[Data]

MNIST

IA - Creeperbots

8200421 – Tiago Pacheco

8190352 – Rui Campos

8190370 – Inês Bastos

Conteúdo

[Introdução 2](#_Toc136736838)

[Problema 2](#_Toc136736839)

[1. Algoritmo genético 3](#_Toc136736840)

[1.1. Deep Learning 3](#_Toc136736841)

[Configurações possíveis no algoritmo genético: 3](#_Toc136736842)

[Nota: 3](#_Toc136736843)

[Resultados: 4](#_Toc136736844)

# Introdução

No âmbito na Unidade Curricular de Inteligência Artificial foi no proposto a realização de um projeto prático no qual era pretendido criar um algoritmo genético capaz de encontrar soluções ótimas para prever o problema específico da classificação de dígitos manuscritos, na área da Visão por Computador. Nesse sentido, será usado o dataset MNIST.

Este projeto contém os conhecimentos adquiridos na Unidade Curricular de Inteligência Artificial, mais especificamente focando a área da Computação Evolucionária.

# Problema

Neste projeto era necessário criar um algoritmo genético que fosse capaz de encontrar soluções ótimas para a classificação de dígitos manuscritos.

O dataset MNIST é um dos mais conhecidos e representa um problema que regra geral é simples para os humanos resolverem, mas difícil para o computador, sendo representado por uma base de dados de dígitos escritos à mão com um conjunto de treino composto por 60 000 exemplos, e um conjunto de teste composto por 10 000 exemplos.

Cada dígito é representado por uma matriz 28x28, totalizando 784 valores de pixels em escala de cinzento, que variam entre 0 (branco) e 255 (preto). O dataset contém ainda uma coluna (label) que contém um valor entre 0 e 9 que identifica o dígito representado. Logo, trata-se, de um problema de ML supervisionado.

Portanto é importante perceber que foram considerados três tipos de algoritmos de treino (Deep Learning, Random Forest e XGBoost). Sendo que este caso o tipo de problema, é necessário algoritmos capazes de treinar modelos multinomiais, logo foram selecionados os previamente mencionados.

Além disto foi também criada uma leaderboard onde tanto nós como as restantes equipas ao longo do semestre íamos submetendo as nossas melhores soluções.

# Algoritmo genético

No desenvolvimento de um algoritmo genético é necessária a implementação de 3 operadores genéticos: Seleção, Crossover e Mutação. A seleção consiste na escolha de determinados elementos da geração atual para uma “mating pool”, estes elementos tornar-se-ão um dos pais dos elementos das gerações seguintes, existem vários tipos de métodos de seleção, o nosso grupo decidiu implementar os seguintes: Fitness Proportionate Selection, Roulette Wheel Selection e Tournament Selection usados no Deep Learning.

O crossover consiste na junção de cromossomas (pais) da geração atual, para criar dois cromossomas assim como acontece na biologia. Como acontecia na seleção também existem vários tipos de métodos de crossover, nomeadamente três: One Point Crossover, Uniform Crossover e Multi Point Crossover usados no Deep Learning.

A mutação consiste na alteração de um gene, de um cromossoma, por outro, assim como o crossover é algo que acontece regularmente na biologia.

## Deep Learning

### Configurações possíveis no algoritmo genético:

- population\_size = 20

- generations = 5

- mutation\_rate = 0.1

- uniform\_crossover\_rate = 0.5

- one\_point\_crossover\_rate = 0.5

- multi\_point\_min\_crossover\_points\_rate = 0.25

- multi\_point\_max\_crossover\_points\_rate = 0.75

- crossover\_type = "uniform"

- selection\_type = "fitness\_selection"

- tournament\_size = 3

### Nota:

O responsável pelo algoritmo Deep Learning (Rui Campos) perdeu muito tempo para consegui fazer o código funcionar por causa da biblioteca H2O que requer a versão do java estar entre o 8 e o 13 quando este estava a usar a versão 17.

Encontrou também problemas ao fazer o export do ficheiro com as previsões uma vez que o programa Excel o qual estava a usar para ver os dados ocultava por defeito os valores que de pois da vírgula tinha um “0”, induzindo em erro que algum problema tinha ocorrido.

Após ter sido realizado um teste para saber aproximadamente o tempo de demora do treino do modelo, submeteu-se uma proposta no Kaggle com as seguintes definições:

Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamente

### Resultados:

Uma imagem com texto, captura de ecrã, preto e branco, Tipo de letra

Descrição gerada automaticamente

Uma imagem com texto, captura de ecrã, preto e branco, Tipo de letra

Descrição gerada automaticamente