Centro Universitário FEI

Aprendizado Indutivo - CC7711

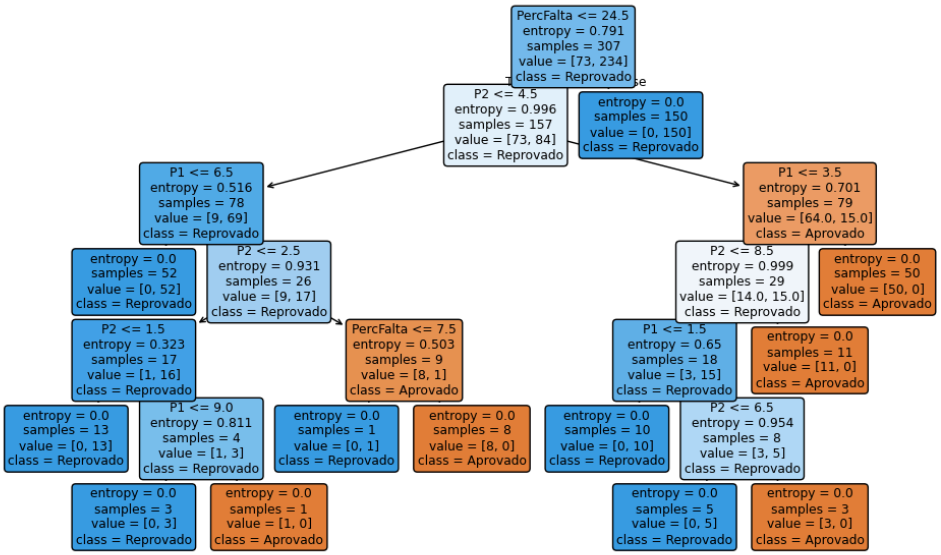
Vinicius henrique Silva 22.122.063-5
Luan Petroucic Moreno 22.122.076-7
Cauê Jacomini Zanatti 22.122.024-7

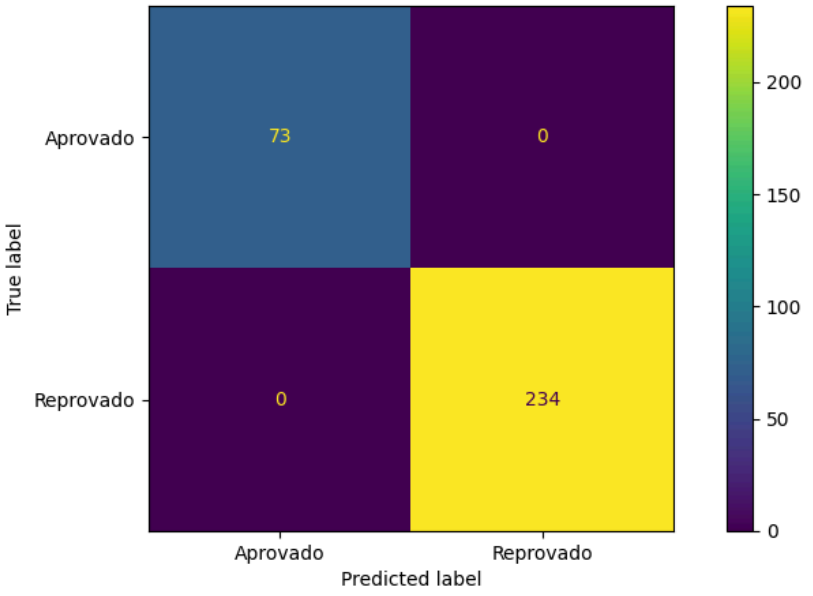
1. Arff - Critério de Provas

Para analisarmos o arquivo CriterioProvas.arff desenvolvemos um código em python com as bibliotecas sklearn para análise dos dados, matplotlib para plotagem da árvore de decisão e scipy.io para leitura do arquivo.

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


def load_arff_data(file_path):
    """Carrega e processa os dados do ARFF, convertendo valores categóricos em numéricos."""
    data, meta = arff.loadarff(file_path)

    features = []
    encoders = {}

    for attr in meta.names()[:-1]:
        col = data[attr]


        if meta[attr][0] == 'nominal':
            col = np.array([x.decode('utf-8').strip() for x in col])
            le = LabelEncoder()
            col = le.fit_transform(col)
            encoders[attr] = le

        features.append(col)

    features = np.column_stack(features)


    target = np.array([x.decode('utf-8').strip() for x in data[meta.names()[-1]]])
    target_encoder = LabelEncoder()
    target = target_encoder.fit_transform(target)

    return features, target, meta.names()[:-1], target_encoder, encoders


def train_decision_tree(features, target):

    model = DecisionTreeClassifier(criterion='entropy', random_state=42)
    model.fit(features, target)
    return model


def plot_decision_tree(model, feature_names, class_names):

    plt.figure(figsize=(12, 7))
    plot_tree(model, feature_names=feature_names, class_names=class_names, filled=True, rounded=True)
    plt.show()


def plot_confusion_matrix(model, features, target, class_names):

    fig, ax = plt.subplots(figsize=(10, 5))
    metrics.ConfusionMatrixDisplay.from_estimator(
        model, features, target, display_labels=class_names, values_format='d', ax=ax
    )
    plt.show()


if __name__ == "__main__":
    file_path = "./CriterioProvas.arff"


    features, target, feature_names, target_encoder, encoders = load_arff_data(file_path)


    model = train_decision_tree(features, target)


    plot_decision_tree(model, feature_names, target_encoder.classes_)


    plot_confusion_matrix(model, features, target, target_encoder.classes_)
```

Gerando então as seguintes saídas no programa:

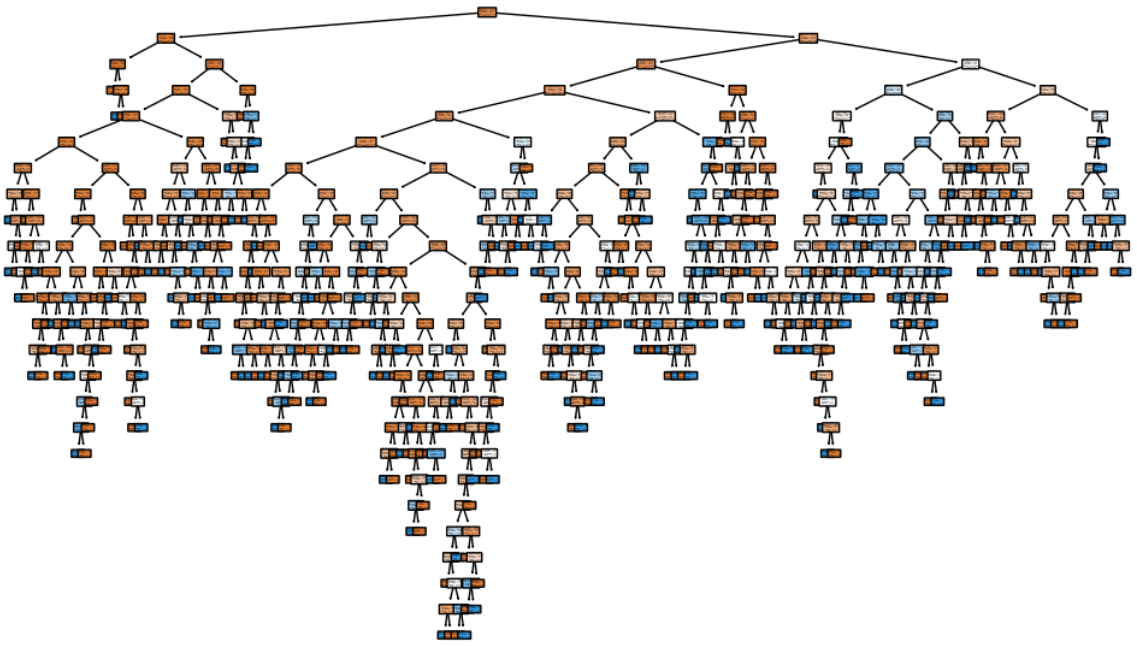


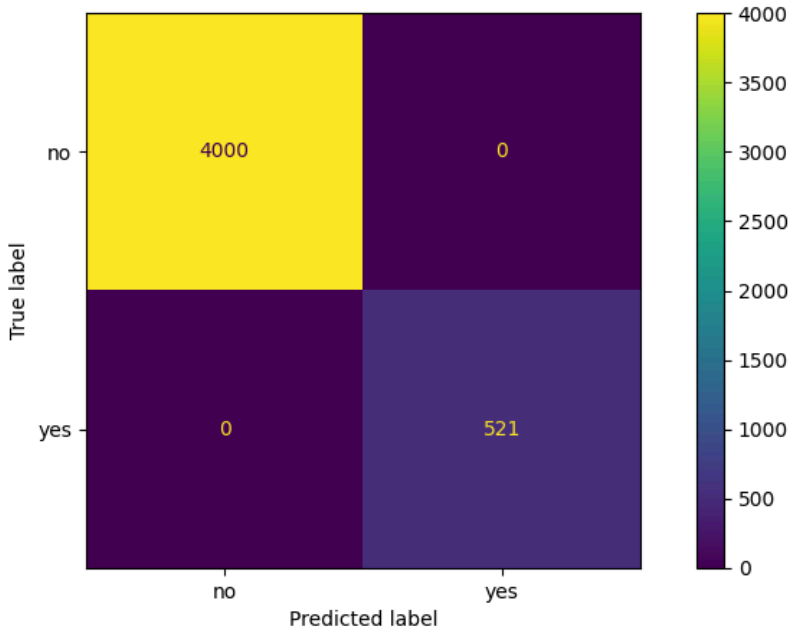


Através dos resultados podemos verificar que o modelo obteve uma grande taxa de acurácia, porém talvez seja um indicativo que a fonte de dados possui um conjunto de dados fácil de se classificar.

2. Bank.arff

Para a análise do arquivo bank.arff, utilizamos o mesmo código desenvolvido anteriormente em python, apenas alterando o filename no código, recebemos então a seguinte saída:





Para utilizarmos o código anterior foi preciso realizar algumas alterações no arquivo arff fonte, como por exemplo espaçamentos indevidos e valores em strings que atrapalham o funcionamento do algoritmo. Através da matriz de confusão conseguimos verificar que a acurácia do código está novamente alta, porém como nossa árvore possui diversos nós e arestas significa que este arquivo fonte já possui dados mais complexos diferente do anterior.