Classificação em Python

Preparando o ambiente

Instalando o Jupyter

1. Abra o cmd (prompt do windows)
2. Vá até o diretório de trabalho (onde vai salvar os arquivos) (use o comando cd)
3. Verifique a instalação do python → python –version
4. Instale o Jupyter → pip install jupyter

Instalando os pacotes necessários

1. pip install pandas
2. pip install scikit-learn

Executando o Jupyter

1. Execute o Jupyter → jupyter server
2. Copie o link impresso na saída do Jupyter para acesso incluindo o token e abra no navegador.

   Exemplo:

   http://127.0.0.1:8888/lab?token=5b6522371edf87fcffbebf15728704cd4edfb00b1e0a82c9

Criando o modelo

Importe os pacotes

```
import pandas as pd

from sklearn import tree

from sklearn.model_selection import train_test_split
```

Lendo o dataset a partir do arquivo csv

```
iris = pd.read_csv('./path-to-file/iris.csv')
```

Checando o dataset

```
[7]: iris.head()
```

[7]:

	sepallength	sepalwidth	petallength	petalwidth	class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

Separando o dataset em data e target

- É preciso separa a coluna que contém a classe (target)

```
iris_data = iris.drop(columns='class', axis=1).values

iris_target = iris['class'].values
```

Separando em treino e teste

```
data_train, data_test, target_train, target_test = train_test_split(iris_data, iris_target, test_size=0.34)
```

Escolhendo o algoritmo de classificação

```
classifier=tree.DecisionTreeClassifier()
```

Treinando o classificador

- O classificador é treinado com os dados de treino e as respectivas classes (target)

```
classifier.fit(data_train, target_train)
```

Realiza a classificação

```
predictions = classifier.predict(data_test)
```

Checando a saída do modelo (classificador)

```
print(predictions)
```

```
['Iris-versicolor' 'Iris-virginica' 'Iris-versicolor' 'Iris-versicolor'
 'Iris-virginica' 'Iris-virginica' 'Iris-setosa' 'Iris-setosa'
 'Iris-setosa' 'Iris-virginica' 'Iris-setosa' 'Iris-setosa'
 'Iris-virginica' 'Iris-setosa' 'Iris-setosa' 'Iris-virginica'
 'Iris-setosa' 'Iris-versicolor' 'Iris-setosa' 'Iris-virginica'
 'Iris-versicolor' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica'
 'Iris-setosa' 'Iris-virginica' 'Iris-setosa' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-setosa' 'Iris-versicolor' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa'
 'Iris-setosa' 'Iris-virginica' 'Iris-setosa' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa' 'Iris-virginica'
 'Iris-virginica' 'Iris-setosa' 'Iris-versicolor' 'Iris-virginica'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica']
```

Verificando a acurácia

- A acurácia é verificada comparando as classes verdadeiras do conjunto de teste (target_test) com as predições (predictions)

```
from sklearn.metrics import accuracy_score

print(accuracy_score(target_test,predictions))
```

Analisando a matriz de confusão

```
from sklearn.metrics import confusion_matrix
confusion_matrix(target_test,predictions)
```

```
array([[21,  0,  0],
       [ 0, 13,  4],
       [ 0,  0, 14]])
```

Checando precision e recall por classe

```
from sklearn.metrics import classification_report
print(classification_report(target_test, predictions))
```

	precision	recall	f1-score	support
Iris-setosa	1.00	1.00	1.00	21
Iris-versicolor	1.00	0.76	0.87	17
Iris-virginica	0.78	1.00	0.88	14
accuracy			0.92	52
macro avg	0.93	0.92	0.91	52
weighted avg	0.94	0.92	0.92	52