1. Introdução

No presente trabalho pretende-se classificar o dataset Íris, a partir das características de cada flor Íris, a flor se divide em três tipos possíveis, também chamados de ***labels***:

- ➔ Iris-Versicolor.
- ➔ Iris-Setosa.
- ➔ Iris-Virginica.

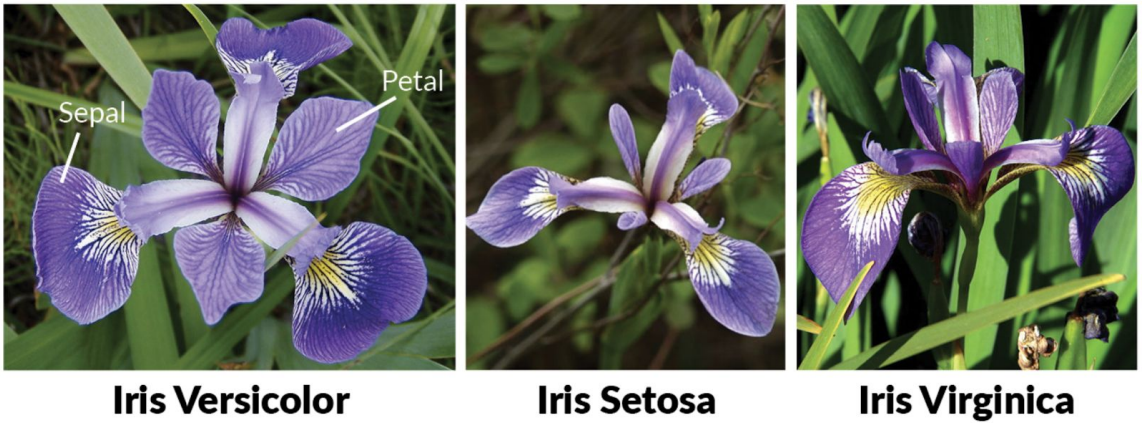

Figura 1 - Representa os três tipos de flor Iris: Iris-Versicolor, Iris-Setosa, Iris-Virginica.

O dataset foi originalmente criado por Ronald Fisher e é composto por 150 amostras. Ao carregar o dataset é possível observar que ele se divide em quatro características(***feature***):

- ➔ sepal length (Comprimento da Sépala em cm)
- ➔ sepal width (Largura da Sépala em cm)
- ➔ petal length (Comprimento da Pétala em cm)
- ➔ petal width (Largura da Pétala em cm)

1.1. Objetivo Geral

O objetivo é criar um modelo de aprendizado de máquina utilizando Árvore de Decisão que possa aprender a partir das medidas disponíveis no dataset íris cujas espécies já são conhecidas, para que seja possível predizer a categoria associada a cada espécie da flor Íris.

1.2. Objetivos Específicos

Como resultados de saída são pedidos:

1) Usar a linguagem python e suas bibliotecas, especialmente, a biblioteca scikit-learn de suporte a machine learning;
2) Ajustar o dataset (pré-processamento);
3) Treinar modelo baseado em Árvore de Decisão;
4) Executar as predições (classificação de espécies de Iris) usando o modelo

treinado;
5) Avaliar a acurácia do modelo;

Na Seção 2, é apresentada a Metodologia utilizada para o desenvolvimento do trabalho . Na Seção 3, são apresentados os resultados alcançados, onde são expressos os gráficos associados à problemática proposta. Na Seção 4 serão descritas as conclusões alcançadas no decorrer da implementação do trabalho.

2. Metodologia

Como temos as medidas corretas para cada espécie de íris, esse é um problema de aprendizado supervisionado. Onde queremos prever uma das várias opções (a espécie de íris), tornando-a um exemplo de problema de classificação. As saídas possíveis (diferentes espécies de íris) são chamadas de classes. Cada íris no conjunto de dados pertence a uma das três classes consideradas no modelo, portanto, este problema é um problema de classificação de três classes. A saída desejada para um único ponto de dados (uma íris) é a espécie da flor, considerando suas características. Para um determinado ponto de dados, a classe / espécie a que pertence, é chamada de rótulo.

Como já foi dito, usaremos o conjunto de dados Iris.data.

Os itens individuais são chamados de amostras no aprendizado de máquina, enquanto suas propriedades são chamadas de recursos. A forma da matriz de dados é o número de amostras multiplicado pelo número de recursos. Neste caso: nossos dados possuem 150 amostras com 4 características cada (comprimento da sépala (cm), largura da sépala (cm), comprimento da pétala (cm), largura da pétala (cm)).

Para construir o modelo foi utilizado o classificador de árvore de decisão simples disponível no Scikit-learn, o Scikit-learn é uma biblioteca para o desenvolvimento de modelos de aprendizado de máquina, seja para problemas de regressão ou classificação, que é amplamente utilizada e apreciada. As árvores de decisão classificam instâncias partindo da raiz da árvore para algum nodo folha que fornece a classe da instância. Cada nodo da árvore específica o teste de algum atributo da instância, e cada arco alternativo que desce daquele nodo corresponde a um dos possíveis valores deste atributo. Uma instância é classificada começando no nodo raiz da árvore, testa o atributo relacionado a este nodo e segue o arco que corresponde ao valor do atributo na instância em questão. Este processo é repetido então para a sub-árvore abaixo até chegar a um nodo folha, um exemplo desse processo será melhor explorado na Seção de Resultados .

Para testar o desempenho do modelo, mostramos a ele novos dados para os quais já temos previamente os rótulos (dados que já estão classificados). Isso geralmente é feito dividindo os dados rotulados que coletamos (neste exemplo, nossas 150 medições de flores) em duas partes. Uma parte dos dados é usada para construir o modelo de aprendizado de máquina e é chamada de dados de treinamento ou conjunto de treinamento. O restante dos dados será usado para testar o funcionamento do modelo, isto é chamado de conjunto de teste, dados de teste ou conjunto de validação.

Os dados foram separados em :

- 80% dos dados foram separados em um conjunto de treinamento
- 20% dos dados foram separados em um conjunto de testes.

O objetivo dessa divisão é garantir que o modelo não seja treinado em todos os dados disponíveis, para que possamos testar seu desempenho em dados não vistos previamente pelo modelo treinado. Se usarmos todos os dados como dados de treinamento, podemos acabar superdimensionando o modelo, o que significa que ele pode ter um desempenho insatisfatório em dados que o modelo ainda não tenha visto.

No classificador de árvore de decisão do scikit-learn a variável de destino pode ser categórica ou numérica. Por ser mais clara na visualização dos resultados dado o conjunto de dados da íris, no presente trabalho a natureza categórica das flores será mantida, no entanto, em alguns casos como na **Figura 3** para que fosse possível plotar o gráfico, houve a necessidade de converter os dados de categóricos para dados numéricos.

3. Resultados

No decorrer do projeto utilizou-se um algoritmo supervisionado usado no aprendizado de máquina chamado de árvore de decisão. O principal objetivo ao usar a árvore de decisão é encontrar o atributo que gera a melhor divisão dos dados e subconjunto com maior pureza. Existem algumas métricas para a definição de pureza, ou seja, qual será a métrica utilizada para decidir qual é o melhor atributo que divide os dados gerando a partição mais pura. Essas métricas são o Índice Gini, Chi-Square, Information Gain e a redução da variância.

No trabalho usou-se o scikit-learn para construir uma árvore de decisão, nos treinamentos os hiperparâmetros utilizados foram os padrões para o classificador *DecisionTreeClassifier* . São eles:

- criterion: medida de qualidade da divisão, aqui que definimos qual dessas métricas (Índice Gini, Chi-Square, Information Gain ou a redução da variância) iremos usar.
- splitter: estratégia utilizada para dividir o nó de decisão.
- max_depth: profundidade máxima da árvore.
- min_samples_split: número de amostras mínimas para considerar um nó para divisão.
- min_samples_leaf: número de amostras mínimas no nível folha.

3.1. Visualização dos dados

Nos gráficos abaixo, o agrupamento diagonal dos pares de atributos sugere uma alta correlação e uma relação previsível. A relação entre pares de características de um Iris-Setosa(em rosa) é distintamente diferente daquelas das outras duas espécies. Há uma sobreposição nas relações entre pares das outras duas espécies, Iris-Versicolor (marrom) e Iris-Virginia (verde).

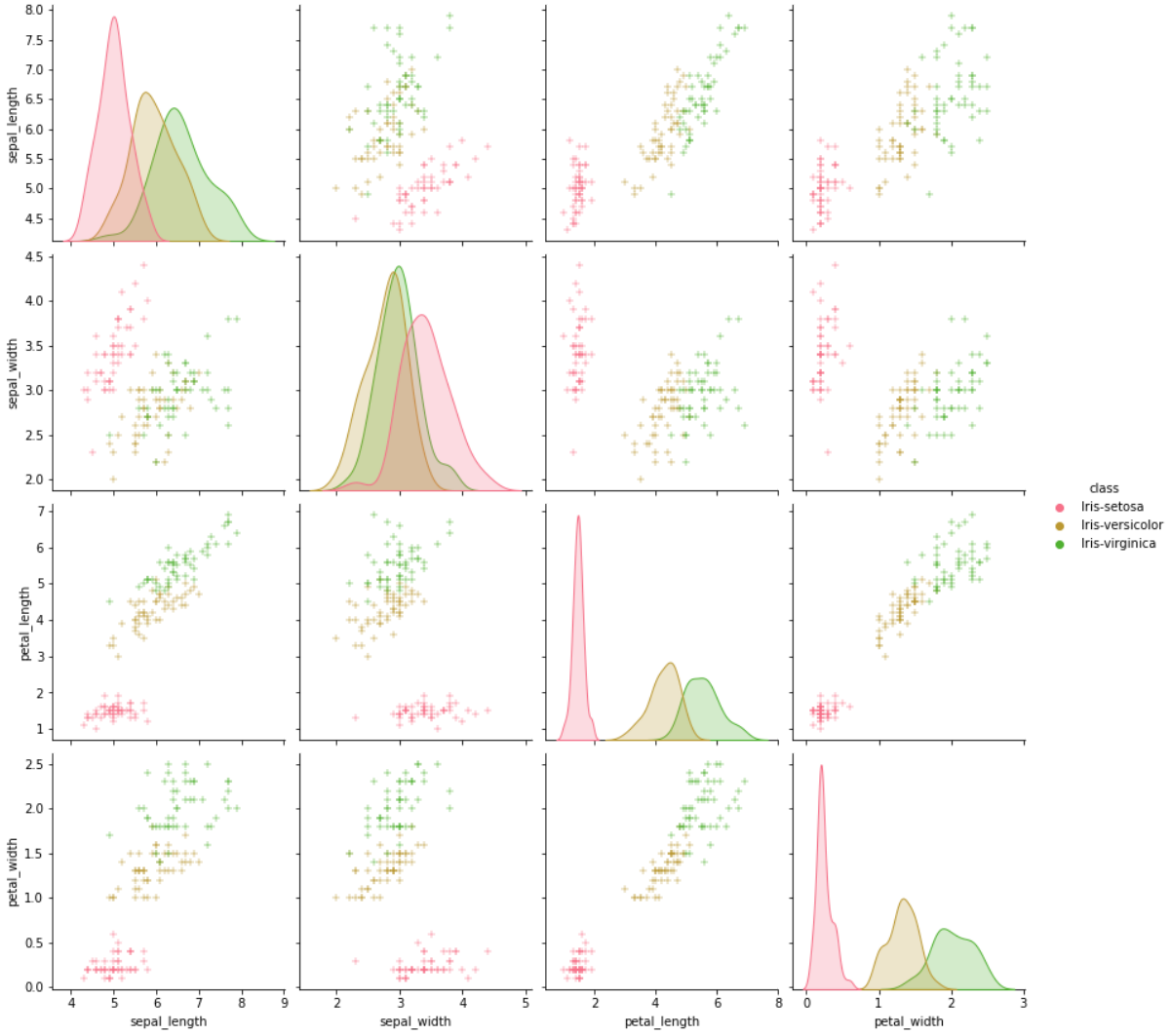

Figura 2: Gráfico com a representação da distribuição de pares de características da base de dados. Em rosa são visíveis as características relacionadas a íris-setosa, em marrom é representado as características da íris-versicolor e em verde as características da iris-virginica.

3.2. Divisão do conjunto de dados em um conjunto de treinamento e um conjunto de teste

Os dados de treinamento e de teste foram divididos de forma aleatória a partir da utilização de alguns recursos disponíveis no Scikit-learn. Ao dividir o conjunto de dados pseudo-aleatoriamente em dois conjuntos separados, podemos treinar usando um conjunto e testar usando outro. Isso garante que não usaremos as mesmas observações em ambos os conjuntos. Além de ser mais flexível e rápido do que criar um modelo usando todo o conjunto de dados para treinamento.

No entanto, a precisão dos modelos depende das observações no conjunto de testes. À medida que a complexidade de um modelo aumenta, a precisão do treinamento (precisão obtida ao treinar e testar o modelo com os mesmos dados) aumenta.

Se um modelo for muito complexo ou não o suficiente, a precisão do teste será menor.

Nos experimentos realizados 80% dos dados foram separados em um conjunto de treinamento e 20% dos dados foram separados em um conjunto de testes.

A melhor forma de inspecionarmos os dados e visualizá-los. Na figura abaixo, podemos ver a distribuição dos dados selecionados para treinamento do modelo. Isso nos permitirá verificar se os dados de medição das pétalas e sépalas estão bem distribuídos.

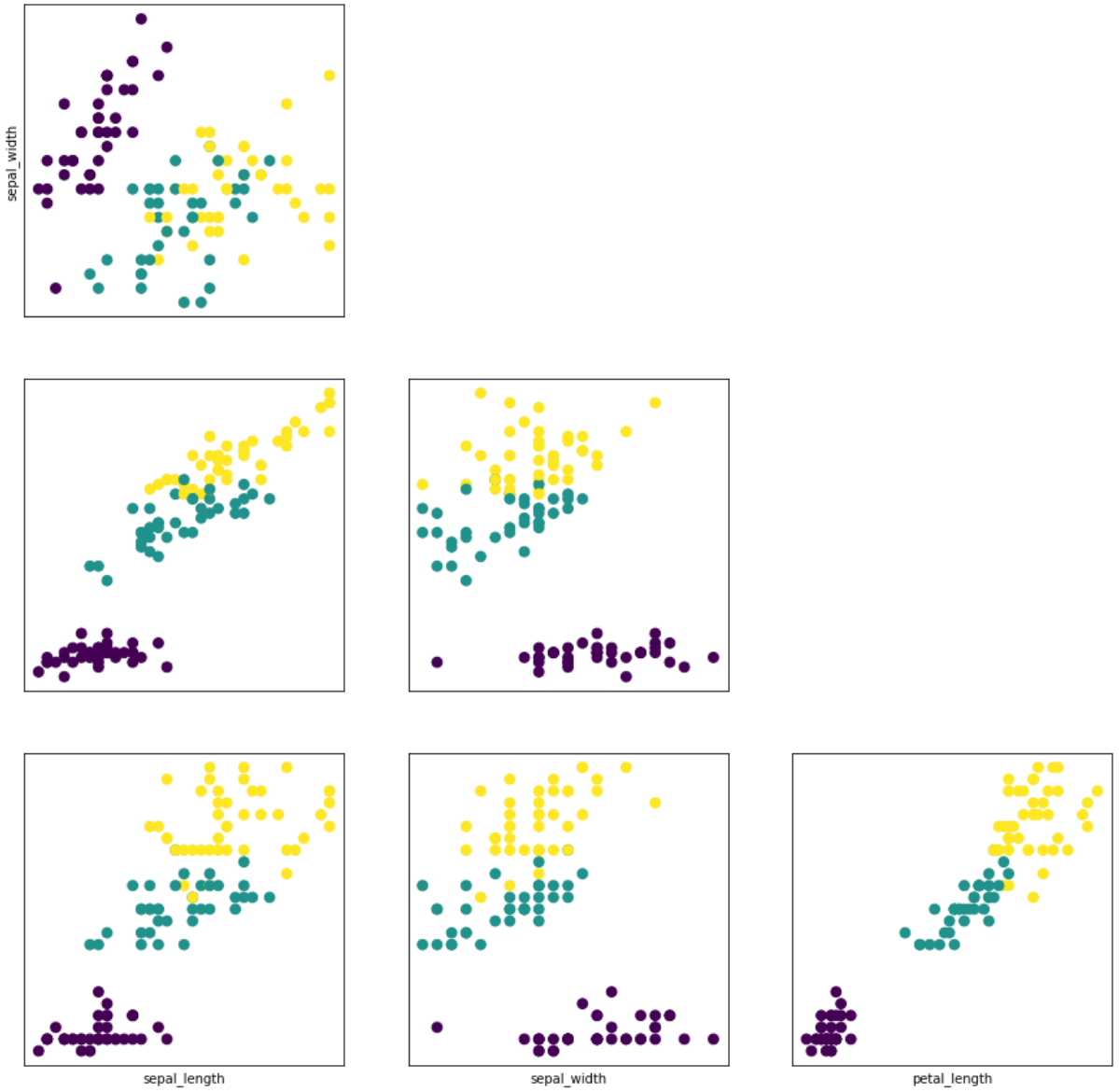

Figura 3: Gráfico para verificar a distribuição dos dados utilizados para treinamento e teste.

Como podemos ver no gráfico, os dados parecem estar muito bem distribuídos. Isso permitirá uma precisão maior do modelo, uma vez que as informações são bem variadas para cada tipo de Íris. Imagine, por exemplo, se as flores da espécie Setosa tivessem as mesmas medidas da Versicolor. Isso faria com que nosso modelo tivesse dificuldades em prever novas flores inseridas mais tarde. É possível visualizar no gráfico que as regiões mais densas dos dados são mais grossas e mais esparsas.

3.3. Diagrama da Árvore gerada

O diagrama abaixo mostra claramente como o algoritmo se comportou. Aqui, a primeira divisão é baseada no comprimento da pétala, sendo menor que 2,45 cm sendo identificado como Íris-virginica, enquanto aqueles com maior sendo classificados como Íris-Setosa. No entanto, uma divisão posterior ocorre para aqueles com comprimento de pétala maior do que 2,45, com duas divisões adicionais para terminar com classificações finais mais precisas.

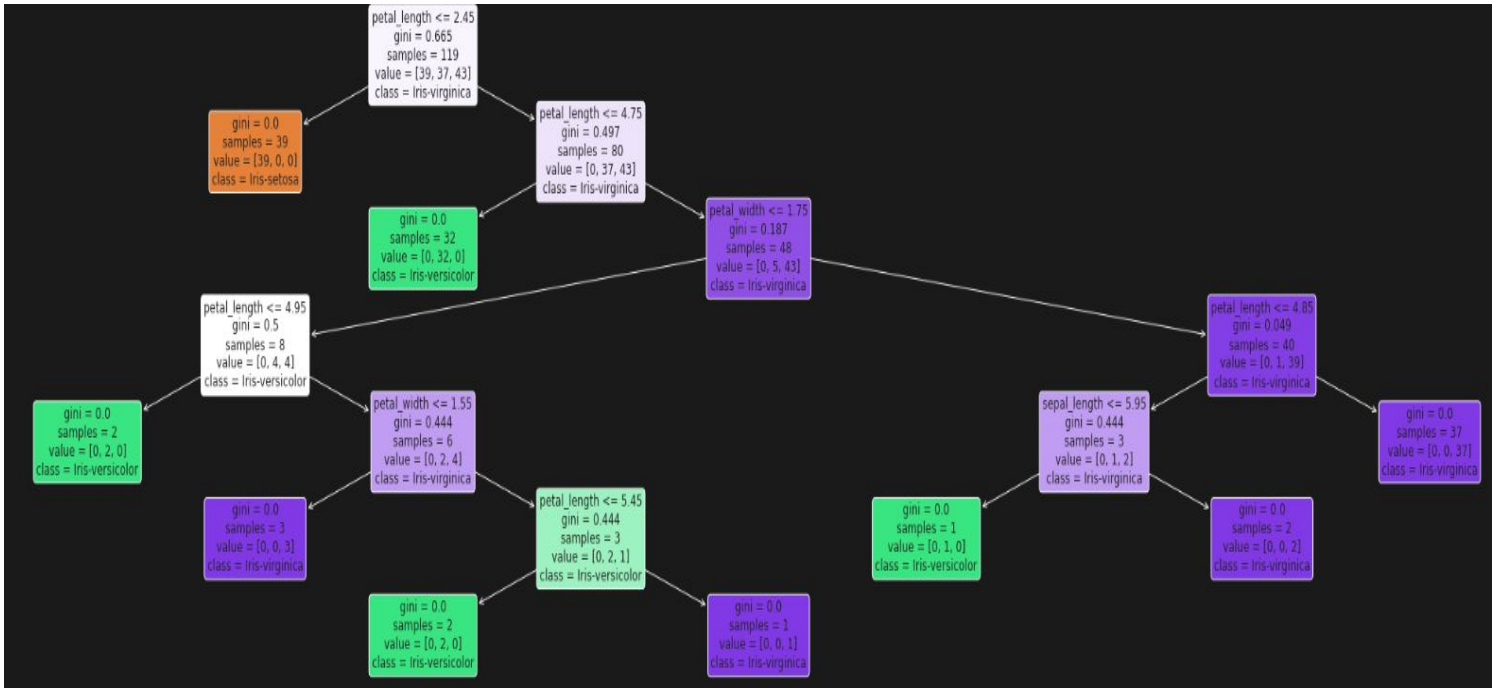

Figura 4 : Plot da árvore gerada no processo de treinamento do modelo

Não estamos apenas interessados em como ele foi executado nos dados de treinamento, mas em quão bem ele executou nos dados de teste. Isso significa que temos que usar o modelo treinado para prever a classe a partir dos valores de teste, o que é feito usando o método predict ().

3.4. Matriz de Confusão

Estamos interessados em saber como isso funciona em termos de verdadeiros positivos (verdadeiros previstos e são verdadeiros), falsos positivos (verdadeiros previstos, mas não são verdadeiros), falsos negativos (falsos previstos, mas na verdade são verdadeiros) e negativos verdadeiros (falsos previstos e são realmente falsos) .

Uma maneira de fazer isso é examinar os resultados em uma matriz de confusão. Uma matriz de confusão nos permite visualizar como os rótulos preditos e que são realmente verdadeiros combinam, mostrando os valores preditos em um eixo e os valores reais no outro. Isso é útil para identificar onde podemos obter falsos positivos ou falsos negativos e, portanto, como o algoritmo foi executado.

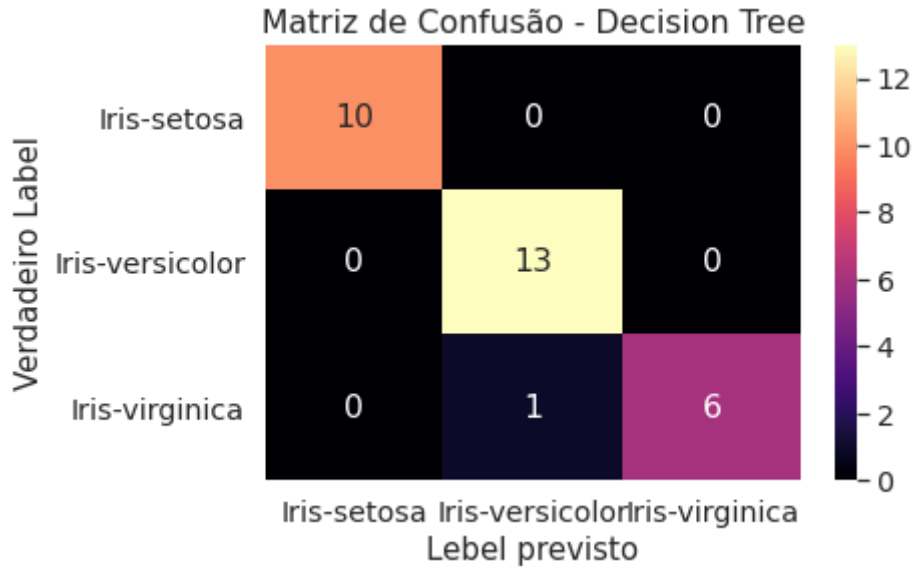

Figura 5: Representação da Matriz de confusão para o conjunto de testes.

Como pode ser visto na **Figura 5**, a partir dos dados vistos, apenas um valor falhou em ser previsto da classe Iris-verginica, o que sugere que, de modo geral, esse algoritmo se saiu bem na previsão de dados não vistos pelo modelo.

3.5. Medindo o desempenho do modelo

Para medir o desempenho, foram usadas as métricas a seguir, tendo como base os valores da matriz de confusão da **Figura 5**:
Ao medir o desempenho do modelo é considerado a quantidade total de ocorrência de Verdadeiros Positivos (VP), Falsos Positivo(FP), Falsos Negativo (FN), Falsos Positivos (FP).

- ***Acurácia***

A pontuação de precisão é a fração de verdadeiros positivos e verdadeiros negativos sobre o número total de rótulos atribuídos, que é calculada como:

$$Acurácia = \frac{VP + VN}{VP + VN + FP + FN}$$

- ***Precisão***

A precisão informa quantos dos valores previstos pelo modelo para uma determinada classe estão realmente nessa classe. É calculado como:

$$Precisão = \frac{VP}{VP+FP}$$

- ***Recall***

O recall informa quantos dos valores em cada classe receberam o rótulo correto, informando assim como ele se saiu em relação aos falsos negativos. É calculado como:

$$Recall = \frac{VP}{VP+FN}$$

- ***f1-Score***

O f1-Score é uma média ponderada da escala de precisão e recall, com 1 sendo o melhor e 0 o pior. Este utiliza a média harmônica, de forma que o valor fique mais próximo do menor número, e evita superestimar o desempenho do modelo nos casos em que um parâmetro é alto e o outro baixo. É calculado como:

$$f1-Score = \frac{2*(Precisão*Recall)}{Precisão*Recall}$$

A seguir o resultado de todas essas métricas acima:

	precision	recall	f1-score	support
Iris-setosa	1.00	1.00	1.00	10
Iris-versicolor	0.93	1.00	0.96	13
Iris-virginica	1.00	0.86	0.92	7
accuracy			0.97	30
macro avg	0.98	0.95	0.96	30
weighted avg	0.97	0.97	0.97	30
Acuracia	0.9666666666666667			

O modelo obteve uma precisão de aproximadamente 96%, considerada uma boa precisão considerando a base de dados e modelo utilizado.

4. Conclusão

As vantagens de usar um classificador baseado em árvore de decisão é que eles são fáceis de acompanhar e interpretar, eles podem lidar com dados numéricos e categóricos, eles limitam a influência de preditores ruins e podemos extrair sua estrutura para visualizar.

É claro que também há desvantagens, tendo em vista que eles podem criar árvores tendenciosas se uma classe for dominante, podemos obter árvores excessivamente complexas que levam a ajustes excessivos e pequenas variações nos dados podem criar resultados amplamente diferentes. No entanto, eles podem ser muito úteis na prática e podem ser usados junto com outros algoritmos de classificação, como k-vizinhos mais próximos ou floresta aleatória para ajudar a tomar decisões e entender como as classificações surgiram.