**Solução**

1. Para este exercício, usamos as bibliotecas Pandas, Matplotlib e Scikit-Learn. Importe as bibliotecas.

from google.colab import files

import pandas as pd

import matplotlib.pyplot as plt

from sklearn import tree

from sklearn.tree import DecisionTreeClassifier, plot\_tree

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import accuracy\_score

import warnings

warnings.simplefilter('ignore')

2. Pegando os dados.

url = '<https://raw.githubusercontent.com/higoramario/univesp-com410-aprendizado-de-maquinas/main/carros-avaliacao.csv>'

carros = pd.read\_csv(url)

carros.head()

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **preco** | **manutencao** | **portas** | **pessoas** | **bagageiro** | **seguranca** | **aceitabilidade** |
| **0** | muitoalto | muitoalto | 2 | 2 | pequeno | baixa | inaceitavel |
| **1** | muitoalto | muitoalto | 2 | 2 | pequeno | media | inaceitavel |
| **2** | muitoalto | muitoalto | 2 | 2 | pequeno | alta | inaceitavel |
| **3** | muitoalto | muitoalto | 2 | 2 | medio | baixa | inaceitavel |
| **4** | muitoalto | muitoalto | 2 | 2 | medio | media | inaceitavel |

3. Transformando os atributos em números para uso na Árvore de Decisão.

carros['preco'] = carros['preco'].map({'muitoalto':3,'alto':2,'medio':1,'baixo':0})

carros['manutencao'] = carros['manutencao'].map({'muitoalto':3,'alto':2,'medio':1,'baixo':0})

carros['portas'] = carros['portas'].map({'2':2,'3':3,'4':4,'5mais':5})

carros['pessoas'] = carros['pessoas'].map({'2':2,'4':4,'5mais':5})

carros['bagageiro'] = carros['bagageiro'].map({'grande':2,'medio':1,'pequeno':0})

carros['seguranca'] = carros['seguranca'].map({'alta':2,'media':1,'baixa':0})

carros.head(10)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **preco** | **manutencao** | **portas** | **pessoas** | **bagageiro** | **seguranca** | **aceitabilidade** |
| **0** | 3 | 3 | 2 | 2 | 0 | 0 | inaceitavel |
| **1** | 3 | 3 | 2 | 2 | 0 | 1 | inaceitavel |
| **2** | 3 | 3 | 2 | 2 | 0 | 2 | inaceitavel |
| **3** | 3 | 3 | 2 | 2 | 1 | 0 | inaceitavel |
| **4** | 3 | 3 | 2 | 2 | 1 | 1 | inaceitavel |
| **5** | 3 | 3 | 2 | 2 | 1 | 2 | inaceitavel |
| **6** | 3 | 3 | 2 | 2 | 2 | 0 | inaceitavel |
| **7** | 3 | 3 | 2 | 2 | 2 | 1 | inaceitavel |
| **8** | 3 | 3 | 2 | 2 | 2 | 2 | inaceitavel |
| **9** | 3 | 3 | 2 | 4 | 0 | 0 | inaceitavel |

4. Separando atributos e classes para o treinamento e teste.

atributos\_nomes = ['preco','manutencao','portas','pessoas','bagageiro','seguranca']

atributos = carros[atributos\_nomes]

classes = carros['aceitabilidade']

5. Separando o conjunto de dados para treinamento e teste.

atributos\_treino, atributos\_teste, classes\_treino, classes\_teste = train\_test\_split(atributos, classes, test\_size=0.1, random\_state=10)

6. Criando o modelo.

arvore = DecisionTreeClassifier()

arvore = arvore.fit(atributos\_treino,classes\_treino)

7. Plotando a árvore de decisão em tamanho grande para conseguir visualizar em detalhes. No caderno Python, clique na figura para alternar entre visão ampliada e reduzida.

plt.figure(figsize=(300,160))

plot\_tree(arvore, filled=True, rounded=True, feature\_names=atributos\_nomes)

plt.show()

8. Testando o caso perfeito em que todos os valores são os melhores possíveis.

print(arvore.predict([[0,0,5,5,2,2]]))

['muitobom']

9. Basta que a segurança seja baixa para que seja classificada como inaceitável.

print(arvore.predict([[0,0,5,5,2,0]]))

['inaceitavel']

10. Verificando a acurácia de classificação.

classes\_predicao = arvore.predict(atributos\_teste)

acuracia = accuracy\_score(classes\_teste,classes\_predicao)

print('Acurácia de classificação: {}'.format(round(acuracia,3)\*100)+'%')

Acurácia de classificação: 98.3%\*

\* O valor de acurácia pode variar um pouco dependendo do modelo gerado.