# Árvores de Decisão com Pruning

Relatório da 1ªParte do Trabalho Prático

Aprendizagem Automática

2020/2021

Docente: Prof. Teresa Gonçalves e Prof. Luís Rato

Discentes: Leonardo Catarro, nº43025; Diogo Solipa, nº 43071

Licenciatura: Engenharia informática

Índice

[Árvores de Decisão com Pruning 1](#_Toc60240451)

[Objetivo 3](#_Toc60240452)

[Análise de Desempenho 3](#_Toc60240453)

[Problemas Encontrados 7](#_Toc60240454)

# Objetivo

Este trabalho tem como objetivo implementar em Python uma classe para gerar árvores de decisão com “Pruning”, usando as diversas medidas de impureza: Gini, Entropia e Erro e REP (Reduced Error Pruning) como método de poda da árvore.

# Análise de Desempenho

- Análise das árvores com e sem Pruning, **não fizemos pruning!**

🡪Diversas Funções de Homogeneidade/Pureza:

No desenvolvimento deste trabalho, foram usadas as funções da Entropia, Gini e Erro, como formas do cálculo da impureza dos dados fornecidos.

De entre as três, a Entropia será a que possui pior desempenho, daí maior complexidade temporal, isto porque, além da sua complexidade resultante de iteração pelas listas que armazenam os dados, ainda envolve o cálculo com funções logarítmicas, que são computacionalmente complexas

O método que utiliza o cálculo do Gini, poderá ser considerado o intermédio, apenas envolve produtos e cálculo fracionário

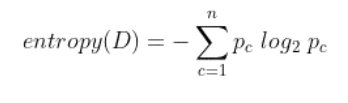
Finalmente, o Erro, será o que possui melhor desempenho.

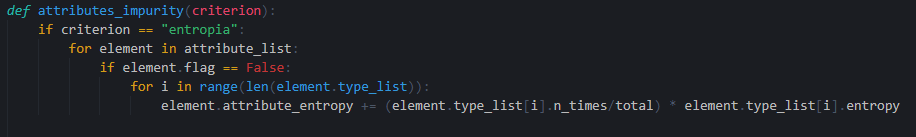
**As fórmulas matemáticas e respetivas implementações são as seguintes:**

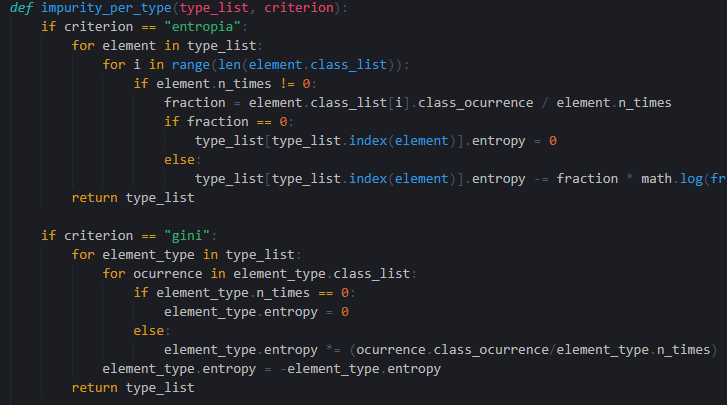
(Nota: Implementação de Gini, Entropia e Erro, todas dentro das funções: attributes\_impurity(criterion) e impurity\_per\_type (type\_list, criterion)

Nota2: Para cada método de cálculo de impureza dos dados, nós possuímos uma implementação para o cálculo por tipos e outra para os atributos, em funções separadas, mas que complementam uma á outra.

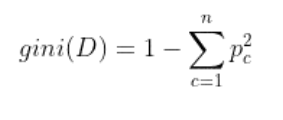
1. Entropia: entropia para dados nominais (envolvem mais que 2 classes)

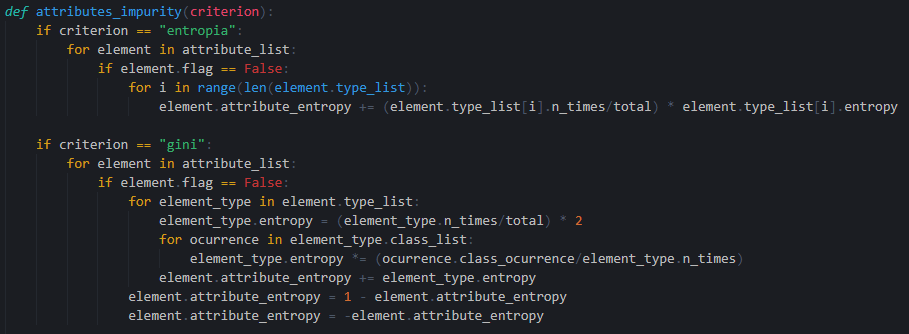




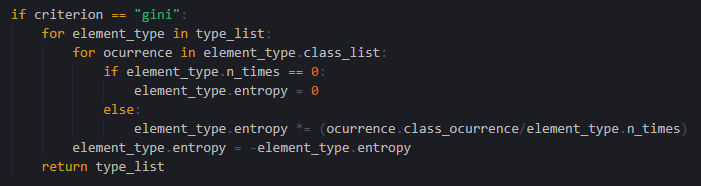


1. Gini: (para dados nominais)





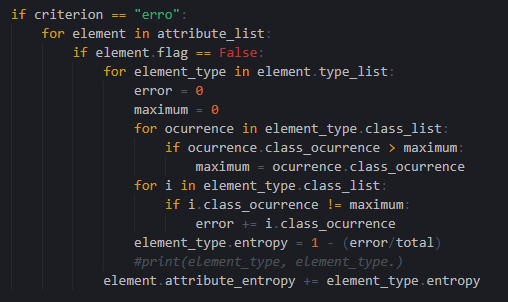
Legenda: Função attributes\_impurity ()

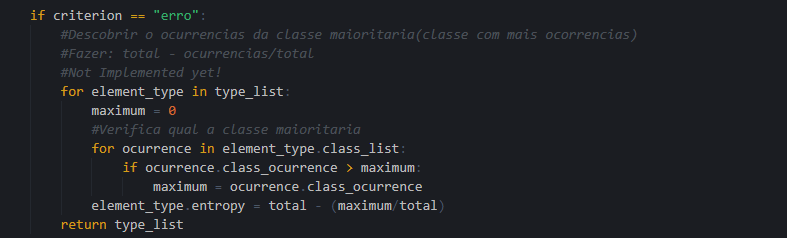


Legenda: Função attributes\_impurity ()

1. Classe Minoritária (Erro):







Legenda: Função attributes\_impurity ()

Legenda: Função impurity\_per\_type ()

# Problemas Encontrados

Ao longo do desenvolvimento deste trabalho fomos nos deparando e enfrentando alguns problemas.

Inicialmente, o armazenamento dos dados, optamos pela programação orientada por objetos, o que para nós nos pareceu a melhor forma para o armazenamento dos dados. Porém, com o desenvolvimento percebemos que talvez não tenha sido a melhor escolha, mas ainda assim conseguimos desenvolver o que era pedido, com a nossa ideia inicial.

De seguida, e talvez o maior problema enfrentado, foi na parte da função recursiva (recursive\_split ()), função esta que constrói a árvore, tratando os dados desde a raiz às folhas. Obrigamos a função a fazer mais iterações após atualizarmos o nó e não necessariamente casos mais generalizados para recursão

Por último, o desenvolvimento do Pruning, onde tivemos bastantes dificuldades, derivado às anteriormente mencionadas, em implementar a poda. Fazendo assim com a nossa árvore apensas seja construída sem Pruning.

Conclusão

A dificuldade deste trabalho afetou o seu desenvolvimento, tornando-o mais lento, porém conseguimos chegar ao resultado desejado