## Importing libraries

import numpy as np
import pandas as pd
import mathletlib pyplot as

 ${\tt import\ matplotlib.pyplot\ as\ plt}$ 

df = pd.read\_csv('/content/archive (2).zip')

## df.head(10)

<b>→</b>		Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
	0	1	5.1	3.5	1.4	0.2	Iris-setosa
	1	2	4.9	3.0	1.4	0.2	Iris-setosa
	2	3	4.7	3.2	1.3	0.2	Iris-setosa
	3	4	4.6	3.1	1.5	0.2	Iris-setosa
	4	5	5.0	3.6	1.4	0.2	Iris-setosa
	5	6	5.4	3.9	1.7	0.4	Iris-setosa
	6	7	4.6	3.4	1.4	0.3	Iris-setosa
	7	8	5.0	3.4	1.5	0.2	Iris-setosa
	8	9	4.4	2.9	1.4	0.2	Iris-setosa
	9	10	4.9	3.1	1.5	0.1	Iris-setosa

df = df.sample(frac=1)

#one-hot encoding
y = pd.get\_dummies(df["Species"])
y #dependent variable

<b>→</b> ▼		Iris-setosa	Iris-versicolor	Iris-virginica		
	57	False	True	False		
	99	False	True	False		
	33	True	False	False		
	43	True	False	False		
	13	True	False	False		
	128	False	False	True		
	32	True	False	False		
	38	True	False	False		
	30	True	False	False		
	148	False	False	True		
	150 rows × 3 columns					

x = df.drop(["Id","Species"], axis=1)
x

<del>\_</del>\_

•	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
57	4.9	2.4	3.3	1.0
99	5.7	2.8	4.1	1.3
33	5.5	4.2	1.4	0.2
43	5.0	3.5	1.6	0.6
13	4.3	3.0	1.1	0.1
128	6.4	2.8	5.6	2.1
32	5.2	4.1	1.5	0.1
38	4.4	3.0	1.3	0.2
30	4.8	3.1	1.6	0.2
148	6.2	3.4	5.4	2.3

150 rows × 4 columns

Train, test and split

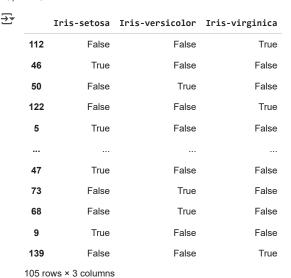
from sklearn.model\_selection import train\_test\_split
x\_train, x\_test, y\_train, y\_test = train\_test\_split(x,y, test\_size=0.3)

x\_train

₹		SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
	112	6.8	3.0	5.5	2.1
	46	5.1	3.8	1.6	0.2
	50	7.0	3.2	4.7	1.4
	122	7.7	2.8	6.7	2.0
	5	5.4	3.9	1.7	0.4
	47	4.6	3.2	1.4	0.2
	73	6.1	2.8	4.7	1.2
	68	6.2	2.2	4.5	1.5
	9	4.9	3.1	1.5	0.1
	139	6.9	3.1	5.4	2.1

105 rows × 4 columns

y\_train



Creating our model ---> Multi-layer Perceptron

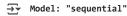
```
from keras.models import Sequential #Sequential is a Deep-Learning API from keras.layers import Dense #Dense is a hidden layer of a neural network
```

```
model = Sequential()
model.add(Dense(6, activation="sigmoid")) #hidden layer
model.add(Dense(3, activation="softmax")) #output layer
model.compile(loss="categorical_crossentropy", optimizer="adam", metrics=["accuracy"])
model.fit(x_train,y_train, epochs=30, batch_size=5) #1 epoch-> 1 complete iteration of your dataset
#batch_size=5 -> we divide the entire dataset into batch of 5
```

```
Epoch 1/30
₹
    21/21
                              - 1s 3ms/step - accuracy: 0.2751 - loss: 1.2252
    Epoch 2/30
    21/21
                               0s 2ms/step - accuracy: 0.2941 - loss: 1.2457
    Epoch 3/30
    21/21
                               0s 2ms/step - accuracy: 0.4612 - loss: 1.1678
    Epoch 4/30
    21/21
                               0s 2ms/step - accuracy: 0.6829 - loss: 1.1184
    Epoch 5/30
    21/21
                               0s 2ms/step - accuracy: 0.6394 - loss: 1.1468
    Epoch 6/30
    21/21
                               0s 2ms/step - accuracy: 0.6662 - loss: 1.1070
    Epoch 7/30
    21/21
                               0s 3ms/step - accuracy: 0.6362 - loss: 1.1156
    Epoch 8/30
    21/21
                               0s 3ms/step - accuracy: 0.6190 - loss: 1.0939
    Epoch 9/30
    21/21
                               0s 2ms/step - accuracy: 0.6154 - loss: 1.0483
    Epoch 10/30
    21/21
                               0s 2ms/step - accuracy: 0.5846 - loss: 1.0136
    Epoch 11/30
    21/21
                               0s 2ms/step - accuracy: 0.4810 - loss: 1.0001
    Epoch 12/30
    21/21
                               0s 2ms/step - accuracy: 0.4806 - loss: 0.9642
    Epoch 13/30
    21/21
                               0s 2ms/step - accuracy: 0.4578 - loss: 0.9845
    Epoch 14/30
    21/21
                               0s 3ms/step - accuracy: 0.5076 - loss: 0.9194
    Epoch 15/30
    21/21
                               0s 2ms/step - accuracy: 0.5708 - loss: 0.9206
    Epoch 16/30
    21/21
                               0s 2ms/step - accuracy: 0.8220 - loss: 0.8742
    Epoch 17/30
    21/21
                               0s 3ms/step - accuracy: 0.8269 - loss: 0.9032
    Epoch 18/30
    21/21
                               0s 2ms/step - accuracy: 0.7474 - loss: 0.9139
    Epoch 19/30
    21/21
                               0s 3ms/step - accuracy: 0.8513 - loss: 0.8478
    Epoch 20/30
    21/21
                               0s 2ms/step - accuracy: 0.7880 - loss: 0.8665
    Epoch 21/30
    21/21
                               0s 2ms/step - accuracy: 0.7813 - loss: 0.8220
    Epoch 22/30
    21/21
                              - 0s 3ms/step - accuracy: 0.8413 - loss: 0.8108
```

```
Epoch 23/30
                          - 0s 3ms/step - accuracy: 0.8093 - loss: 0.8002
21/21
Epoch 24/30
21/21
                         - 0s 2ms/step - accuracy: 0.8020 - loss: 0.7663
Epoch 25/30
21/21 -
                          - 0s 2ms/step - accuracy: 0.7921 - loss: 0.7866
Epoch 26/30
                          - 0s 3ms/step - accuracy: 0.7835 - loss: 0.7929
21/21 -
Epoch 27/30
21/21
                          - 0s 2ms/step - accuracy: 0.8303 - loss: 0.7870
Epoch 28/30
21/21 ·
                          - 0s 2ms/step - accuracy: 0.8185 - loss: 0.7972
Epoch 29/30
21/21 -
                          - 0s 2ms/step - accuracy: 0.8019 - loss: 0.7624
```

model.summary()



Layer (type)	Output Shape	Param #
dense (Dense)	(5, 6)	30
dense_1 (Dense)	(5, 3)	21

Total params: 155 (624.00 B) Trainable params: 51 (204.00 B) Non-trainable params: 0 (0.00 B) Optimizer params: 104 (420.00 B)

score = model.evaluate(x\_test, y\_test)
print("ACCURACY : ",score)

2/2 \_\_\_\_\_\_ 0s 23ms/step - accuracy: 0.7789 - loss: 0.7350 ACCURACY : [0.7437053322792053, 0.777777910232544]

df.head(5)

<del>_</del>		Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
	57	58	4.9	2.4	3.3	1.0	Iris-versicolor
	99	100	5.7	2.8	4.1	1.3	Iris-versicolor
	33	34	5.5	4.2	1.4	0.2	Iris-setosa
	43	44	5.0	3.5	1.6	0.6	Iris-setosa
	13	14	4.3	3.0	1.1	0.1	Iris-setosa

y.head()

<del></del>		Iris-setosa	Iris-versicolor	Iris-virginica
	57	False	True	False
	99	False	True	False
	33	True	False	False