Trabalho Prático I
Programação Genética

Computação Natural

Gabriel Victor Carvalho Rocha

gabrielcarvalho@dcc.ufmg.br - 2018054907

1. Introdução

O projeto se resume à implementação de um algoritmo que utiliza programação genética gerando uma função matemática, esta função calcula a distância entre duas linhas do dataframe e será usada para classificar os dados por meio de k-means.

2. Implementação

A implementação consiste na construção de uma árvore binária contendo funções matemáticas (adição, subtração, multiplicação e divisão) e terminais que correspondem à valores das linhas do dataframe.

Primeiramente é gerada a população inicial utilizando o método Ramped half-and-half, ou seja, dado que o tamanho máximo da árvore é 7, serão criadas árvores de tamanho 2 a 7 sendo metade com método Full e metade com método Grow.

Após isso se dá início à criação de novas populações. O primeiro passo é utilizar o elitismo (caso esteja habilitado) introduzindo o melhor indivíduo da antiga população na nova, em seguida será selecionado por meio de torneio os dois indivíduos que serão aplicados às operações de mutação e/ou crossover. Para a operação de mutação, se a probabilidade for alcançada, esta será feita pelo método de um ponto, alterando uma função ou um terminal da respectiva árvore. Já para crossover, se a probabilidade também for alcançada, as subárvores (escolhidas aleatoriamente) das duas árvores vencedoras do torneio serão trocadas. Alguns critérios devem ser respeitados, como: As duas subárvores não poderão ser as árvores completas, para que a operação não se torne apenas um swap; e que a altura máxima seja respeitada.

Isso será feito até que se chegue à geração máxima definida ou que o fitness de alguma árvore seja igual a 1 (máximo).

3. Experimentos

Os experimentos foram feitos para 30 gerações e utilizando populações de tamanho 36. Essa escolha foi feita, pois o tempo de execução para valores maiores do que esses eram inviáveis e exigiam muito tempo para executá-los. Mesmo escolhendo um tamanho pequeno como 36, a duração da execução ainda foi bastante longa.

Ademais, todos os testes foram feitos uma única vez, pois era totalmente impraticável executar mais ocorrências a fim de gerar as médias e desvios padrões solicitados considerando a lentidão no cálculo da fitness pelo método k-means disponibilizado.

Para base de dados GLASS

1. População = 36, Gerações = 30, Torneio = 2, Pc = 0.9, Pm = 0.05 e elitismo

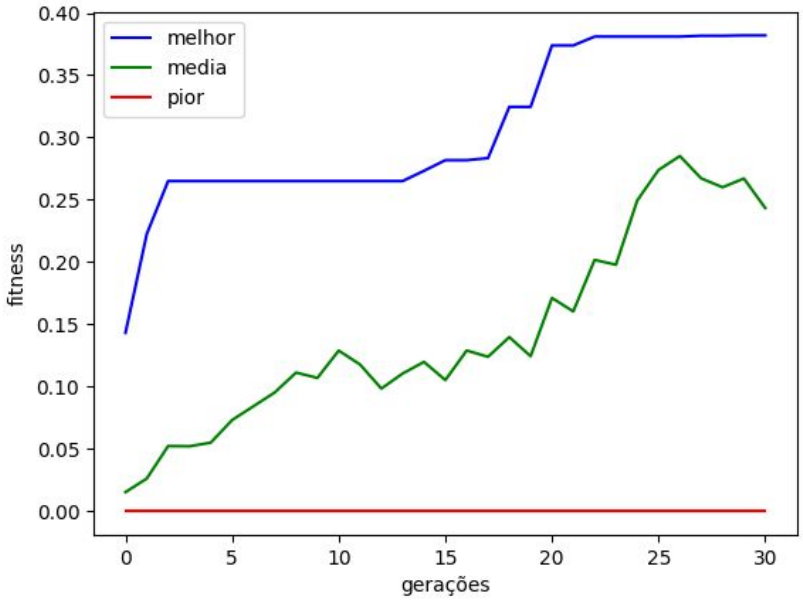

Grafico 1

Melhor: 0.38194175873942326

Pior: 9.063402944912141e-16

Media: 0.24344203797931865

Fitness do teste: 0.4468634751570927

Total filhos melhores que os pais: 304

Total filhos piores que os pais: 632

Total mutações melhores que os pais: 9

2. População = 36, Gerações = 30, Torneio = 2, Pc = 0.6, Pm = 0.3 e elitismo

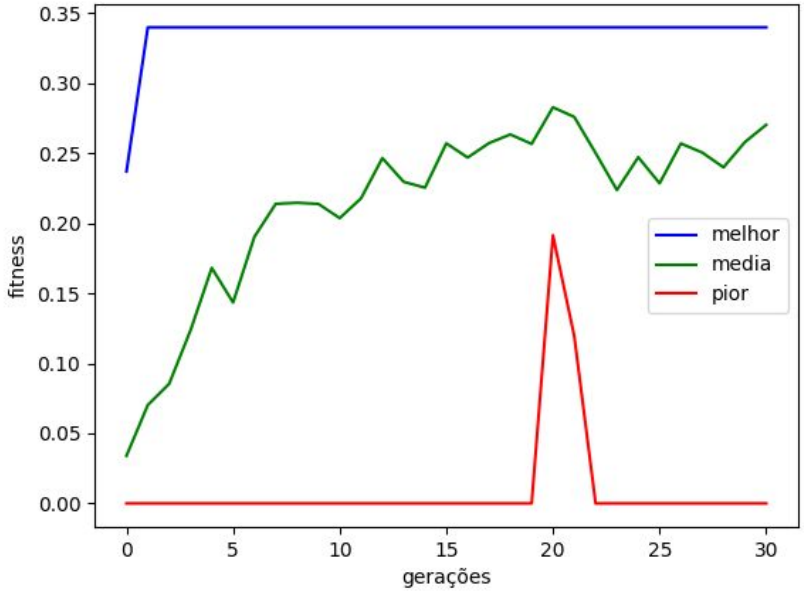

Grafico 2

Melhor: 0.33990794505766553

Pior: 9.063402944912141e-16

Media: 0.27035962246295203

Fitness do teste: 0.1367978479962033

Total filhos melhores que os pais: 206

Total filhos piores que os pais: 418

Total mutações melhores que os pais: 55

Pelo caso 1 é possível perceber que por possuir probabilidade de crossover maior, são gerados mais filhos piores como também filhos melhores em comparação com o caso 2, porém no caso 2 se gera mais mutações boas do que o anterior. Com isso, o valor final de crossover e mutação pode ser um misto entre os dois casos, aumentando a probabilidade de mutação, como Pc = 0.9 e Pm = 0.10 ou Pm = 0.15.

3. População = 36, Gerações = 30, Torneio = 5, Pc = 0.9, Pm = 0.05 e elitismo

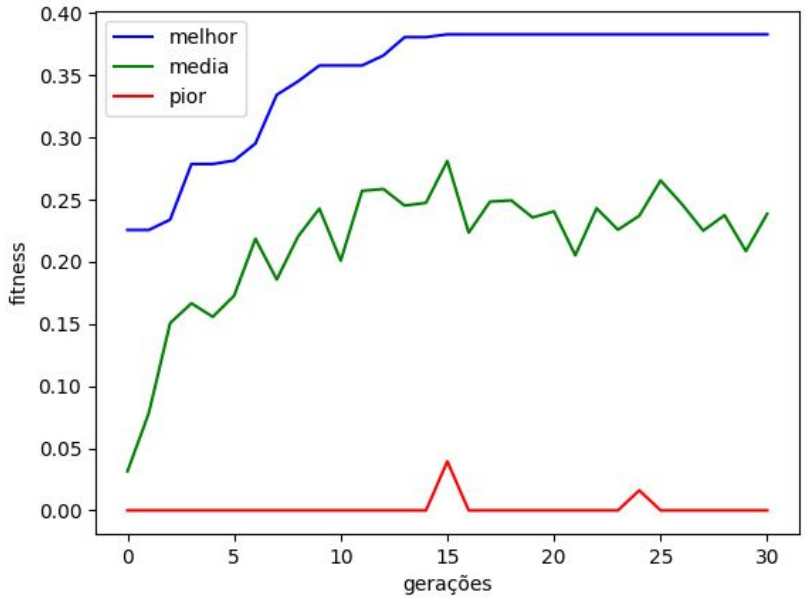

Grafico 3

Melhor: 0.38287241418718637

Pior: 9.063402944912141e-16

Media: 0.23857089065954531

Fitness do teste: 0.12121496635802648

Total filhos melhores que os pais: 150

Total filhos piores que os pais: 848

Total mutações melhores que os pais: 2

Entre os outros casos testados, este obteve o indivíduo com maior fitness no fim da geração, porém com um valor bem próximo do primeiro.

4. Melhor anterior: População = 36, Gerações = 30, Torneio = 5, Pc = 0.9, Pm = 0.05 e sem elitismo:

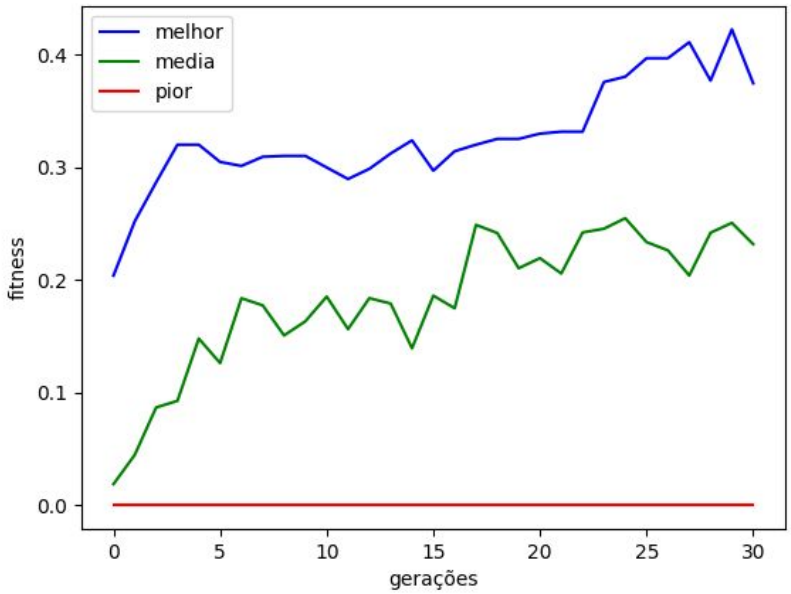

Grafico 4

Melhor: 0.37488971001968685

Pior: 9.063402944912141e-16

Media: 0.23199447708646337

Fitness do teste: 0.5516317404370641

Total filhos melhores que os pais: 206

Total filhos piores que os pais: 799

Total mutações melhores que os pais: 1

Os resultados não são os mesmos quando comparados com os resultados do caso em que se utiliza elitismo, porém esses valores foram bem próximos.

Para base de dados BREAST CANCER

O parâmetro utilizado que obteve o resultado mais apropriado foi o explicitado no caso 3 da base de dados GLASS: População = 36, Gerações = 30, Torneio = 5, Pc = 0.9, Pm = 0.05 e elitismo. Entretanto, todos os 4 casos foram experimentados para se ter uma análise mais detalhada, como pode-se ver abaixo:

1. População = 36, Gerações = 30, Torneio = 2, Pc = 0.9, Pm = 0.05 e elitismo

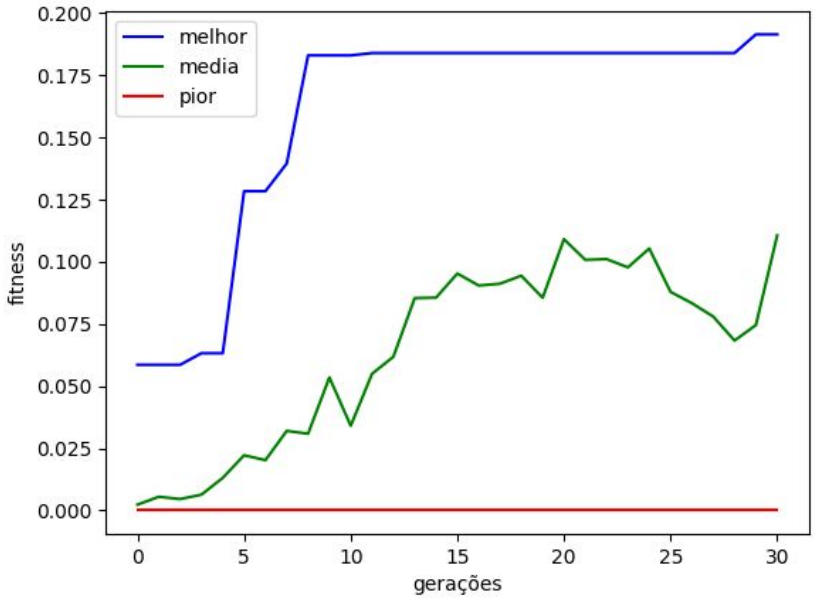

Grafico 5

Melhor: 0.1914528587839511

Pior: 6.486687860771986e-16

Media: 0.11059330392107487

Fitness do teste: 0.19769959815999544

Total filhos melhores que os pais: 283

Total filhos piores que os pais: 623

Total mutações melhores que os pais: 15

2. População = 36, Gerações = 30, Torneio = 2, Pc = 0.6, Pm = 0.3 e elitismo

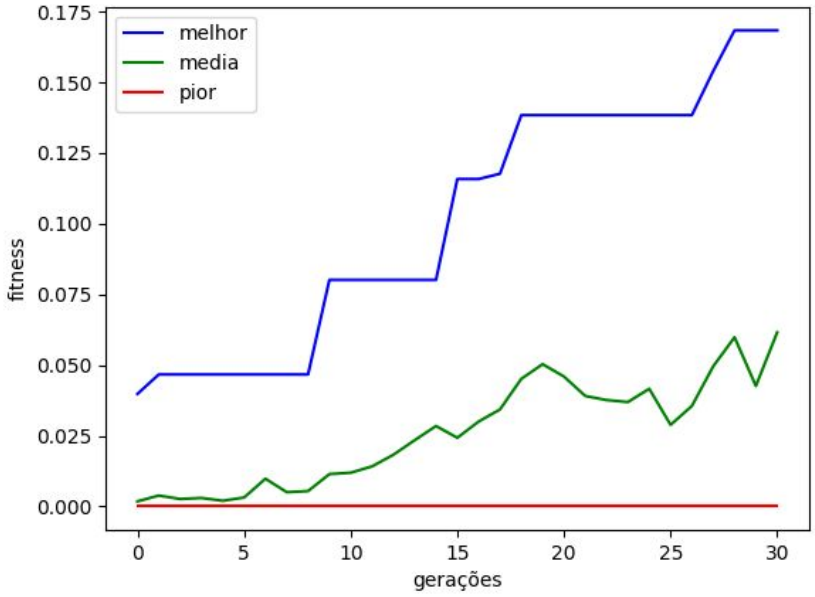

Grafico 6

Melhor: 0.16837957998114989

Pior: 6.486687860771986e-16

Media: 0.06157447895695165

Fitness do teste: 0.020931528470560363

Total filhos melhores que os pais: 174

Total filhos piores que os pais: 435

Total mutações melhores que os pais: 59

3. População = 36, Gerações = 30, Torneio = 5, Pc = 0.9, Pm = 0.05 e elitismo

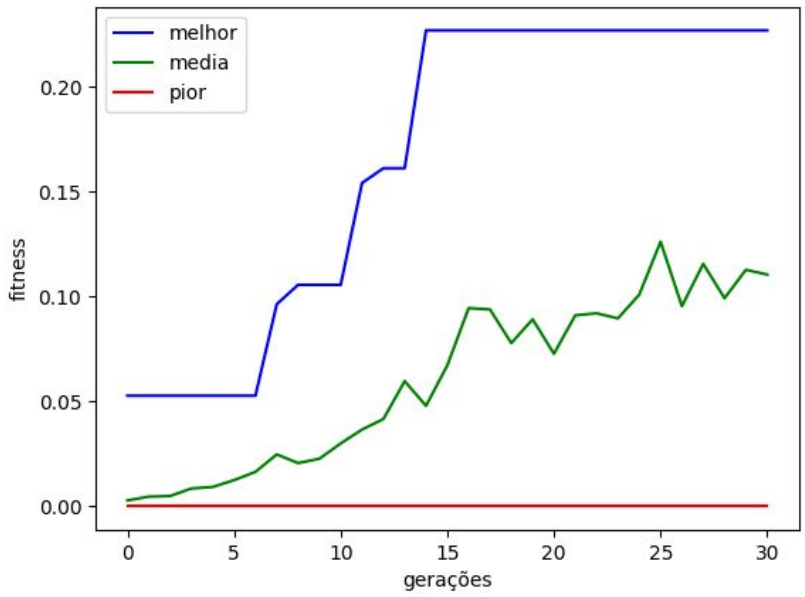

Grafico 7

Melhor: 0.2269377666679807

Pior: 6.486687860771986e-16

Media: 0.11052565519840787

Fitness do teste: 0.12562537992157202

Total filhos melhores que os pais: 98

Total filhos piores que os pais: 739

Total mutações melhores que os pais: 2

Utilizando o melhor parâmetro encontrado na base de dados GLASS, também obteve o melhor resultado entre todos os outros.

4. Melhor anterior: População = 36, Gerações = 30, Torneio = 5, Pc = 0.9, Pm = 0.05 e sem elitismo

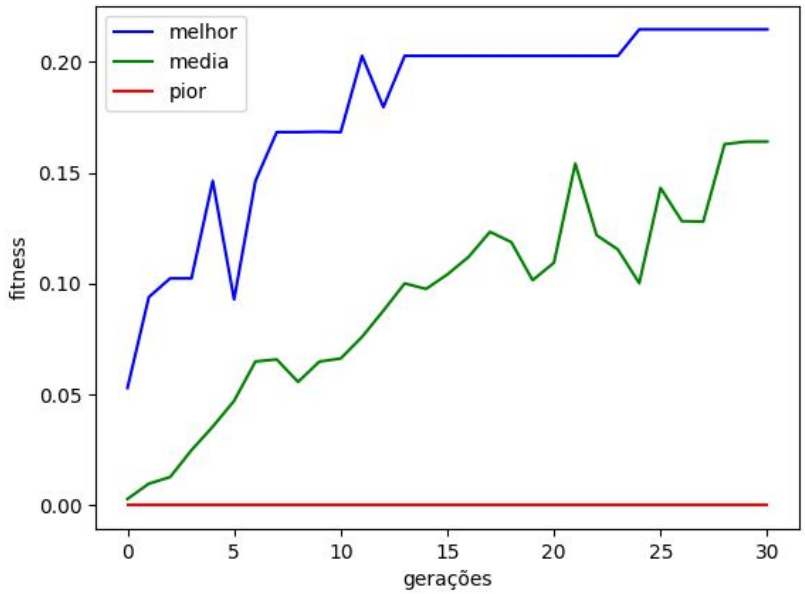

Grafico 8

Melhor: 0.21472999444244614

Pior: 6.486687860771986e-16

Media: 0.16412749727879283

Fitness do teste: 0.029944596847387952

Total filhos melhores que os pais: 117

Total filhos piores que os pais: 674

Total mutações melhores que os pais: 0

Novamente, os resultados não são os mesmos quando comparados com os resultados do caso em que se utiliza elitismo, porém esses valores foram bem próximos.

4. Conclusão

Ao final do processo de implementação e experimentação pode-se analisar que os resultados não foram bons o suficiente, talvez pela quantidade baixa de população e por não ser executado em muitas gerações, impactando assim a solução final. Entretanto, não havia outra solução a não ser utilizar tais valores, levando em conta que para valores maiores iria demandar horas ou até mesmo dias para finalizar.

Por sua vez, ainda é possível perceber que o algoritmo evolui de fato ao longo das gerações. Otimizando o cálculo da fitness e utilizando populações maiores, possivelmente o algoritmo gere em tempo hábil valores melhores do que os apresentados.

5. Bibliografia

PAPPA, Gisele. Slides: Busca e Decisões de Design e Experimentação. PDF disponibilizado via Moodle UFMG.

PAPPA, Gisele. Slides - Programação Genética baseada em Gramáticas: Parte 1. PDF disponibilizado via Moodle UFMG.

PAPPA, Gisele. Slides - Programação Genética baseada em Gramáticas: Parte 2. PDF disponibilizado via Moodle UFMG.