CENTRO UNIVERSITÁRIO FEI
CIÊNCIAS DA COMPUTAÇÃO

ANGELO GABRIEL VASCONCELOS BAPTISTA RA: 22.122.081-7

THIAGO MONTEIRO TINONIN RA: 22.122.044-5

JUAN CAIO PARONITTI GALERA RA: 22.122.067-6

Aprendizado Indutivo - Árvore de Decisão

São Bernardo do Campo

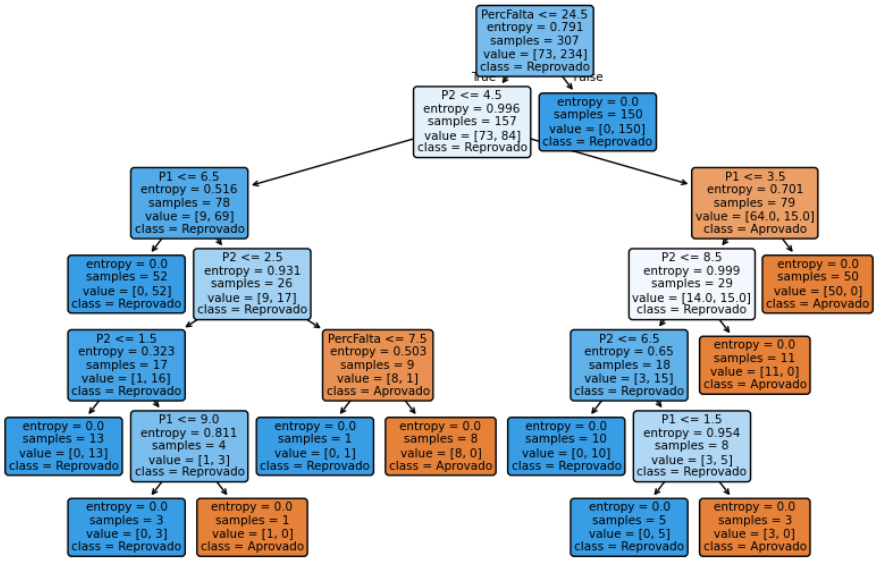
2025

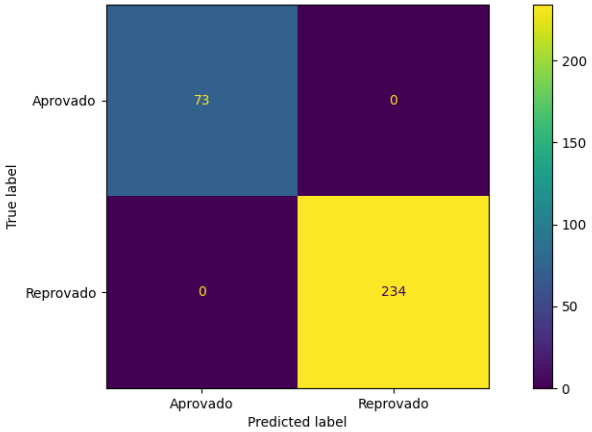
1. Critério de Provas

Para desenvolver a árvore de decisão e a matriz de confusão para o critério de notas foi utilizado o seguinte código:

```
import pandas as pd
import numpy as np
from sklearn import tree, metrics
from sklearn.tree import DecisionTreeClassifier
import matplotlib.pyplot as plt
from scipy.io import arff

data,meta = arff.loadarff('./CriterioProvas.arff')

attributes = meta.names()
data_value = np.asarray(data)

p1 = np.asarray(data['P1']).reshape(-1,1)
p2 = np.asarray(data['P2']).reshape(-1,1)
percFalta = np.asarray(data['PercFalta']).reshape(-1,1)
features = np.concatenate((p1, p2, percFalta),axis=1)
target = data['resultado']


Arvore = DecisionTreeClassifier(criterion='entropy').fit(features, target)

plt.figure(figsize=(10, 6.5))
tree.plot_tree(Arvore,feature_names=['P1', 'P2','PercFalta'],class_names=['Aprovado', 'Reprovado'],
               filled=True, rounded=True)
plt.show()

fig, ax = plt.subplots(figsize=(25, 10))
metrics.ConfusionMatrixDisplay.from_estimator(Arvore,features,target,display_labels=['Aprovado', 'Reprovado'], values_format='d', ax=ax)
plt.show()
```

Que geraram os seguintes resultados:





Após uma análise minuciosa dos dados foi confirmado que ambos estão corretos.

2. Partida de Tênis

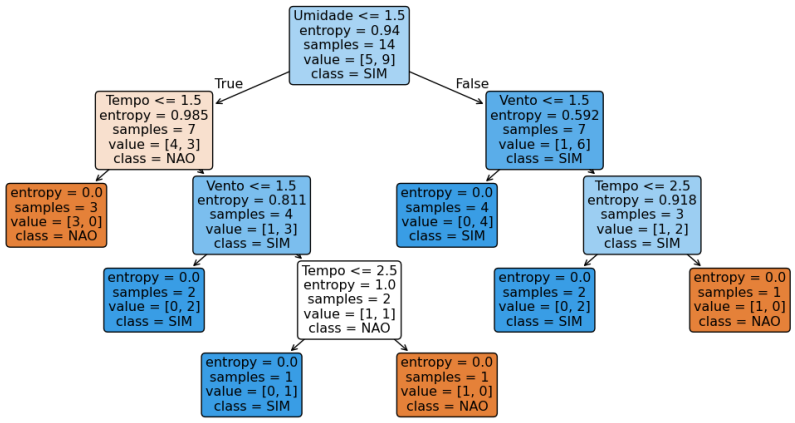
Como base de dados para a segunda questão, foi decidido acatar a sugestão dada em aula e utilizar a base de dados da partida de Tênis apresentada na aula de teoria de aprendizado indutivo. Antes de desenvolver a árvore, foi necessário adaptar os dados para o formato numérico, haja vista que a maioria dos dados estavam em forma de texto, o qual não pode ser processado pelo algoritmo.

```
% tempo = {1 = Sol, 2 = Nublado, 3 = Chuva}
% Temperatura {1 = Quente, 2 = Morno, 3 = Frio}
% Umidade {1 = Alta, 2 = Normal}
% Vento {1=Fraco, 2=Forte}
% Partida {1 = SIM, 2 = 2}

@relation partida

%@attribute Dia integer
@attribute Tempo integer
@attribute Temperatura integer
@attribute Umidade integer
@attribute Vento integer
@attribute Partida {SIM, NAO}

@data
1,1,1,1,NAO
1,1,1,2,NAO
2,1,1,1,SIM
3,2,1,1,SIM
3,3,2,1,SIM
3,3,2,2,NAO
2,3,2,2,SIM
1,2,1,1,NAO
1,3,2,1,SIM
3,2,2,1,SIM
1,2,2,2,SIM
2,2,1,2,SIM
2,1,2,1,SIM
3,2,1,2,NAO
```

O código necessário para desenvolver a DTC e a CM:

```
import pandas as pd
import numpy as np
from sklearn import tree, metrics
from sklearn.tree import DecisionTreeClassifier
from sklearn.preprocessing import LabelEncoder
import matplotlib.pyplot as plt
from scipy.io import arff


data,meta = arff.loadarff('./tenis.arff')

attributes = meta.names()
data_value = np.asarray(data)

Tempo = np.asarray(data['Tempo']).reshape(-1, 1)
Temperatura = np.asarray(data['Temperatura']).reshape(-1, 1)
Umidade = np.asarray(data['Umidade']).reshape(-1, 1)
Vento = np.asarray(data['Vento']).reshape(-1, 1)

features = np.concatenate((Tempo, Temperatura, Umidade, Vento),axis=1)
target = data['Partida']

Arvore = DecisionTreeClassifier(criterion='entropy').fit(features, target)

plt.figure(figsize=(10, 6.5))
tree.plot_tree(Arvore,feature_names=['Tempo', 'Temperatura', 'Umidade', 'Vento'],class_names=['NAO', 'SIM'],
               filled=True, rounded=True)
plt.show()

fig, ax = plt.subplots(figsize=(25, 10))
metrics.ConfusionMatrixDisplay.from_estimator(Arvore,features,target,display_labels=['NAO', 'SIM'], values_format='d', ax=ax)
plt.show()
```

A árvore produzida por esse código:



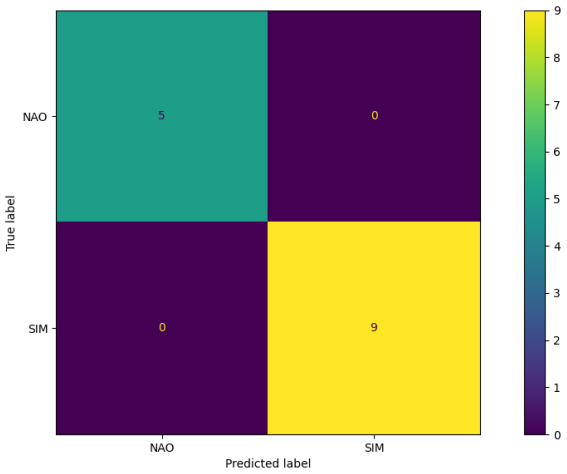
Para confirmar que a árvore realmente funciona, foram separados 3 testes.

1 – Caso o dia tenha umidade = 1 (Alta) e tempo = 1 (Sol) no dia obrigatoriamente não tem jogo. Isso é confirmado pois há 3 dias em que essas condições são atendidas e Partida = NÃO.

2 – Caso a umidade = 2 (Normal) e vento = 1 (Fraco) obrigatoriamente há jogo, pois há 4 amostras em que essas condições são atendidas e partida = SIM.

3 - Caso a umidade = 2 (Normal), vento = 2 (Forte) e tempo = 1 ou 2 (Sol ou Nublado) também há jogo, havendo duas amostras que comprovam isso.

A matriz de confusão gerada pela árvore:



A Matriz está totalmente correta.

3. Conclusão

Por tanto, é evidente que o algoritmo possui uma boa capacidade para detectar padrões numéricos nos datasets e fazer a classificação dos dados baseado nesses padrões.