Lista 04
Disciplina: Inteligência Artificial
Profa.: Cristiane Neri Nobre
Data: 21/09/2025
Aluna: Alessandra Faria Rodrigues
Matrícula: 828333

1 INTRODUÇÃO

As árvores de decisão são algoritmos amplamente utilizados em aprendizado de máquina supervisionado, aplicados tanto em problemas de **classificação** quanto de **regressão**. Neste trabalho foram implementados, do zero em Python, três algoritmos de árvores de decisão: **ID3**, **C4.5** e **CART**, sem utilizar bibliotecas prontas de árvores. Apenas as bibliotecas pandas e numpy foram empregadas para manipulação de dados.

O objetivo foi compreender na prática os fundamentos matemáticos desses algoritmos, como **entropia**, **ganho de informação**, **razão de ganho** e **índice Gini**, e observar suas diferenças na construção da árvore.

A biblioteca desenvolvida foi validada através da sua aplicação em dois datasets: um dataset simples e puramente categórico ("Play Tennis") para verificação inicial da lógica, e o dataset complexo e realista do "Titanic" para testar a robustez, a capacidade de lidar com diferentes tipos de dados e a eficácia das técnicas de poda implementadas.

2 ALGORITMOS DE ÁRVORE DE DECISÃO

2.1 ALGORITMO ID3

- **Critério de Divisão:** Ganho de Informação (Information Gain), baseado na Entropia. A Entropia mede o grau de "impureza" ou "incerteza" em um conjunto de dados. O ID3 escolhe o atributo que, ao dividir os dados, resulta na maior redução da Entropia.
- **Características:** Funciona apenas com atributos categóricos, cria galhos para cada valor do atributo (divisão múltipla) e é suscetível a favorecer atributos com muitos valores distintos.

A **entropia** é definida como:

$$H(S) = -\sum_{i=1}^{k} p_i \log_2(p_i)$$

Onde:

- S é o conjunto de exemplos,
- k é o número de classes,
- p_i é a proporção de elementos da classe i.

Quanto mais homogêneo o conjunto (ou seja, exemplos pertencendo à mesma classe), menor a entropia.

O **ganho de informação** mede quanto a incerteza diminui ao dividir os dados por um atributo A:

$$ganho(atributo) = Entropia(classe) - Entropia(atributo)$$

O atributo com maior ganho de informação é escolhido como nó da árvore. O processo é repetido recursivamente até que os dados fiquem puros ou não haja mais atributos.

2.2 ALGORITMO C4.5

- **Critério de Divisão:** Razão de Ganho (Gain Ratio). É uma melhoria do Ganho de Informação que normaliza o resultado, penalizando atributos com muitos valores e resolvendo uma das principais desvantagens do ID3.
- **Características:** Também realiza divisão múltipla e é a base para a implementação do ID3 neste projeto. A principal diferença reside no cálculo para a escolha do atributo.

O C4.5 utiliza a **razão de ganho**, que normaliza o ganho de informação pelo valor chamado **Split Information**:

$$GainRatio(S, A) = \frac{InformationGain(S, A)}{SplitInfo(S, A)}$$

Onde:

$$SplitInfo(S, A) = -\sum_{v \in Valores(A)} \frac{|Sv|}{|S|} \log_2\left(\frac{|Sv|}{|S|}\right)$$

2.3 ALGORITMO CART

- **Critério de Divisão:** Índice de Gini. O Índice de Gini mede a probabilidade de um elemento ser classificado incorretamente se fosse rotulado aleatoriamente de acordo com a distribuição das classes no nó. O CART busca a divisão que resulta no menor Índice de Gini ponderado.
- **Características:** Produz exclusivamente árvores binárias (cada nó tem exatamente dois galhos). É versátil, funcionando tanto para classificação quanto para regressão, e lida de forma robusta com atributos numéricos e categóricos.

  O CART utiliza o **índice Gini** para medir a impureza de um nó:

$$Gini(S) = 1 - \sum_{i=1}^{k} P_i^2$$

Se todos os exemplos pertencem a uma única classe Gini=0 (puro), e quanto mais misturado, maior o valor de Gini.

3 DETALHES DA IMPLEMENTAÇÃO

A implementação foi feita em Python, dividindo o projeto em módulos:

- no.py (Estrutura da Árvore):
  - A classe NoArvore é o bloco de construção fundamental. Cada instância representa um nó (seja de decisão ou uma folha). Seus atributos principais (atributo, criterio_divisao, resultado, filhos) foram projetados para serem flexíveis o suficiente para suportar a lógica binária do CART e a lógica de múltiplos galhos do ID3/C4.5.
- funcoes.py (Cálculos Matemáticos):
  - ste módulo contém as implementações puras dos critérios de divisão: indice_gini, entropia, ganho_informacao e gain_ratio.
  - A função melhor_divisao_cart foi implementada para encontrar o melhor ponto de corte para atributos numéricos, varrendo os valores e testando os pontos médios para minimizar o Gini.
- Implementação dos Algoritmos (id3.py, c45.py, cart.py):
  - Todos os três algoritmos foram implementados usando uma abordagem recursiva.
  - **ID3/C4.5:** A recursão percorre os atributos, e após um atributo ser usado para uma divisão, ele é removido da lista de candidatos para os nós filhos daquele galho.

- **CART:** A recursão também percorre os atributos, mas a implementação permite que atributos numéricos sejam reutilizados em um mesmo galho com diferentes pontos de corte.
- **Poda (Pruning):** Para evitar o overfitting e controlar o tamanho das árvores, foram implementadas técnicas de pré-poda. As funções id3, c45 e cart foram aprimoradas para aceitar os parâmetros max_depth (profundidade máxima) e min_samples_leaf (número mínimo de amostras por folha), que servem como condições de parada para a recursão. No CART, também foi adicionado min_impurity_decrease para garantir que uma divisão só ocorra se trouxer uma melhora mínima na pureza.

4 APLICAÇÃO NO DATASET TITANIC

Para validar a biblioteca, foi realizado um experimento completo com o dataset do Titanic.

- **Pré-processamento:**
  - **Para CART:** O foco foi converter todos os dados para um formato numérico. Colunas irrelevantes (PassengerId, Name, etc.) foram removidas. Sex foi mapeado para 0/1. Embarked foi convertido usando get_dummies. Valores faltantes em Age foram preenchidos com a mediana.
  - **Para ID3/C4.5:** O foco foi converter todos os dados para um formato categórico. Além da limpeza inicial, os atributos contínuos Age e Fare foram discretizados em faixas (ex: 'Criança', 'Adulto'; 'Baixa', 'Média') usando a função pandas.cut.

- **Metodologia de Avaliação:**
  - Os dados foram divididos em conjuntos de treino (80%) e teste (20%) para simular um cenário realista.
  - Os modelos foram treinados exclusivamente com o conjunto de treino.
  - O desempenho foi medido no conjunto de testes usando as métricas de Acurácia e Precisão.

5 RESULTADOS E DISCUSSÃO

Após o treinamento e avaliação, os modelos apresentaram os seguintes resultados no conjunto de teste:

- **Modelo ID3:**
  - Acurácia: 82.12%
  - Precisão: 78.95%

- **Modelo C4.5:**
    - Acurácia: 83.24%
    - Precisão: 87.23%
- **Modelo CART:**
    - Acurácia: 83.80%
    - Precisão: 79.03%

A análise dos gráficos gerados revelou que todos os modelos aprenderam padrões lógicos e historicamente consistentes. O atributo Sex foi consistentemente escolhido como o divisor mais importante na raiz da árvore, refletindo a política de "mulheres e crianças primeiro". Em seguida, atributos como Pclass (classe social) e Age (idade) foram usados para refinar as previsões. A implementação das técnicas de poda foi crucial para gerar árvores menores, mais interpretáveis e que evitam o overfitting.

Agora, fazendo uma comparação da estrutura e visualização das árvores, uma observação importante na análise dos resultados foi a notável diferença na interpretabilidade visual entre a árvore binária gerada pelo CART e as árvores de múltiplos galhos do ID3 e C4.5.

A estrutura binária do CART, onde cada nó de decisão se divide em exatamente dois galhos (correspondendo a uma condição "Sim/Verdadeiro" ou "Não/Falso"), demonstrou ser significativamente mais fácil de visualizar e seguir. O caminho da raiz até uma folha em uma árvore CART se assemelha a um fluxograma claro, onde cada passo é uma decisão simples e atômica. Isso reduz a carga cognitiva necessária para entender a lógica do modelo.

Em contraste, as árvores geradas pelo ID3 e C4.5, com sua capacidade de criar múltiplos galhos para cada categoria de um atributo (divisão múltipla), resultaram em gráficos visualmente mais largos e densos. Por exemplo, enquanto o nó Pclass na árvore ID3/C4.5 se dividiu em três galhos distintos ('1', '2', '3'), o CART realizou uma série de divisões binárias mais simples (ex: Pclass <= 2.5?) para alcançar uma segmentação similar dos dados.

Embora ambas as abordagens sejam eficazes para modelar os dados, a simplicidade visual da árvore binária do CART representa uma vantagem prática, especialmente ao apresentar o modelo para uma audiência ou ao realizar uma análise exploratória das regras de decisão geradas.

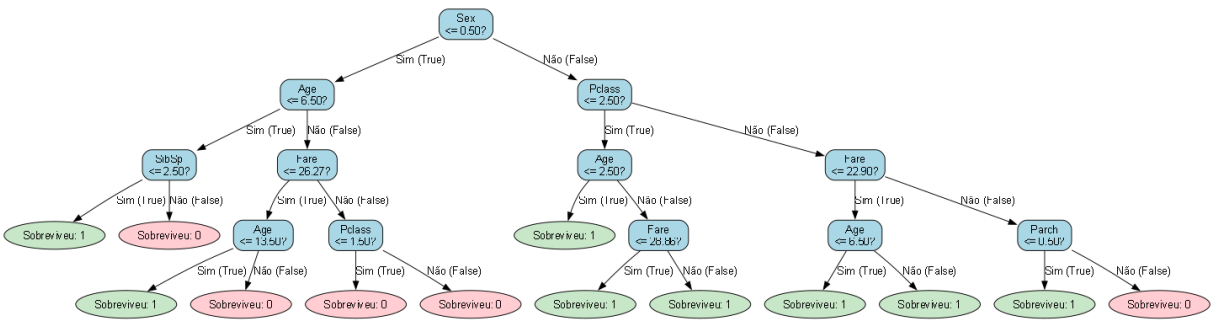

Figura 1: árvore gerada a partir do modelo CART

Figura 2: árvore gerada a partir do modelo ID3

Figura 3: árvore gerada a partir do modelo C4.5

6 CONCLUSÃO

O projeto foi concluído com sucesso, resultando em uma biblioteca funcional e bem estruturada para a criação de árvores de decisão. A implementação a partir do zero permitiu um entendimento profundo dos mecanismos internos de cada algoritmo, e a aplicação no dataset do Titanic demonstrou sua capacidade de extrair padrões complexos de dados realistas.

7 COMO USAR A BIBLIOTECA

1) Clone o repositório que está disponível no link: https://github.com/ale-faria/CC/tree/main/IA/Lista04/minha_arvore
2) Abra o terminal e entre na pasta “minha_arvore”: cd minha_arvore
3) Execute o comando “pip install .” no terminal
4) Agora a biblioteca já está pronta para uso, lembre-se de importar a biblioteca nos seus projetos, por exemplo: from minha_arvore import id3, c45, cart, imprimir_arvore