from google.colab import drive drive.mount('/content/drive')

Mounted at /content/drive

from google.colab import files uploaded = files.upload()

> Choose Files No file chosen enable. Saving iris.csv to iris.csv

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to

import pandas as pd

iris = pd.read_csv('/content/iris.csv')

from matplotlib import pyplot as plt import numpy as np iris

		pecarin	petal.length	sepal.width	sepal.length		\rightarrow
0 5.1 3.5 1.4 0.2 Setosa	0		1.4	3.5	5.1	0	
1 4.9 3.0 1.4 0.2 Setosa	0		1.4	3.0	4.9	1	
2 4.7 3.2 1.3 0.2 Setosa	0		1.3	3.2	4.7	2	
3 4.6 3.1 1.5 0.2 Setosa	0		1.5	3.1	4.6	3	
4 5.0 3.6 1.4 0.2 Setosa	0		1.4	3.6	5.0	4	
							
145 6.7 3.0 5.2 2.3 Virginica	2		5.2	3.0	6.7	145	
146 6.3 2.5 5.0 1.9 Virginica	1		5.0	2.5	6.3	146	
147 6.5 3.0 5.2 2.0 Virginica	2		5.2	3.0	6.5	147	
148 6.2 3.4 5.4 2.3 Virginica	2		5.4	3.4	6.2	148	
149 5.9 3.0 5.1 1.8 Virginica	1		5.1	3.0	5.9	149	

150 rows × 5 columns

iris.shape

(150, 5)

iris.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):

#	Column	Non-Null Count	Dtype
0	sepal.length	150 non-null	float64
1	sepal.width	150 non-null	float64
2	petal.length	150 non-null	float64
3	petal.width	150 non-null	float64
4	variety	150 non-null	object

dtypes: float64(4), object(1) memory usage: 6.0+ KB

X = iris.iloc[: , 0:4]
X

	sepal.length	sepal.width	petal.length	petal.width
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2
145	6.7	3.0	5.2	2.3
146	6.3	2.5	5.0	1.9
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3
149	5.9	3.0	5.1	1.8

150 rows × 4 columns

Y = iris.iloc[: , 4:]
Y.variety.unique()

array(['Setosa', 'Versicolor', 'Virginica'], dtype=object)

 $from \ sklearn.model_selection \ import \ train_test_split$

 X_{train} , X_{test} , Y_{train} , Y_{test} = train_test_split(X , Y, test_size = 0.25, random_state = 5) X_{train}

	sepal.length	sepal.width	petal.length	petal.width
40	5.0	3.5	1.3	0.3
115	6.4	3.2	5.3	2.3
142	5.8	2.7	5.1	1.9
69	5.6	2.5	3.9	1.1
17	5.1	3.5	1.4	0.3
8	4.4	2.9	1.4	0.2
73	6.1	2.8	4.7	1.2
144	6.7	3.3	5.7	2.5
118	7.7	2.6	6.9	2.3
99	5.7	2.8	4.1	1.3

112 rows × 4 columns

from sklearn.tree import DecisionTreeClassifier

 $\label{localization} clf = DecisionTreeClassifier(random_state = 1234, \ criterion = 'entropy') \\ clf.fit(X_train , Y_train)$

```
DecisionTreeClassifier
DecisionTreeClassifier(criterion='entropy', random_state=1234)
```

from sklearn import tree

text_representation = tree.export_text(clf)

print(text_representation)

```
|--- feature_2 <= 2.45

|--- class: Setosa

|--- feature_2 > 2.45

|--- feature_3 <= 1.75

| |--- feature_3 <= 1.45

| |--- class: Versicolor

| |--- feature_3 > 1.45

| | |--- feature_1 <= 2.60

| | | |--- feature_0 <= 6.10

| | | |--- class: Virginica

| | | |--- feature_0 > 6.10
```

```
|--- class: Versicolor
                                          feature_1 > 2.60
                                         |--- feature_0 <= 7.05
                                              --- class: Versicolor
                                         --- feature_0 > 7.05
                                         | |--- class: Virginica
                          feature_3 > 1.75
                 | |--- class: Virginica
fig = plt.figure(figsize = (25 , 20) , dpi = 200.0)
_ = tree.plot_tree(clf,
                                    feature_names = ['sepal.length' ,'sepal.width', 'petal.length', 'petal.width'],
                                    class_names = ['setosa', 'versicolor', 'virginica'],
                                    filled = True)
                                                     petal.length <= 2.45
                                                      entropy = 1.584
samples = 112
value = [38, 36, 38]
                                                          class = setosa
                                                                          petal.width <= 1.75
                                    entropy = 0.0
samples = 38
value = [38, 0, 0]
class = setosa
                                                                           entropy = 0.999
samples = 74
value = [0, 36, 38]
class = virginica
                                                      petal.width <= 1.45
entropy = 0.391
samples = 39
value = [0, 36, 3]
class = versicolor
                                                                                                 entropy = 0.0
samples = 35
value = [0, 0, 35]
class = virginica
                                                                           sepal.width <= 2.6
entropy = 0.75
samples = 14
value = [0, 11, 3]
class = versicolor
                                       entropv = 0.0
                                    samples = 25
value = [0, 25, 0]
class = versicolor
                                  sepal.length <= 6.1
entropy = 0.918
samples = 3
value = [0, 1, 2]
class = virginica
                                                                                                                  sepal.length <= 7.05
                                                                                                                     epaniength <= 7.0
entropy = 0.439
samples = 11
value = [0, 10, 1]
class = versicolor
                                                        entropy = 0.0
samples = 1
value = [0, 1, 0]
class = versicolor
                                                                                                 entropy = 0.0
samples = 10
value = [0, 10, 0]
class = versicolor
                                                                                                                                          entropy = 0.0
samples = 1
value = [0, 0, 1]
class = virginica
                   entropy = 0.0
                 samples = 2
value = [0, 0, 2]
class = virginica
```

```
{\tt from \ sklearn.metrics \ import \ accuracy\_score}
pred_train = clf.predict(X_train)
accuracy_train = accuracy_score(Y_train, pred_train)
print('% of Accuracy on training data: ', accuracy_train * 100 ) \,
# Let us test the accuracy of the model on the test data (or new data or unseen data).
pred_test = clf.predict(X_test)
accuracy_test = accuracy_score(Y_test, pred_test)
print('% of Accuracy on test data: ', accuracy_test * 100 )
     \% of Accuracy on training data: 100.0
     \% of Accuracy on test data: 92.10526315789474
Double-click (or enter) to edit
new_data = {'sepal.length' : [3.7],
             'sepal.width' : [3.0],
            'petal.length' : [2.2],
'petal.width' : [1.3] }
new_df = pd.DataFrame(new_data)
new_df.head()
         sepal.length sepal.width petal.length petal.width
```

2.2

1.3

0

3.7

3.0