributos presentes nos indivíduos, na primeira parte assumiu-se um único objetivo, estimando a função de fitness como sendo o somatório dos objetivos. Para população inicial foram determinados 100 indivíduos constituídos por um vetor de 18 posições. A seleção dos pais foi feita através de torneios, e a reprodução feita com probabilidades fixas. Por fim, foram selecionados 100 indivíduos sobreviventes para compor a próxima geração. O processo se repete 100 vezes, e ao final obtém-se a geração com os melhores indivíduos, e o melhor dentre todos.

3.4.1 População Inicial

Para a população inicial foram gerados 100 indivíduos. Cada individuo é constituído por um vetor de 18 posições, onde a primeira posição é um valor inteiro, variando de 0 a 4, representando o modelo. As demais posições indicam os atributos, que podem receber o valor $True$ ou $False$. Se a posição tem valor $True$, então o atributo correspondente está presente naquele individuo. Se $False$, o atributo não está presente. A representação dos indivíduos pode ser verificada na tabela 4

Tabela 4. Indivíduos da população

model	feature 1	feature 2	...	feature 16	feature 17
1	True	False	...	False	True
4	True	False	...	True	False
0	True	False	...	False	False
...	...	...	...	...	...
2	False	False	...	True	True
0	True	False	...	True	True

3.4.2 Fitness

A função de fitness é definida como sendo o somatório do erro médio quadrático (*Root Mean Square Error*), medida de erro comumente usada para aferir a qualidade do ajuste de um modelo. E o numero de atributos presentes no individuo. Portanto, possuímos dois objetivos: encontrar o melhor individuo que é simultaneamente capaz de minimizar o erro e a quantidade de atributos. Problemas dessa natureza pertencem a uma classe chamada "Multi-objetivo". Entretanto, conforme nessa primeira etapa do trabalho não trataremos desse ponto, utilizaremos apenas a minimização do somatório como objetivo.

Como os valores somados possuem escalas diferentes, foi estipulado um peso para equilibrar a soma. Dessa forma, multiplicou-se por 20 valor do erro. Então, a cada individuo da população foi atribuído um valor de fitness, que será fundamental na etapa de seleção.

3.4.3 Seleção

Na etapa de seleção utilizou-se o método de seleção por torneio, por ser um método simples e rápido de aplicar. O método de torneio foi implementado definindo-se uma quantidade n de torneios a serem executados, onde em cada etapa são selecionados dois pais de forma aleatória (podendo haver repetição), que são avaliados de acordo com o sua aptidão. O individuo vencedor é aquele que possui o menor valor de $fitness$.

3.4.4 Mutação ou Reprodução

Dentro da geração são selecionados pares de pais aleatoriamente e realizado a mutação ou reprodução de acordo com a probabilidade sorteada. A probabilidade de mutação foi definida como 0.1 e a de reprodução como 0.9. Desta forma, se a probabilidade sorteada for menor do que 0.1, é feita a mutação. Caso contrário, é feita a reprodução. Os processos ocorrem da seguinte maneira:

- Mutação: O filho gerado pela mutação é proveniente apenas do primeiro pai, e ocorre em duas etapas: no primeiro gene (o gene corresponde a uma posição do vetor, ou seja, um atributo), que é responsável por indicar o modelo. Esse gene recebe um novo valor aleatório entre 0 e 4. E ocorre também em outro gene que é escolhido aleatoriamente. Se esse gene possuir valor $True$, ele passa a ser $False$, e vice-versa.
- Recombinação (Crossover): A recombinação utiliza os dois pais, e gera dois filhos. Inicialmente é escolhido uma posição aleatória denominada "Ponto de Reprodução". Este ponto determina qual será o tamanho do vetor que sairá de cada pai e irá para os filhos. O primeiro filho recebe da primeira posição do vetor do pai, até o ponto de reprodução. E recebe do segundo pai o vetor que inicia na posição de reprodução até o final. E o mesmo ocorre com o segundo filho, mas na ordem inversa. A figura 1 demostra o processo de recombinação. Neste caso, o pronto de reprodução ocorre no 5º gene. Cada pai doa sua parte correspondente, e o processo finaliza gerando dois novos filhos.

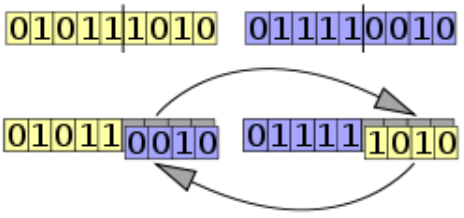


Figura 1. Aplicação da recombinação gerando novos filhos. Fonte: Wikipédia (2019)

3.4.5 Sobreviventes

Por fim, os filhos gerados têm sua aptidão avaliada através da função de *fitness*. Para a próxima geração, são selecionados 5 melhores pais e 95 melhores filhos, que irão recomeçar o ciclo até atingir o critério de parada.

3.4.6 Resultados

O critério de parada estabelecido foram 100 interações. Ao final do processo, obteve-se uma população com os 100 melhores indivíduos sobreviventes (figura 2). Além disso, foram selecionados o melhor individuo da população inicial, e o melhor de cada geração (figura 3).

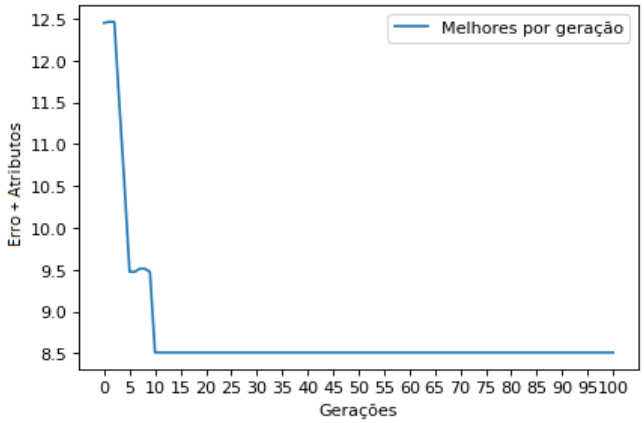


Figura 2. Gráfico dos melhores indivíduos de cada geração.

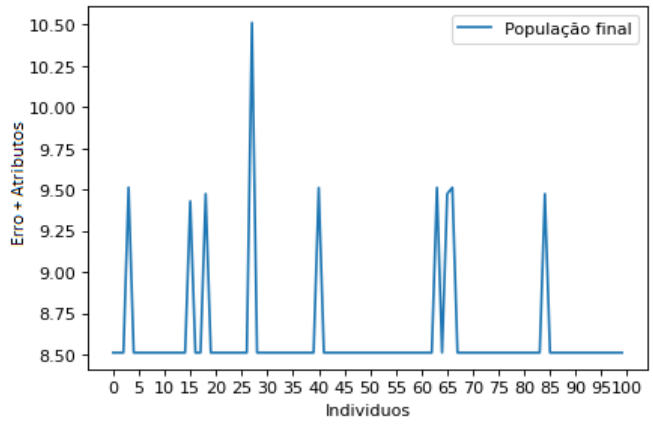


Figura 3. Gráfico dos indivíduos da população final.

Através do gráfico da figura 2 é possível observar que nas primeiras gerações o algoritmo convergiu para o melhor individuo. Entretanto, por volta da geração 10, a escolha do individuo se mantém constante até o final do processo. Na população final obteve-se uma geração de indivíduos com aptidões variadas, com a maioria no intervalo de 8,50 a 9,50.

O melhor indivíduo geral apresenta a seguinte configuração:

[0, False, False, False, False, False, False, False, False, False, True, False, False, False, False, False, False]
Fitness: 8.51
Acurácia: 86,17%

O modelo selecionado foi a Arvore de Decisão com profundidade máxima 14, e o individuo usou apenas o atributo na posição 10, que representa o sexo do criminoso. Através dessa única informação, o modelo foi capaz de classificar com 86,17% de acurácia. Embora este valor esteja abaixo dos valores obtidos inicialmente com o conjunto completo, o fato de ter usado só um atributo e ter obtido uma diferença de acurácia de 9,17%, é algo notável.

3.5 Parte II

Na segunda parte deste estudo incluiu-se ao algoritmo genético o "Controle de Parâmetros", onde cada indivíduo passou a carregar além dos atributos e do modelo, uma intensidade de mutação e uma probabilidade de recombinação. Além disso, o problema foi tratado como "Multi-Objetivo", implementando-se o NSGA-II *(Non-dominated Sorting Genetic Algorithm II)*. Nas sessões seguintes são descritas a teoria empregada e o passo-a-passo realizado na implementação do algoritmo genético, assim como os resultados obtidos.

3.5.1 Controle de Parâmetros

Os algoritmos genéticos carecem da determinação de uma série de parâmetros que exercem papel fundamental na sua performance, e que geralmente, são estipulados no inicio da execução e permanecem fixos durante todo o processo. No entanto, devido a natureza dinâmica e adaptativa destes algoritmos, determinar parâmetros fixos pode não ser boa alternativa, pois contrasta com esse espirito (Freitas, 2020). Através de diversas abordagens, verificou-se que a variação destes parâmetros se mostrava eficaz aos algoritmos genéticos. Assim, o presente estudo buscou através da variação dos parâmetros de intensidade de mutação e probabilidade de recombinação, otimizar a performance do algoritmo idealizado na parte I (ver seção 3.4).

Ao criar a população inicial, mantendo-se o tamanho de 100 indivíduos, além dos 17 atributos e o modelo presentes, foi acrescentado a cada indivíduo um vetor contendo a intensidade de mutação e a probabilidade de recombinação. Os valores inicias foram determinados aleatoriamente, no intervalo de 0 a 0.5 para a intensidade de mutação, e de 0.5 a 1 para a probabilidade de recombinação. Assim, os indivíduos da população passaram a ter o formato conforme demonstrado na tabela 5:

Tabela 5. Novo formato dos indivíduos

model	parameters	...	feature 16	feature 17
0	[0.250438, 0.718520]	...	True	False
3	[0.176497, 0.945894]	...	False	False
3	[0.354603, 0.943208]	...	False	True
...	...	...	...	...
2	[0.287538, 0.571001]	...	True	True
4	[0.145632, 0.963299]	...	False	True

Outra alteração resultante do controle de parâmetros ocorre na reprodução, onde a probabilidade de recombinação para que esta aconteça é definida como a média da probabilidade dos pais. E caso haja a recombinação, os filhos recebem novos valores, de acordo com a equações 1 e 2:

$$child\_mutation\_strength = n \times ms1 + (1 - n) \times ms2 \quad (1)$$

$$child\_crossover\_probability = n \times cp1 + (1-n) \times cp2 \quad (2)$$

Em que $ms1$ é a intensidade de mutação do pai 1, $ms2$ é a intensidade de mutação do pai 2, $cp1$ é a probabilidade de recombinação do pai 1, $cp2$ é a probabilidade de recombinação do pai 2, e n é um valor aleatório uniforme entre 0 e 1.

De maneira semelhante ocorre na mutação, utilizando os valores de intensidade de mutação e probabilidade de recombinação apenas do pai 1, sendo determinado através das equações 3 e 4:

$$mutation\_strength = ms1 \times e^{tau*n} \quad (3)$$

$$crossover\_probability = cp1 \times e^{tau*n} \quad (4)$$

Em que tau recebeu o valor de 0.5 (o tau pode variar de 0 a 1, este valor foi escolhido ao acaso para posteriormente ser ajustado conforme o resultado do algoritmo). E n é o mesmo valor aleatório uniforme mencionado anteriormente.

Assim, a cada geração os indivíduos recebem novos valores para os parâmetros, que são continuamente reajustados até se completarem todas as iterações.

3.5.2 Problemas Multi-Objetivo

Conforme mencionado na sessão 3.4, o problema abordado neste trabalho pertence a classe de problemas "Multi-objetivo (MOPs)", em que a qualidade de uma solução é definida pelo seu desempenho em relação a vários objetivos (Freitas, 2020). Portanto, possuímos dois objetivo: encontrar o melhor individuo que é simultaneamente capaz de minimizar o erro e a quantidade de atributos.

Para a construção do algoritmo multi-objetivo foram utilizados os conceitos de Dominância e Ótimo de Pareto, que são descritos a seguir:

3.5.2.1 Dominância

No problema abordado desejamos minimizar os objetivos. Neste caso, dado duas soluções em um espaço de busca, uma solução domina outra, se esta possuir uma pontuação pelo menos tão baixa para todos os objetivos, e estritamente baixa para pelo menos um. Em objetivos conflitantes, não há uma única solução que domine todas as outras. E uma solução é dita "não-dominada"se esta não for dominada por nenhuma outra.

3.5.2.2 Ótimo de Pareto

Em problemas multi-objetivo tem-se ao invés de uma única solução ótima, um conjunto de soluções. Estas soluções são ótimas por não existirem outras no espaço de busca que sejam melhores do que elas, quando todos os objetivos são simultaneamente considerados. Assim, o conjunto de soluções ótimas (aquelas que não são dominadas por nenhuma outra solução viável) é chamado de Ótimo de Pareto (de Castro, 2001). Para determinar o Ótimo de Pareto, são estimadas as Frentes de Pareto, em que cada frente recebe um posto (no inglês *rank*) que indica sua otimalidade (Vargas, 2018). Esse valor pode ser associado ao fitness correspondente a cada individuo da população. Quanto menor o posto (ou fitness), mais próximas as soluções daquela frente estão do valor ótimo. Na figura 4 é exemplificado a frente de Pareto, constituída pelo conjunto de soluções ótimas que passam pela linha em vermelho. O ponto C não está presente na frente porque é dominado tanto pelo ponto A quanto pelo ponto B. Os pontos A e B não são estritamente dominados por nenhum outro e, portanto, ficam na fronteira (Wikipédia, 2020).

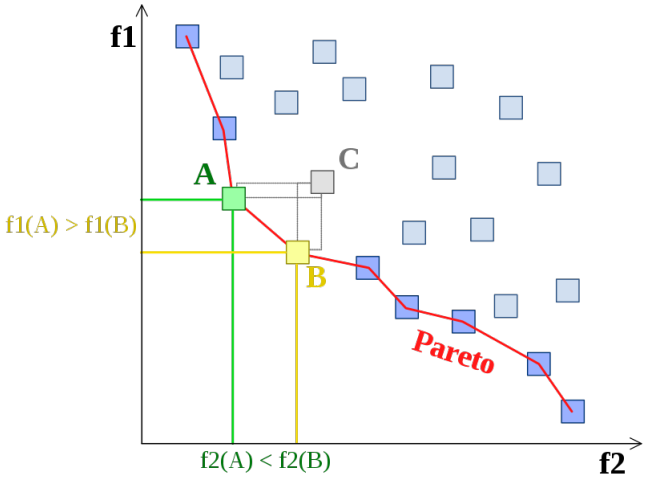


Figura 4. Exemplo de Frente de Pareto (em vermelho)
Fonte: Wikipédia (2020)

3.5.3 NSGA-II

Segundo Vargas (2018), dentre todos os algoritmos evolucionistas presentes na literatura, o NSGA-II é o mais popular entre eles. Este algoritmo consiste na seleção de uma nova população através de dois operadores principais: a Ordenação de Pareto (no inglês *Non-dominated Sorting*) e a *crowding distance (CD)*, que é um mecanismo de promoção da diversidade da população, livre de parâmetros. O principio de operação do NSGA-II consiste na seleção da próxima geração através da união da população corrente com os filhos gerados, que são classificados de acordo com a Ordenação de Pareto. Os indivíduos pertencentes as frentes com os postos mais baixos terão preferencia de entrada na próxima geração da população. Caso seja necessário selecionar apenas alguns indivíduos da frente, é utilizado como critério de desempate aquele que possuir maior valor de CD (Vargas, 2018).

Na figura 5 é representado o procedimento de funcionamento do NSGA-II, onde R_t corresponde a geração formada pela união da geração corrente e os filhos gerados, indicados por P e Q. Assim, R_t é dividido em frentes, representados por F, de acordo com a qualidade de suas soluções. As frentes com menor posto são selecionadas primeiro, até completar o tamanho definido para a próxima geração. Caso uma frente não seja inteiramente selecionada, é utilizado o CD para determinar quais indivíduos que serão escolhidos. Os demais indivíduos são rejeitados, e a nova população P é definida.

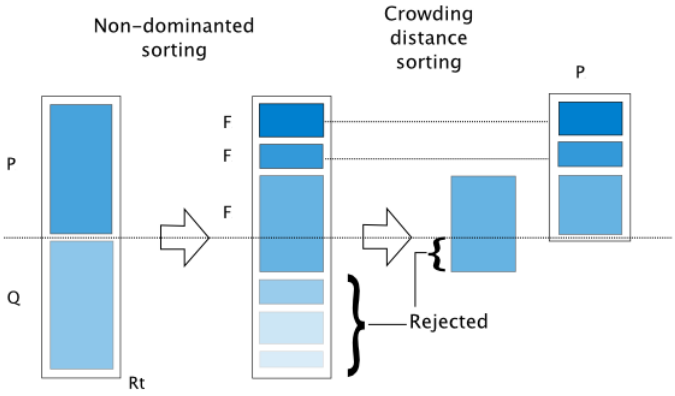


Figura 5. Ilustração do procedimento do NSGA-II
Fonte: pymoo (2020)

3.5.4 Resultados

Foram feitos dois experimentos: um com 50 iterações e outro com 100, ambos com uma população inicial de 100 indivíduos. O intuito da estratégia adotada é verificar se é possível diminuir a complexidade computacional – pois rodar um modelo para todo indivíduo presente na população a cada geração promove um custo muito elevado – utilizando-se menos iterações, e obter uma acurácia de mesmo valor, ou superior a obtida com 100 iterações.

Na população inicial foram avaliados os indivíduos da primeira frente de Pareto, que são descritos abaixo:

Indivíduo 1: [4, [0.1027021502858, 0.532205411662], False, False, False, True, False, False, True, False, False, False, True, False, False, False, True, False, True]
Objetivos: [0.376744466879, 5]
Acurácia: 86,21%

Indivíduo 2: [0, [0.395197126707, 0.613504131886], False, True, True, False, False, True, False, False, False, False, True, False, False, True, False, True, True]
Objetivos: [0.3716378175581, 7]
Acurácia: 86,20%

Indivíduo 3: [0, [0.00549329202211, 0.540649943473], True, False, True, False, False, False, False, False, False, True, True, False, False, False, True, True, False]
Objetivos: [0.372658441167, 6]
Acurácia: 86,21%

Indivíduo 4: [0, [0.0838060215487, 0.974390866888], True, False, False, True, True, True, True, True, True, False, True, True, False, True, False, True, False]
Objetivos: [0.3698952358864, 11]
Acurácia: 86,21%

Indivíduo 5: [0, [0.1400633841930, 0.689186388844], False, True, True, True, True, False, True, False, False, True, True, True, False, True, True, True, False]
Objetivos: [0.3698211266179, 12]
Acurácia: 86,21%

Na figura 6 é exibido todos os indivíduos presentes na população inicial (em verde), e os indivíduos pertencentes à primeira frente de Pareto (em azul).

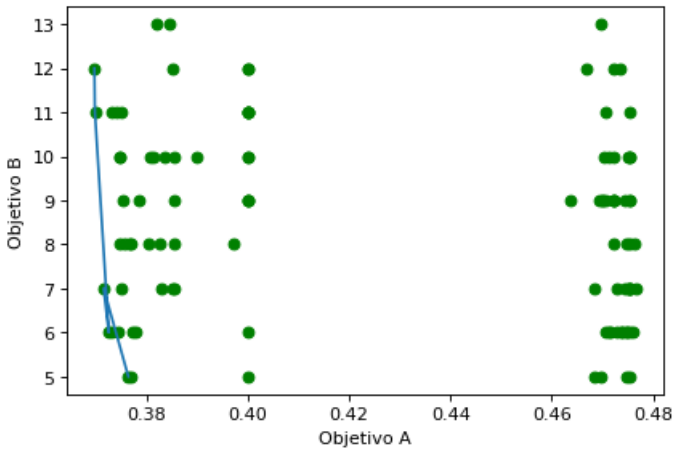


Figura 6. Primeira frente de Pareto da população inicial

Já na figura 7 são exibidas todas as frentes de Pareto da população inicial, representadas pelas retas em diferentes cores.

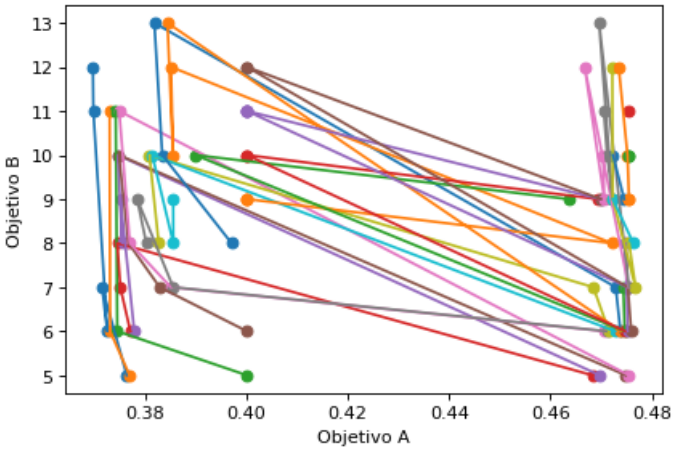


Figura 7. Frentes de Pareto da população inicial

Resultados com 50 iterações

Ao término das iterações, obteve-se a população final com os melhores indivíduos ao longo das gerações. Na figura 9 é representada a geração final, e o seu melhor indivíduo apresenta a seguinte configuração:

Melhor individuo: [0, [0.821808793081, 0.849431426604], False, False, False, False, False, True, False, False, False, False, True, False, False, False, False, True, False]
Objetivos: [0.3717115646275, 3]
Acurácia: 86,21 %

Resultados com 100 iterações

O mesmo procedimento realizado com 50 iterações, foi feito com as 100. Assim, ao término das 100 iterações, obteve-se a população final com os melhores indivíduos ao longo das gerações. Na figura 9 é representada a geração final, e o seu melhor indivíduo apresenta a seguinte configuração:

Melhor individuo: [0, [0.807330595654, 0.807350893106], False, False, False, True, False, False, False, False, False,

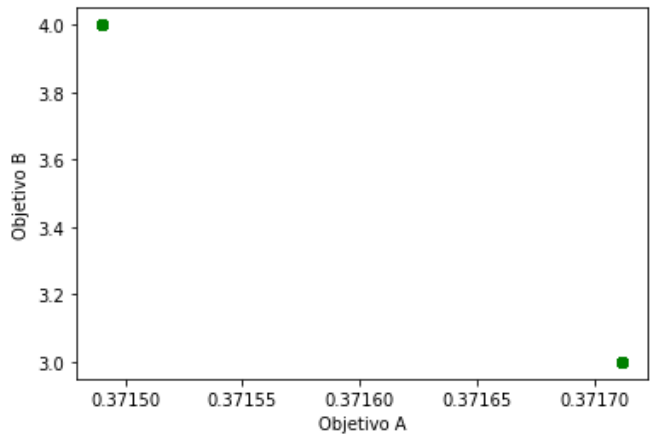


Figura 8. Indivíduos da população final com 50 iterações

False, True, False, False, False, False, False, False]
Objetivos: [0.3755785881919, 2]
Acurácia: 85,92 %

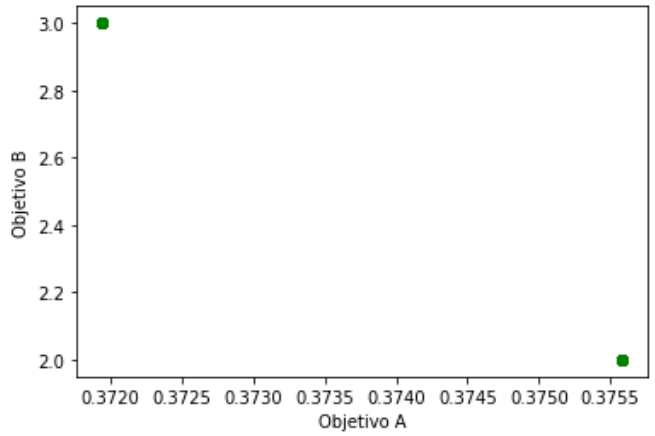


Figura 9. Indivíduos da população final com 100 iterações

4 Conclusão

Os algoritmos genéticos, tipo mais conhecido de algoritmo evolutivo, são amplamente utilizados como método de otimização de funções (Abd-Alsabour, 2014). Ainda que considerados simples, são empregados para propósitos de ensino, e para *benchmarking* de novos algoritmos, como também para a resolução de problemas relativamente simples em que a representação binária é adequada (Freitas, 2020).

Desta forma, o presente estudo buscou aplicar o algoritmo genético na otimização da seleção de atributos de modelos regressivos de classificação. Os modelos utilizados foram Árvore de Decisão (*Decision Tree*) e Floresta Aleatória (*Random Forest*).

O estudo se deu em duas etapas: na primeira assumiu-se um único objetivo, estimando a função de fitness como sendo o somatório dos objetivos (minimizar o erro e a quantidade de atributos). Foi realizado um único experimento, em que para a população inicial foram determinados 100 indivíduos constituídos por um vetor de 18 posições. A seleção dos pais foi feita através de torneios, e a reprodução feita com probabilidades fixas. Por fim, foram selecionados 100 indivíduos sobreviventes para compor a próxima geração. E o processo se repetiu por 100 iterações, até obter a geração final.

Na segunda etapa, incluiu-se ao algoritmo genético o "Controle de Parâmetros", onde cada indivíduo passou a carregar além dos atributos e do modelo, uma intensidade de mutação e uma probabilidade de recombinação. E o problema foi tratado como "Multi-Objetivo", implementando-se o NSGA-II. Assim, foram realizados dois experimentos, um com 50 iterações e outro com 100, ambos com uma população inicial de 100 indivíduos. O intuito da estratégia adotada foi verificar se era possível diminuir a complexidade computacional utilizando-se menos iterações, e obter uma acurácia de mesmo valor, ou superior a obtida com 100 iterações.

Na primeira etapa o melhor indivíduo geral apresentou a seguinte configuração:

[0, False, False, False, False, False, False, False, False, False, True, False, False, False, False, False, False]
Fitness: 8.51
Acurácia: 86,17%

Enquanto na segunda etapa, o melhor individuo apresentou a seguinte configuração com 50 e com 100 iterações, respectivamente:

[0, [0.821808793081, 0.849431426604], False, False, False, False, False, True, False, False, False, False, True, False, False, False, False, True, False]
Objetivos: [0.3717115646275, 3]
Acurácia: 86,21 %

[0, [0.807330595654, 0.807350893106], False, False, False, True, False, False, False, False, False, False, True, False, False, False, False, False, False]
Objetivos: [0.3755785881919, 2]
Acurácia: 85,92 %

Portanto, pode-se concluir que a inclusão do Controle de Parâmetros e do NSGA-II atribuiu ao algoritmo maior acurácia em ambos experimentos realizados. O melhor resultado foi obtido utilizando-se 50 iterações, o que além de atribuir maior acurácia, ainda promoveu menor complexidade computacional ao algoritmo genético proposto.

Referências

Abd-Alsabour, N. (2014). A review on evolutionary feature selection. In *2014 European Modelling Symposium*, 20–26. IEEE.

de Castro, R.E. (2001). *Otimização de estruturas com multi-objetivos via algoritmos genéticos*. Ph.D. thesis, Tese de Doutorado, Universidade Federal do Rio de Janeiro, Rio de Janeiro ....

Freitas, A.d. (2020). Introdução a resolução de problemas. `http://alandefreitas.com/cursos/ce/topicos/problemas#title5`. Acessado em: 15/09/2020.

pymoo (2020). Nsga-ii: Non-dominated sorting genetic algorithm. `https://pymoo.org/algorithms/nsga2.html`. Acessado em: 12/10/2020.

Rong, M., Gong, D., and Gao, X. (2019). Feature selection and its use in big data: challenges, methods, and trends. *IEEE Access*, 7, 19709–19725.

Sturup, J., Karlberg, D., and Kristiansson, M. (2015). Unsolved homicides in sweden: A population-based study of 264 homicides. *Forensic science international*, 257, 106–113.

UNODC (2013). *Global Study on Homicide*, volume 14.IV.1. United Nations publication, Sales No.

Vargas, D.E. (2018). Um estudo dos parametros do algoritmo nsga-ii com o operador sbx em problemas de otimizacao estrutural multiobjetivo. *Proceeding Series of the Brazilian Society of Computational and Applied Mathematics*, 6(2).

Wikipédia (2019). Recombinação (computação evolutiva). `https://pt.wikipedia.org/wiki/Recombinacao_(computacao_evolutiva)`. Acessado em: 20/09/2020.

Wikipédia (2020). Multi-objective optimization. `https://en.wikipedia.org/wiki/Multi-objective_optimization`. Acessado em: 12/10/2020.