

Trabalho Prático 1
Programação Genética

Data de Entrega: 13 de novembro de 2024

1 Introdução

O principal objetivo deste trabalho é desenvolver conceitos chave para a construção de soluções para problemas usando Programação Genética (GP), envolvendo o entendimento e a implementação dos componentes básicos de um arcabouço de GP, bem como a análise de sensibilidade dos seus parâmetros (como eles afetam o resultado final, a natureza da convergência, etc) e procedimentos para avaliação das soluções alcançadas.

Uma dos problemas mais populares que podem ser resolvidos com técnicas de programação genética é a regressão simbólica. Conforme visto nos vídeos da disciplina, dado um conjunto de m amostras provenientes de uma função *desconhecida* $f : \mathbb{R}^n \mapsto \mathbb{R}$, representadas por uma dupla $\langle X, Y \rangle$ onde $X \in \mathbb{R}^{m \times n}$ e $Y \in \mathbb{R}^m$, o objetivo é encontrar a expressão simbólica de f que melhor se ajusta às amostras fornecidas.

Neste trabalho você utilizará a técnica de regressão simbólica para criar uma função $d(e_i, e_j)$, que representa a função de distância d entre pares de exemplos $\langle e_i, e_j \rangle$ de uma base de dados qualquer. Esta função será então utilizada por um algoritmo de agrupamento de dados (*clustering*).

Algoritmos de agrupamento (ou *clustering*) recebem como entrada um conjunto de dados e identificam, a partir destes dados, um conjunto de grupos de exemplos (ou instâncias) que tem características similares. Ao contrário das tarefas de regressão ou classificação, onde o rótulo que se quer prever é conhecido para os exemplos de treinamento, na tarefa de agrupamento não se sabe a priori o tipo de grupos que se quer encontrar. Por isso, algoritmos de agrupamento criam grupos com base em uma medida de distância. O objetivo é que exemplos similares (com distância pequena entre si) estejam no mesmo grupo, e exemplos distantes estejam em grupos diferentes.

A maioria dos algoritmos de *clustering* utiliza medidas simples de distância, como a distância Euclidiana, que sofrem de problemas com a maldição da dimensionalidade.¹ Por isso, encontrar uma função de distância apropriada para uma dada base de dados pode melhorar muito a qualidade dos grupos encontrados pelo algoritmo. Além disso, nesse caso o algoritmo de programação genética funciona também como um seletor de atributos, uma vez que ele é capaz de utilizar na medida sendo evoluída apenas os atributos que considerar mais relevantes (você não deve se preocupar com isso, o algoritmo faz tudo pra você!) Como para esse problema não se conhecem *a priori* os grupos que se quer prever, algoritmos de agrupamento normalmente trabalham otimizando métricas que chamamos de similaridade intra- e inter-cluster. Porém, para facilitar o desenvolvimento deste trabalho, trabalharemos com bases de

¹https://en.wikipedia.org/wiki/Curse_of_dimensionality#:~:text=The%20curse%20of%20dimensionality%20refers,was%20coined%20by%20Richard%20E.

dados onde conhecemos o rótulo, mas este será ignorado durante a construção do modelo e apenas considerado para avaliação da função de distância sendo evoluída.

Assim, para avaliar a função sendo evoluída, precisaremos rodar um algoritmo de clustering. Aqui trabalharemos com um algoritmo de clustering aglomerativo. Nesse tipo de algoritmo, realiza-se um agrupamento hierárquico onde, inicialmente, cada exemplo é considerado um clustering, e esses clusters são unidos até que cada exemplo pertença a um único clustering². Para usar a função sendo evoluída no algoritmo, você precisará trocar sua função de distância. A versão do algoritmo de clustering aglomerativo do Scikit-Learn inclui um parâmetro relativo a distância que será utilizada, que pode ser facilmente substituída por uma matriz de distâncias³.

Assim, cada função representada por um indivíduo deve ser usada para calcular uma matriz de distância entre as instâncias da base de dados e passada para o algoritmo de clustering (parâmetro *metric*), cujos resultados serão então avaliados. Dado que conhecemos o rótulo verdadeiro, iremos utilizar como métrica de avaliação (fitness) a medida V . Esta medida é definida pela média harmônica das medidas de homogeneidade e completude, conforme definido na Eq. 1, onde β normalmente assume valor 1 e representa a proporção do peso atribuído a cada uma das duas medidas combinadas. Valores de $\beta > 1$ privilegiam a completude, enquanto valores de $\beta < 1$ privilegiam a homogeneidade.

$$V = \frac{(1 + \beta)homogeneidade \times completude}{\beta \times homogeneidade + completude} \quad (1)$$

Um grupo é dito homogêneo quando todos os exemplos pertencentes a ele estão associados a uma mesma classe. A medida de homogeneidade assume valor 1 (seu valor máximo) quando todos os grupos possuem apenas exemplos de uma mesma classe. A medida de completude, por sua vez, atinge seu valor máximo quando todos os exemplos de uma mesma classe pertencem a um mesmo grupo. As duas medidas retornam valores no intervalo $[0,1]$. A medida V também está disponível na biblioteca Scikit-Learn.⁴ Um exemplo do uso da métrica também está disponível no Moodle.

Decisões de Implementação:

1. Como representar um indivíduo (genótipo), que é uma função de distância;
2. Como gerar a população inicial;
3. Quais operadores genéticos serão utilizados;
4. Facilidades para variação de parâmetros—parâmetros *hardcoded* no arcabouço certamente dificultarão a avaliação dos parâmetros;

2 Bases de Dados

Dois conjuntos de dados serão utilizados neste trabalho, conforme descritos na Tabela 1, e estão disponíveis no Moodle. Todos os arquivos estão no formato CSV, e a última coluna

²<https://medium.com/@khalidassalafy/agglomerative-hierarchical-clustering-a-study-and-implementation-in-python>

³<https://scikit-learn.org/dev/modules/generated/sklearn.cluster.AgglomerativeClustering.html>

⁴https://scikit-learn.org/stable/modules/generated/sklearn.metrics.v_measure_score.html#sklearn.metrics.v_measure_score

Tabela 1: Bases de dados.

Base de dados	# Atributos	# Exemplos	# Classes (k)
Breast cancer ⁵	9	116	2
Wine quality ⁶	11	1599	7

contém a saída desejada (y). Esta saída deverá ser comparada com a saída estimada, que corresponde ao cluster a que o exemplo será associado.

Cada base de dados possui dois arquivos:

1. <nome-da-base>-train.csv
2. <nome-da-base>-test.csv

O primeiro deverá ser usado para evoluir as soluções até o número máximo de gerações ser alcançado. Finalizada esta etapa, as melhores soluções encontradas deverão ser avaliadas utilizando a base de teste, usando novamente a medida V . A ideia do teste é, dado que chegam exemplos novos, associá-los a um dos grupos já criados.

3 Metodologia Experimental

O GP deve ser testado nas 2 bases de dados descritas na Tabela 1. A avaliação experimental descrita abaixo deve ser feita para uma das bases. Os parâmetros considerados mais apropriados podem ser novamente utilizados para a outra base.

A parte de escolha e estudo dos parâmetros deve ser feita da seguinte forma:

- Definir o tamanho máximo do indivíduo como 7. Esse parâmetro não precisa ser obrigatoriamente variado.
- Escolher o tamanho da população e o número de gerações apropriados. O tamanho da população pode ser testado, por exemplo, utilizando 30, 50, 100, 500 indivíduos. O número de gerações pode também ser escolhido usando esses mesmos números. Mas como saber se o escolhido é o mais apropriado? Vocês podem avaliar como o aumento no número da população ou de gerações melhora a solução encontrada (em termos do erro gerado), se a população converge, etc. Considere também o custo computacional nessa escolha.
- Testar duas configurações de parâmetros para crossover e mutação. Na primeira, a probabilidade de crossover (p_c) deve ser alta (por exemplo, 0.9), e a probabilidade de mutação (p_m) deve ser baixa (por exemplo, 0.05). Na segunda, p_c deve ser mais baixa (por exemplo, 0.6) e p_m mais alta (por exemplo, 0.3). Para ambas as configurações, deve-se avaliar o efeito do crossover e da mutação na evolução, isto é, em quantos casos esses operadores contribuem positivamente (os filhos gerados são melhores que os pais) ou negativamente para a evolução? A partir desse estudo inicial, que valores finais você proporia?
- Analisar as mudanças ocorridas quando o tamanho do torneio aumenta de 2 para 5 ou 3 para 7, dependendo do tamanho da população, e considerando a pressão seletiva.
- Utilizar elitismo.

- Existe uma forma simples de medir bloating no seu algoritmo?

Lembrem-se que ao mexer em um dos parâmetros, todos os outros devem ser mantidos constantes, e que a análise dos parâmetros é de certa forma interativa. A configuração de parâmetros raramente vai ser ótima, mas pequenos testes podem melhorar a qualidade das soluções encontradas.

Por ser um método estocástico, a avaliação experimental do algoritmo baseado em GP deve ser realizada com *repetições*, de forma que os resultados possam ser reportados segundo o valor médio obtido e o respectivo desvio-padrão. A realização de 10 repetições pode ser um bom ponto de partida (lembrando que desvio-padrão alto sugere um maior número de repetições).

Ao final, você deve reportar tanto o valor da medida V no treino quanto no teste. Mostre a média das 10 repetições e o desvio padrão, e reporte também o valor de V da melhor solução encontrada. Para escolher a melhor solução você deve usar o valor de V do treino, e reportá-lo juntamente com o V do teste. Note que nem sempre o maior V do teste será o do treino, mas na teoria, você só conhece o teste depois e esses dados não podem ser usados para guiar nenhuma decisão relativa ao algoritmo.

Guia para execução dos experimentos

1. Escolha do tamanho da população e número de gerações (utilizar tamanho máximo do indivíduo como 7, elitismo, torneio de tamanho 2 e $p_c = 0.9$ e $p_m = 0.05$).
2. Após alguns testes, defina o tamanho da população e o número de gerações e varie p_c e depois p_m . Os parâmetros escolhidos no passo 1 ainda são apropriados?
3. Definidos o tamanho da população, número de gerações, p_c e p_m , aumente o tamanho do torneio.
4. Escolha os melhores parâmetros dos anteriores e retire o elitismo. Os resultados obtidos são os mesmos?
5. Se desejar, teste outras características, como métodos para garantir a diversidade da população.

Estatísticas importantes

Estas estatísticas devem ser coletadas para todas as gerações.

1. Fitness do melhor e pior indivíduos
2. Fitness média da população
3. Número de indivíduos repetidos na população
4. Número de indivíduos gerados por crossover melhores e piores que a fitness média dos pais

O que deve ser entregue...

- Código fonte do programa
- Documentação do trabalho:
 - Introdução
 - Implementação: descrição sobre a implementação do programa, incluindo detalhes da representação, fitness e operadores utilizados
 - Experimentos: Análise do impacto dos parâmetros no resultado obtido pelo AE.
 - Conclusões
 - Bibliografia

A entrega DEVE ser feita pelo Moodle na forma de um único arquivo zipado, contendo o código e a documentação do trabalho.

Considerações Finais

- Os parâmetros listados para execução dos experimentos são sugestões iniciais, e podem ser modificados a sua conveniência.
- Depois da entrega do trabalho, faremos uma competição em sala de aula para avaliar as diversas decisões de implementação do algoritmo e como a otimização dos parâmetros podem levar ao sucesso ou fracasso do algoritmo.
- **Não é permitido o uso de bibliotecas prontas de GP. Você deve implementar o algoritmo completo, incluindo os métodos de seleção, cruzamento e mutação.**