Relatório do Modelo de Árvore de Decisão

O dataset não veio com suas colunas nomeadas, então elas foram adicionadas conforme a Figura 1.

	sepal_length	sepal_with	petal_length	petal_width	name
0	5.1	3.5	1.4	0.2	Iris-setosa

Figura 1

Essa renomeação de colunas foi feita seguindo as descrições disponibilizadas no seguinte documento:

```
7. Attribute Information:
   1. sepal length in cm
   2. sepal width in cm
   3. petal length in cm
   4. petal width in cm
   5. class:
      -- Iris Setosa
      -- Iris Versicolour
      -- Iris Virginica
```

Figura 2

Tal documento está disponível para download no mesmo link do dataset disponibilizado para o projeto.

Em um primeiro momento foi analisado o dataset para verificar a necessidade de qualquer processamento mais sofisticado. Porém. Ele parece bem balanceado [Figura 3], então técnicas de balanceamento não foram necessárias.

```
iris_df[['name']].value_counts()
```

name	data
Iris-setosa	50
Iris-versicolor	50
Iris-virginica	50

Figura 3

Devido a isso, a única adaptação feita no dataset foi a mudança dos nomes de espécies das plantas de seu nome original para um identificador único como mostrado abaixo.

```
plant_names = iris_df.name.unique()
plant_names
```

	0
0	Iris-setosa
1	Iris-versicolor
2	Iris-virginica

3 rows × 1 columns Open in new tab

```
iris_df.name = iris_df.name.apply(lambda x: list(plant_names).index(x))
iris_df
```

	sepal_length	sepal_with	petal_length	petal_width	name
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0

Figura 4

Finalizada essa parte inicial, foi feita a divisão dos dados em treino e teste. Obedecendo os 30% dos dados disponíveis para validação.

```
x = iris_df[['sepal_length', 'sepal_with', 'petal_length', 'petal_width']].values
y = iris_df[['name']].values


x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=1)
```

Figura 5

Por fim, foi feito o treino do modelo.

```
model = DecisionTreeClassifier(criterion='entropy', max_depth=5, min_samples_leaf=3, random_state=1)
model.fit(x_train, y_train)
```

DecisionTreeClassifier

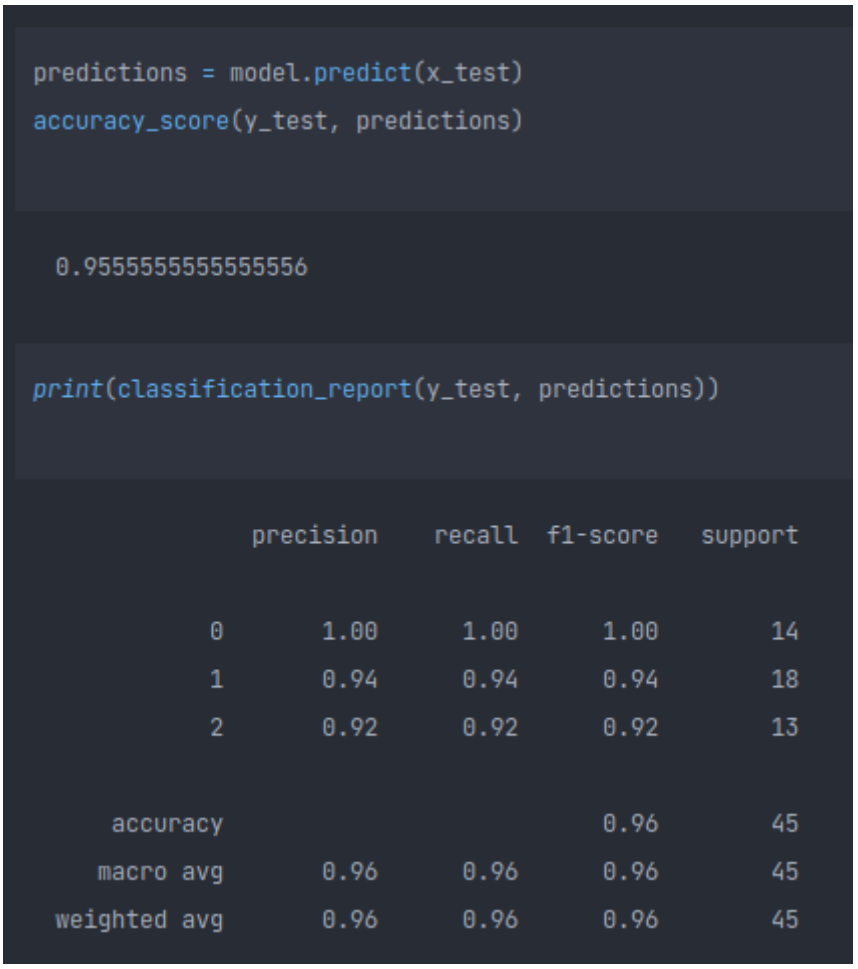
```
DecisionTreeClassifier(criterion='entropy', max_depth=5, min_samples_leaf=3,
                       random_state=1)
```

Figura 6

Como pode ser observado, foi utilizada **entropy** como métrica para a Decision Tree. Isso se deve exclusivamente ao fato de, pelas pesquisas realizadas, a métrica **gini** ser utilizada para datasets muito grandes, onde o tempo de treino pode ser impactado positivamente por ela.

Resultados

Feito o treino, foi avaliada a acurácia de previsão do modelo e também foi gerado um relatório de classificação de descriminação entre as espécies de plantas e um detalhamento maior sobre a acurácia.



```
predictions = model.predict(x_test)
accuracy_score(y_test, predictions)
```

```
0.9555555555555556
```

```
print(classification_report(y_test, predictions))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        14
           1       0.94      0.94      0.94        18
           2       0.92      0.92      0.92        13

    accuracy                           0.96        45
   macro avg       0.96      0.96      0.96        45
weighted avg       0.96      0.96      0.96        45
```

Figura 7

Analisando esses resultados, verificamos que para a classe Iris-setosa (0) foi predita com mais exatidão que as demais. Sendo que a classe Iris-virginica (2) foi a com menor precisão.

Apesar disso, analisando o modelo no geral, foi atingida uma acurácia de aproximadamente 96%.