Seção 1 — Preparação dos dados

Para o desenvolvimento e avaliação dos algoritmos, dois conjuntos de dados foram utilizados, cada um com um propósito específico. O dataset Titanic serviu como a base de dados principal para o treinamento e aferição de métricas de desempenho em um cenário mais complexo e realista. Em paralelo, o dataset clássico Play Tennis foi empregado para a validação inicial dos algoritmos e para a geração dos artefatos visuais.

Dataset Principal: Titanic
O trabalho de preparação mais extenso foi realizado sobre o dataset do Titanic, cujos resultados (dados limpos e divididos) foram salvos em um arquivo .pkl para garantir a reprodutibilidade dos experimentos.

Limpeza de Valores Ausentes (Missing Values): A base continha valores ausentes em colunas críticas. A estratégia adotada foi preencher Age (Idade) com a mediana e Embarked (Porto de Embarque) com a moda (o valor mais frequente).

Partição dos Dados: O dataset foi dividido em 80% para treino e 20% para teste, utilizando uma partição estratificada. A estratificação com base na coluna alvo (Survived) é fundamental para garantir que a proporção de sobreviventes seja a mesma nos dois conjuntos, permitindo uma avaliação mais justa dos modelos.

Dataset de Validação: Play Tennis
Conforme sugerido na ementa da disciplina, o dataset "Play Tennis" foi utilizado como uma base de validação inicial. Sendo um dataset com apenas 14 linhas e atributos exclusivamente categóricos, ele serviu para confirmar a corretude das implementações dos algoritmos antes de aplicá-los ao dataset do Titanic.

Adicionalmente, devido ao seu tamanho reduzido, a árvore de decisão e as regras geradas a partir do Play Tennis são mais claras e legíveis. Por este motivo, serão elas as utilizadas nas seções posteriores deste relatório para a demonstração visual dos artefatos solicitados, como a árvore aprendida e as regras de decisão.

Preparação Específica por Algoritmo
Os algoritmos possuem requisitos diferentes quanto aos tipos de dados de entrada:

Para o ID3 (Discretização de Contínuos): O ID3 não processa atributos contínuos. Para a análise com o Titanic, foi necessário discretizar as features Age e Fare, convertendo-as em faixas categóricas (ex: "Criança", "Adulto") para que o algoritmo pudesse utilizá-las.

Para C4.5 e CART (Tratamento Nativo de Contínuos): Diferentemente do ID3, os algoritmos C4.5 e CART não exigem a pré-discretização, pois, conforme implementado, ambos encontram nativamente o "ponto de corte" (limiar) ótimo para uma variável numérica. Contudo, para

garantir uma comparação direta e consistente entre os três algoritmos, foi decidido utilizar a versão discretizada das features Age e Fare como entrada para todos os modelos nos resultados principais deste relatório. Adicionalmente, foram realizados testes com Age e Fare em seu formato numérico original para o C4.5 e o CART, nos quais não se observou uma diferença de desempenho significativa em relação ao uso dos dados já categorizados.

Seção 2

Nesta seção, é detalhada a implementação dos três algoritmos de árvore de decisão sendo eles ID3, C4.5 e CART desenvolvidos em Python. O código foi estruturado em classes distintas (`DecisionTreeID3`, `DecisionTreeC45`, `DecisionTreeCART`) e encapsulado em uma biblioteca Python customizada, chamada `trees_classifiers`, para facilitar a reutilização e a organização do projeto.

Detalhes da Biblioteca `trees_classifiers`

- **Repositório:** A biblioteca está disponível no GitHub, no seguinte endereço:
  - `https://github.com/Pedro-HFelix/IA.git`
- **Instalação:** Para instalar o pacote em um ambiente Python, utilize o comando:
  - `pip install git+https://github.com/Pedro-HFelix/IA.git`

Exemplo de Uso Básico:

```
from trees_classifiers.id3 import DecisionTreeID3
import pandas as pd
import pickle

# Carregar dados pré-processados
with open('Titanic.pkl', 'rb') as f: X_train, X_test, y_train, y_test = pickle.load(f)

# Preparar dados para o modelo
train_data = pd.concat([X_train, y_train], axis=1)
features = list(X_train.columns)
target = y_train.name

# Instanciar e treinar o modelo
model_id3 = DecisionTreeID3(data=train_data, features=features, target_class=target)
 model_id3.build()
```

2.1. Utilidades e Estratégias Comuns

Apesar de suas diferenças, os algoritmos compartilham certas funcionalidades e estratégias que foram implementadas como métodos auxiliares dentro de suas respectivas classes.

- Métricas de Divisão**:** As funções matemáticas que guiam a construção das árvores foram implementadas como métodos privados:
    1. Entropia de Shannon: Base para os algoritmos ID3 e C4.5.
    2. Ganho de Informação: Critério principal do ID3, calculado a partir da redução da entropia.
    3. Razão de Ganho: Critério principal do C4.5, que normaliza o Ganho de Informação pelo Split Info do atributo.
    4. Índice Gini: Critério principal do CART, que mede a impureza de um nó.
- Procura pela Melhor Divisão: A estratégia para encontrar o melhor atributo e o melhor ponto de corte varia conforme o tipo de atributo e o algoritmo:
    1. Atributos Categóricos:
        - Nos algoritmos ID3 e C4.5, a divisão é multi-ramificada (multi-way split), onde um ramo é criado para cada valor único do atributo.
        - No CART, a divisão é sempre binária, utilizando uma estratégia de binarização ótima, que testa cada valor contra todos os outros (`== valor` vs. `!= valor`) para encontrar a divisão que mais reduz a impureza Gini.
    2. Atributos Contínuos:
        - Nos algoritmos C4.5 e CART, o tratamento é nativo. O método implementado realiza uma varredura por limiar: ordena os valores únicos do atributo e testa os pontos médios entre valores adjacentes como potenciais pontos de corte, escolhendo aquele que maximiza a Razão de Ganho ou minimiza a Impureza Gini.
- Critério de Desempate: Para os casos em que múltiplos atributos apresentam o mesmo valor de critério (ex: mesmo Ganho de Informação), foi implementado um critério de desempate de dois níveis:
    1. Primeiro, é priorizado o atributo que gera o menor número de ramificações, favorecendo árvores mais simples, está decisão foi tomada por um motivo principal, desta forma a árvore consegue ficar em um tamanho menor além de também ser mais generalista desta forma.
    2. Se o empate persistir, o critério final é a ordem de entrada, selecionando o primeiro atributo encontrado na lista de features.

2.2. Implementação: ID3 (do zero)

A classe `DecisionTreeID3` implementa o algoritmo em sua forma clássica.

- Critério: Utiliza o Ganho de Informação como única métrica para a seleção de atributos.

- Atributos: Foi projetada para operar exclusivamente com atributos categóricos. Por essa razão, ao ser aplicada ao dataset do Titanic, utilizou-se a versão pré-processada onde as features contínuas `Age` e `Fare` foram discretizadas.

2.3. Implementação: C4.5 (do zero)

A classe `DecisionTreeC45` representa uma evolução do ID3, incorporando melhorias importantes.

- Critério: Utiliza a Razão de Ganho, que normaliza o Ganho de Informação, tornando o algoritmo menos suscetível a atributos com alta cardinalidade.
- Atributos: Lida nativamente tanto com atributos categóricos (multi-ramificados) quanto contínuos, selecionando o limiar ótimo para estes últimos.
- Valores Ausentes: A classe possui um método interno (`_handle_missing_values`) que trata valores ausentes antes da construção da árvore, utilizando a média para atributos numéricos e a moda para os categóricos.

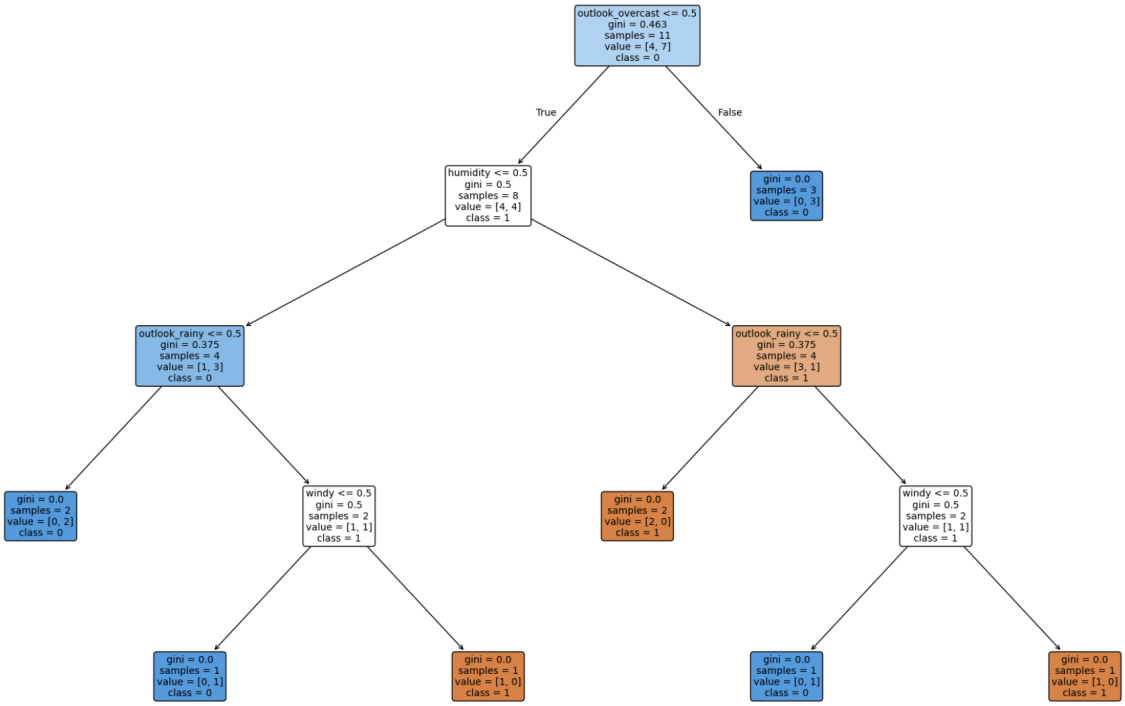
2.4. Implementação: CART (do zero)

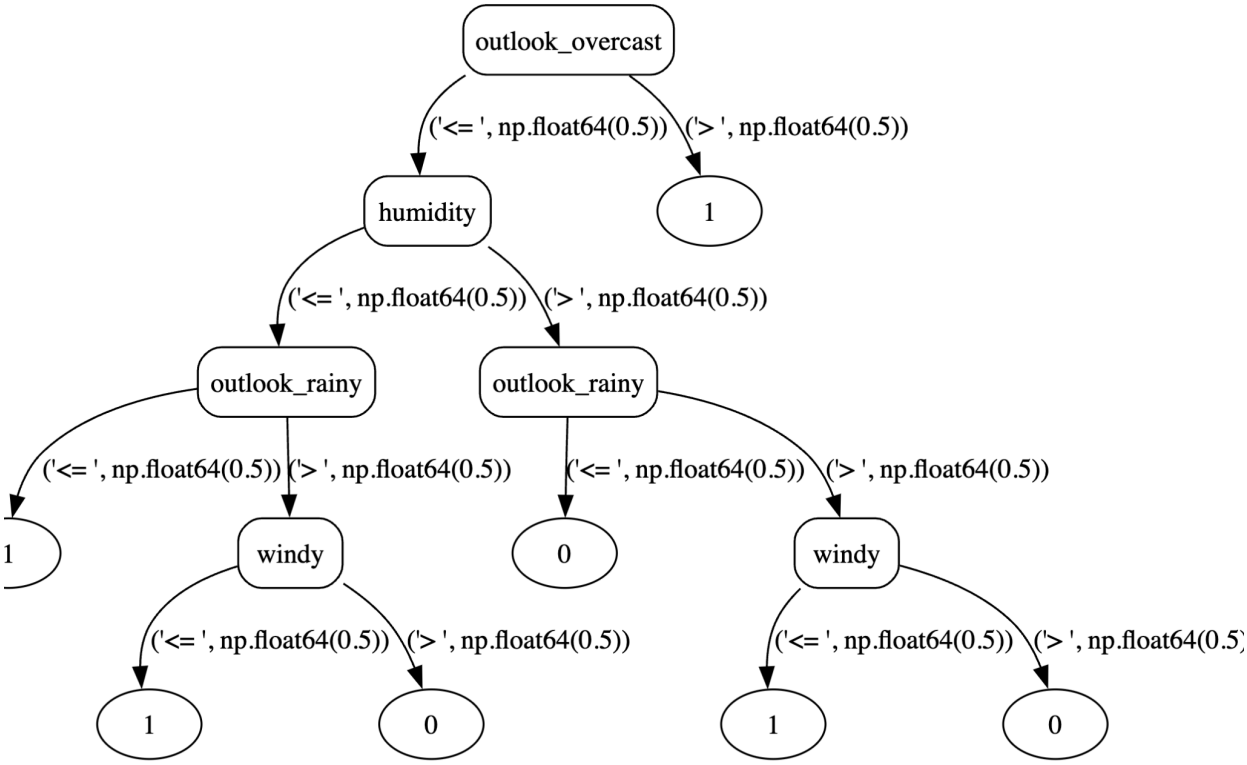
A classe `DecisionTreeCART` implementa o algoritmo que se diferencia significativamente dos baseados em entropia.

- Critério: Utiliza o Índice Gini para medir a impureza dos nós e selecionar os melhores pontos de corte.
- Divisões Binárias: Sua característica mais marcante é a criação de árvores estritamente binárias. Conforme implementado, todos os nós, independentemente do tipo de atributo, geram sempre dois ramos, o que leva a uma estrutura de árvore diferente do ID3 e C4.5.

Uma análise comparativa entre a árvore gerada pela implementação customizada do CART e a árvore da biblioteca Scikit-learn revela que a lógica de decisão de ambas é praticamente idêntica. As duas árvores escolhem outlook_overcast como o atributo mais importante para a primeira divisão e seguem com as mesmas escolhas de atributos e pontos de corte nos níveis seguintes, como humidity e windy. Essa consistência na estrutura, desde a raiz até as folhas com as previsões finais, é a principal validação de que a implementação do critério Gini e da busca por divisões binárias no algoritmo customizado está funcionando de forma correta e análoga à biblioteca padrão da indústria.

A principal diferença entre as duas imagens está na apresentação visual e no nível de detalhe. A árvore do Scikit-learn é mais informativa, exibindo em cada nó estatísticas como a impureza Gini, o número de amostras e a distribuição das classes, o que ajuda a entender a pureza de cada divisão. Em contrapartida, a visualização da árvore customizada foca de maneira mais limpa e direta na estrutura lógica das regras aprendidas, mostrando o atributo no nó e a condição da divisão nos ramos. Essa diferença, no entanto, é apenas na exibição, e a forte semelhança na lógica de decisão confirma o sucesso da implementação.


outlook_overcast <= 0.5
gini = 0.463
samples = 11
value = [4, 7]
class = 0
True
False
humidity <= 0.5
gini = 0.5
samples = 8
value = [4, 4]
class = 1
gini = 0.0
samples = 3
value = [0, 3]
class = 0
outlook_rainy <= 0.5
gini = 0.375
samples = 4
value = [1, 3]
class = 0
outlook_rainy <= 0.5
gini = 0.375
samples = 4
value = [3, 1]
class = 1
gini = 0.0
samples = 2
value = [0, 2]
class = 0
windy <= 0.5
gini = 0.5
samples = 2
value = [1, 1]
class = 1
gini = 0.0
samples = 2
value = [2, 0]
class = 1
windy <= 0.5
gini = 0.5
samples = 2
value = [1, 1]
class = 1
gini = 0.0
samples = 1
value = [0, 1]
class = 0
gini = 0.0
samples = 1
value = [1, 0]
class = 1
gini = 0.0
samples = 1
value = [0, 1]
class = 0
gini = 0.0
samples = 1
value = [1, 0]
class = 1



Seção 3

Relembrando que aqui está sendo usado os a base Play Tennis

ID3

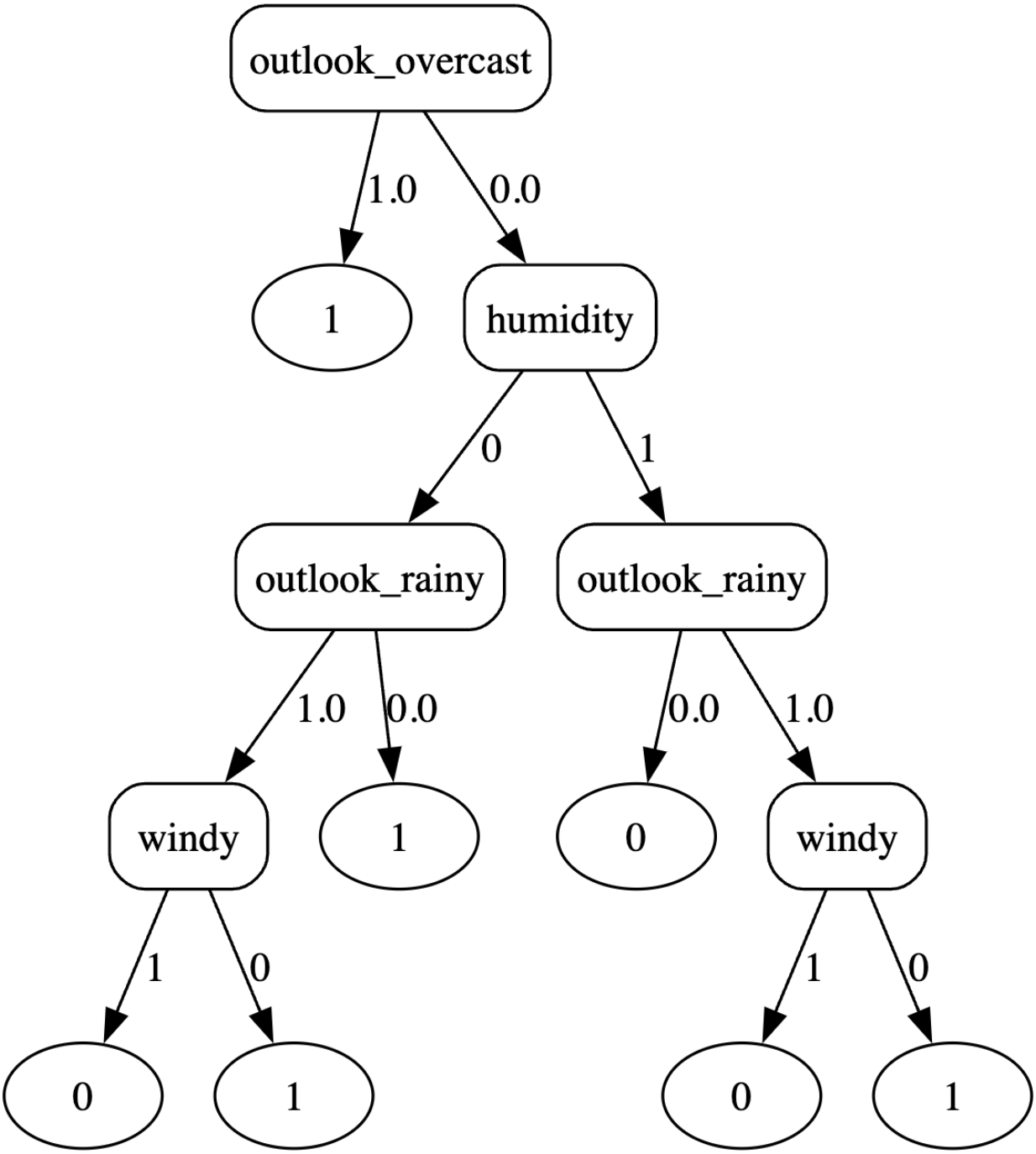
Métricas de Avaliação para o Modelo ID3
Acurácia: 1.0000

Relatório de Classificação:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	1
1	1.00	1.00	1.00	2
accuracy			1.00	3
macro avg	1.00	1.00	1.00	3

weighted avg 1.00 1.00 1.00 3

Matriz de Confusão:
[[1 0]
[0 2]]
Regra 1: SE outlook_overcast == 1.0 ENTÃO 1
Regra 2: SE outlook_overcast == 0.0 E humidity == 0 E outlook_rainy == 1.0 E windy == 1 ENTÃO 0
Regra 3: SE outlook_overcast == 0.0 E humidity == 0 E outlook_rainy == 1.0 E windy == 0 ENTÃO 1
Regra 4: SE outlook_overcast == 0.0 E humidity == 0 E outlook_rainy == 0.0 ENTÃO 1
Regra 5: SE outlook_overcast == 0.0 E humidity == 1 E outlook_rainy == 0.0 ENTÃO 0
Regra 6: SE outlook_overcast == 0.0 E humidity == 1 E outlook_rainy == 1.0 E windy == 1 ENTÃO 0
Regra 7: SE outlook_overcast == 0.0 E humidity == 1 E outlook_rainy == 1.0 E windy == 0 ENTÃO 1



C45

Métricas de Avaliação para o Modelo C45

Acurácia: 1.0000

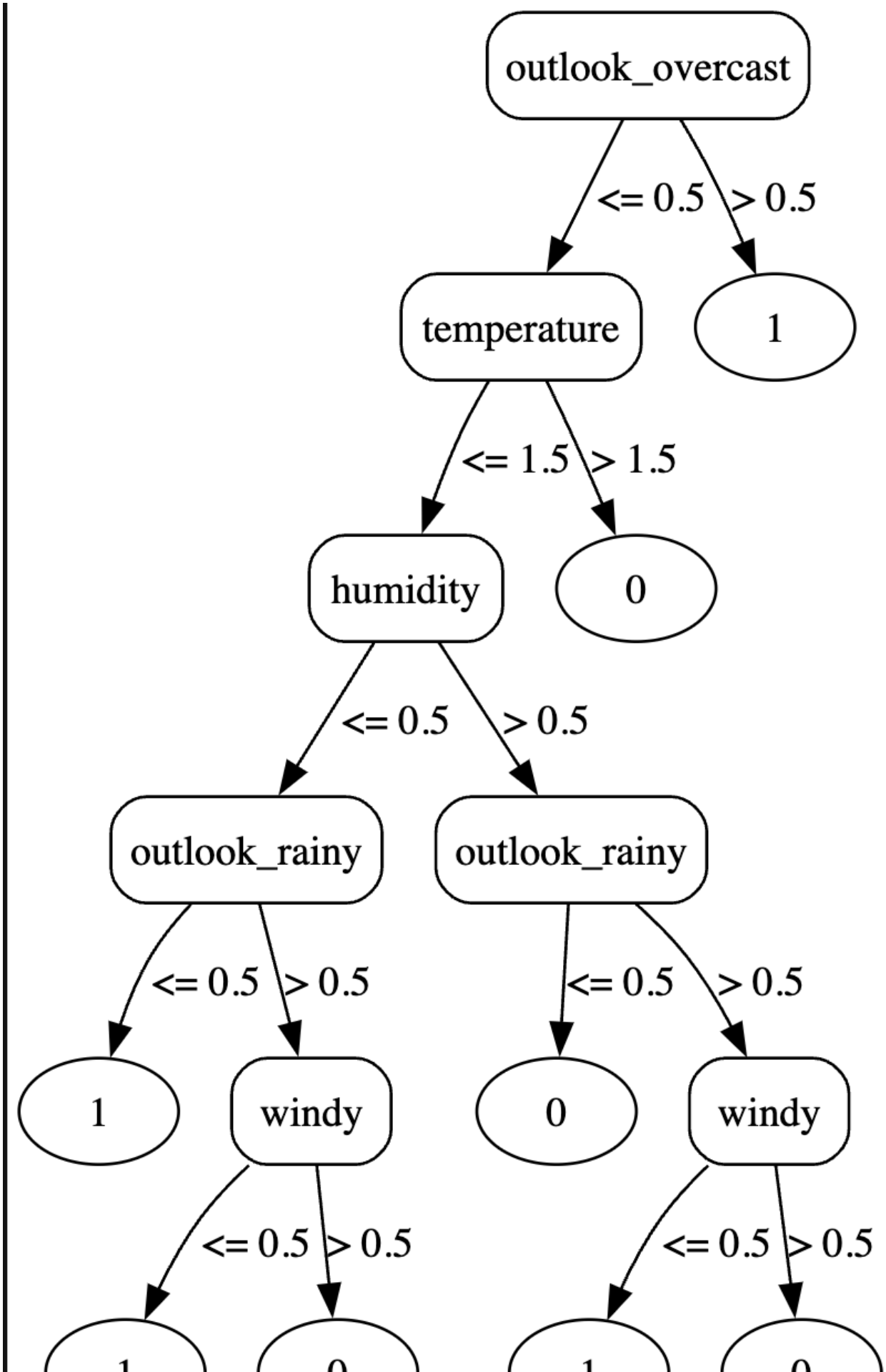
Relatório de Classificação:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	1
1	1.00	1.00	1.00	2
accuracy			1.00	3
macro avg	1.00	1.00	1.00	3
weighted avg	1.00	1.00	1.00	3

Matriz de Confusão:

```
[[1 0]
 [0 2]]
```

Regra 1: SE outlook_overcast <= 0.5 E temperature <= 1.5 E humidity <= 0.5 E outlook_rainy <= 0.5 ENTÃO 1
Regra 2: SE outlook_overcast <= 0.5 E temperature <= 1.5 E humidity <= 0.5 E outlook_rainy > 0.5 E windy <= 0.5 ENTÃO 1
Regra 3: SE outlook_overcast <= 0.5 E temperature <= 1.5 E humidity <= 0.5 E outlook_rainy > 0.5 E windy > 0.5 ENTÃO 0
Regra 4: SE outlook_overcast <= 0.5 E temperature <= 1.5 E humidity > 0.5 E outlook_rainy <= 0.5 ENTÃO 0
Regra 5: SE outlook_overcast <= 0.5 E temperature <= 1.5 E humidity > 0.5 E outlook_rainy > 0.5 E windy <= 0.5 ENTÃO 1
Regra 6: SE outlook_overcast <= 0.5 E temperature <= 1.5 E humidity > 0.5 E outlook_rainy > 0.5 E windy > 0.5 ENTÃO 0
Regra 7: SE outlook_overcast <= 0.5 E temperature > 1.5 ENTÃO 0
Regra 8: SE outlook_overcast > 0.5 ENTÃO 1


outlook_overcast
<= 0.5
> 0.5
temperature
1
<= 1.5
> 1.5
humidity
0
<= 0.5
> 0.5
outlook_rainy
outlook_rainy
<= 0.5
> 0.5
<= 0.5
> 0.5
1
windy
0
windy
<= 0.5
> 0.5
<= 0.5
> 0.5

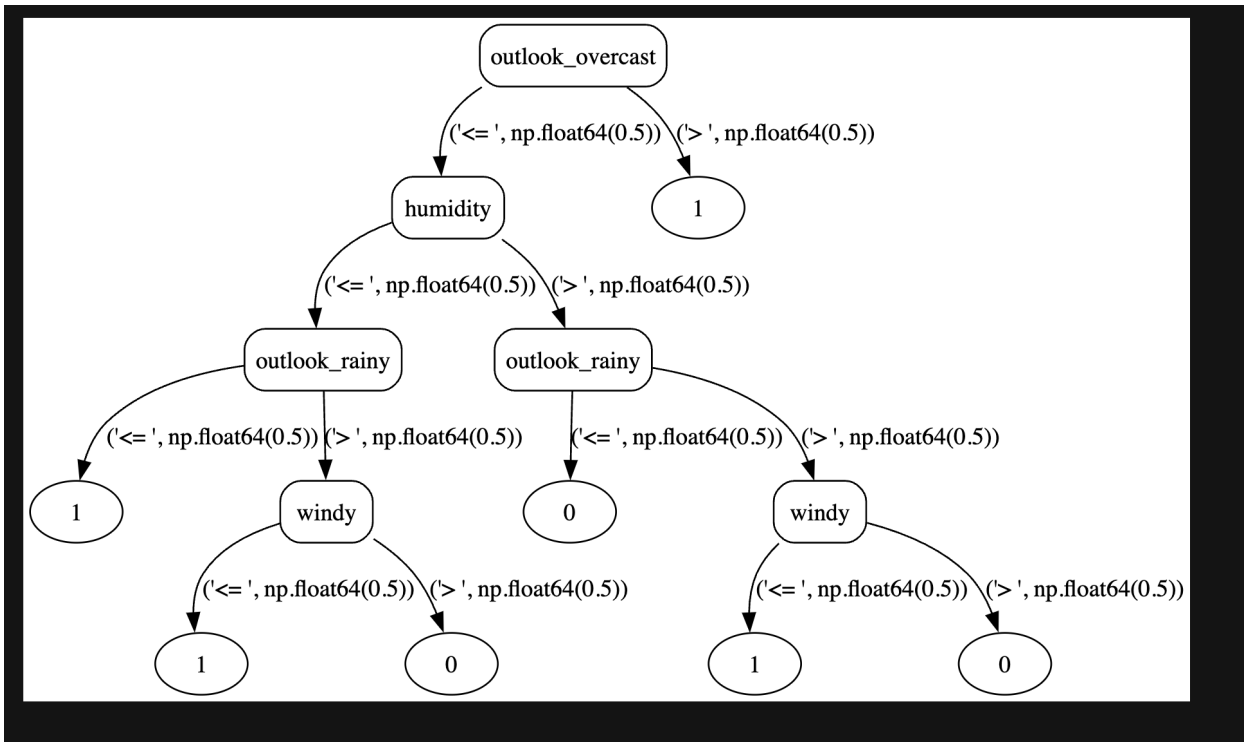
CART

Métricas de Avaliação para o Modelo Cart
Acurácia: 0.6667

Relatório de Classificação:

	precision	recall	f1-score	support
0	0.00	0.00	0.00	1
1	0.67	1.00	0.80	2
accuracy			0.67	3
macro avg	0.33	0.50	0.40	3
weighted avg	0.44	0.67	0.53	3

Matriz de Confusão:
[[0 1]
 [0 2]]
Regra 1: SE outlook_overcast <= 0.5 E humidity <= 0.5 E outlook_rainy <= 0.5 ENTÃO 1
Regra 2: SE outlook_overcast <= 0.5 E humidity <= 0.5 E outlook_rainy > 0.5 E windy <= 0.5 ENTÃO 1
Regra 3: SE outlook_overcast <= 0.5 E humidity <= 0.5 E outlook_rainy > 0.5 E windy > 0.5 ENTÃO 0
Regra 4: SE outlook_overcast <= 0.5 E humidity > 0.5 E outlook_rainy <= 0.5 ENTÃO 0
Regra 5: SE outlook_overcast <= 0.5 E humidity > 0.5 E outlook_rainy > 0.5 E windy <= 0.5 ENTÃO 1
Regra 6: SE outlook_overcast <= 0.5 E humidity > 0.5 E outlook_rainy > 0.5 E windy > 0.5 ENTÃO 0
Regra 7: SE outlook_overcast > 0.5 ENTÃO 1



Agora somente as métricas porém com a base do Titanic

Métricas de Avaliação para o Modelo ID3

Acurácia: 0.7654

Relatório de Classificação:

	precision	recall	f1-score	support
0	0.78	0.86	0.82	110
1	0.74	0.61	0.67	69
accuracy			0.77	179
macro avg	0.76	0.74	0.74	179

weighted avg 0.76 0.77 0.76 179

Matriz de Confusão:
[[95 15]
[27 42]]
Regra 1: SE Sex == 1 E Fare == 3 E SibSp == 0 E Age == 2 E Pclass == 3 ENTÃO 1
Regra 2: SE Sex == 1 E Fare == 3 E SibSp == 0 E Age == 2 E Pclass == 1 E Embarked_C == 0.0 ENTÃO 0
Regra 3: SE Sex == 1 E Fare == 3 E SibSp == 0 E Age == 2 E Pclass == 1 E Embarked_C == 1.0 E Parch == 0 ENTÃO 0
Regra 4: SE Sex == 1 E Fare == 3 E SibSp == 0 E Age == 2 E Pclass == 1 E Embarked_C == 1.0 E Parch == 1 ENTÃO 0
Regra 5: SE Sex == 1 E Fare == 3 E SibSp == 0 E Age == 2 E Pclass == 1 E Embarked_C == 1.0 E Parch == 2 ENTÃO 0
Regra 6: SE Sex == 1 E Fare == 3 E SibSp == 0 E Age == 2 E Pclass == 2 ENTÃO 0
...
Regra 133: SE Sex == 0 E Pclass == 1 E Parch == 2 E SibSp == 2 ENTÃO 1
Regra 134: SE Sex == 0 E Pclass == 1 E Parch == 2 E SibSp == 0 ENTÃO 1

Métricas de Avaliação para o Modelo C45

Acurácia: 0.7821

Relatório de Classificação:

	precision	recall	f1-score	support
0	0.80	0.86	0.83	110
1	0.75	0.65	0.70	69
accuracy			0.78	179
macro avg	0.77	0.76	0.76	179
weighted avg	0.78	0.78	0.78	179

Matriz de Confusão:
[[95 15]
[24 45]]
Regra 1: SE Sex <= 0.5 E SibSp <= 6.0 E Parch <= 5.5 E Pclass <= 2.5 E Fare == 2 E Parch <= 1.5 E Age == 2 E SibSp <= 1.5 E SibSp <= 0.5 E Embarked_C <= 0.5 ENTÃO 1

Regra 2: SE Sex <= 0.5 E SibSp <= 6.0 E Parch <= 5.5 E Pclass <= 2.5 E Fare == 2 E Parch <= 1.5 E Age == 2 E SibSp <= 1.5 E SibSp <= 0.5 E Embarked_C > 0.5 ENTÃO 1
Regra 3: SE Sex <= 0.5 E SibSp <= 6.0 E Parch <= 5.5 E Pclass <= 2.5 E Fare == 2 E Parch <= 1.5 E Age == 2 E SibSp <= 1.5 E SibSp > 0.5 E Embarked_C <= 0.5 E Parch <= 0.5 ENTÃO 1
Regra 4: SE Sex <= 0.5 E SibSp <= 6.0 E Parch <= 5.5 E Pclass <= 2.5 E Fare == 2 E Parch <= 1.5 E Age == 2 E SibSp <= 1.5 E SibSp > 0.5 E Embarked_C <= 0.5 E Parch > 0.5 ENTÃO 1
Regra 5: SE Sex <= 0.5 E SibSp <= 6.0 E Parch <= 5.5 E Pclass <= 2.5 E Fare == 2 E Parch <= 1.5 E Age == 2 E SibSp <= 1.5 E SibSp > 0.5 E Embarked_C > 0.5 ENTÃO 1
Regra 6: SE Sex <= 0.5 E SibSp <= 6.0 E Parch <= 5.5 E Pclass <= 2.5 E Fare == 2 E Parch <= 1.5 E Age == 2 E SibSp > 1.5 ENTÃO 1
Regra 7: SE Sex <= 0.5 E SibSp <= 6.0 E Parch <= 5.5 E Pclass <= 2.5 E Fare == 2 E Parch <= 1.5 E Age == 1 ENTÃO 1
...
Regra 110: SE Sex > 0.5 E Pclass > 1.5 E Age == 0 E SibSp > 2.5 E Parch > 1.5 E SibSp > 3.5 E SibSp <= 4.5 ENTÃO 0
Regra 111: SE Sex > 0.5 E Pclass > 1.5 E Age == 0 E SibSp > 2.5 E Parch > 1.5 E SibSp > 3.5 E SibSp > 4.5 ENTÃO 0

Métricas de Avaliação para o Modelo Cart

Acurácia: 0.6145

Relatório de Classificação:

	precision	recall	f1-score	support
0	0.61	1.00	0.76	110
1	0.00	0.00	0.00	69
accuracy			0.61	179
macro avg	0.31	0.50	0.38	179
weighted avg	0.38	0.61	0.47	179

Matriz de Confusão:
[[110 0]
[69 0]]
Regra 1: SE Sex <= 0.5 E Pclass <= 2.5 E Fare == 3 E Parch <= 1.5 ENTÃO 1
Regra 2: SE Sex <= 0.5 E Pclass <= 2.5 E Fare == 3 E Parch > 1.5 E Embarked_C <= 0.5 E SibSp <= 0.5 ENTÃO 1

Regra 3: SE Sex <= 0.5 E Pclass <= 2.5 E Fare == 3 E Parch > 1.5 E Embarked_C <= 0.5 E SibSp > 0.5 E Pclass <= 1.5 ENTÃO 0
Regra 4: SE Sex <= 0.5 E Pclass <= 2.5 E Fare == 3 E Parch > 1.5 E Embarked_C <= 0.5 E SibSp > 0.5 E Pclass > 1.5 ENTÃO 1
Regra 5: SE Sex <= 0.5 E Pclass <= 2.5 E Fare == 3 E Parch > 1.5 E Embarked_C > 0.5 ENTÃO 1
Regra 6: SE Sex <= 0.5 E Pclass <= 2.5 E Fare != 3 E Parch <= 1.5 E SibSp <= 0.5 E Embarked_C <= 0.5 E Fare == 1 E Embarked_Q <= 0.5 ENTÃO 1
...
Regra 105: SE Sex > 0.5 E Age != 0 E Pclass > 1.5 E Fare != 0 E Pclass > 2.5 E Embarked_C > 0.5 E Parch <= 0.5 E Age != 1 E Fare != 2 ENTÃO 0
Regra 106: SE Sex > 0.5 E Age != 0 E Pclass > 1.5 E Fare != 0 E Pclass > 2.5 E Embarked_C > 0.5 E Parch > 0.5 ENTÃO 1

Link do notebook: https://github.com/Pedro-HFelix/IA/blob/main/lista04/projeto.ipynb
Link do arquivo para limpeza: https://github.com/Pedro-HFelix/IA/blob/main/lista06/Lendo%20e%20Tratando%20Arquivo%20v2%20-%20AI%20-%20PUC%20Minas%202025.ipynb
Link dos códigos da biblioteca: https://github.com/Pedro-HFelix/IA/tree/main/trees_classifiers
Link do meu git com tudo que foi feito no trabalho: https://github.com/Pedro-HFelix/IA.git