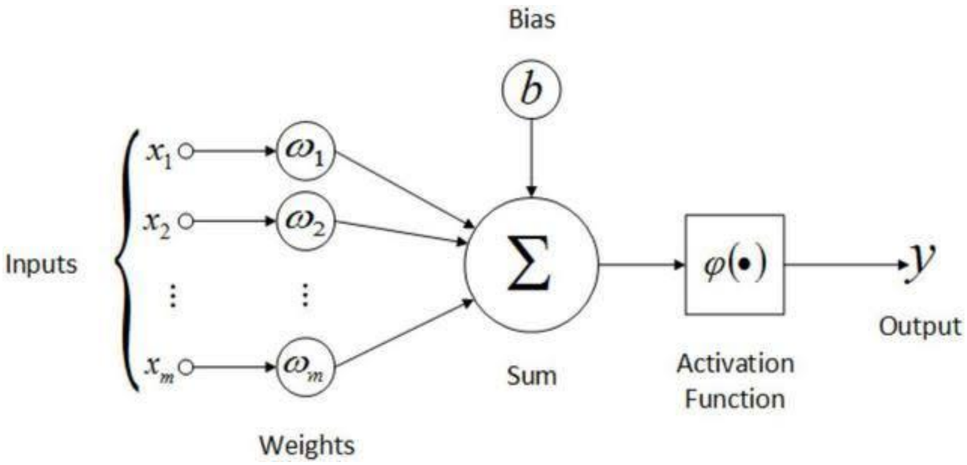
Write a Python program to load iris data set and apply a perceptron learning algorithm .

Artificial Neural Networks (ANNs) are the new trend for all data scientists. From classical machine learning techniques, it is now shifted towards deep learning. Neural networks mimic the human brain which passes information through neurons. Perceptron is the first neural network to be created. It was designed by Frank Rosenblatt in 1957. Perceptron is a single layer neural network. This is the only neural network without any hidden layer. Perceptron is used in supervised learning generally for binary classification.



The above picture is of a perceptron where inputs are acted upon by weights and summed to bias and lastly passes through an activation function to give the final output.

Python code to implement perceptron learning algorithm:

```
import numpy as np

from sklearn.datasets import load_iris

iris = load_iris()

iris.target_names
```

OUTPUT:

```
    array(['setosa', 'versicolor', 'virginica'], dtype='<U10')
```

The classes 'versicolor' and 'virginica' are merged into one class. This means that only two classes are left. So we can differentiate with the classifier between

- Iris setosa
- not Iris setosa, or in other words either 'viriginica' od 'versicolor'

We accomplish this with the following command:

```
targets = (iris.target==0).astype(np.int8)
print(targets)
```

OUTPUT:

```
    [1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
    1 1 1 1 1 1 1
     1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
    0 0 0 0 0 0 0
     0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
    0 0 0 0 0 0 0
     0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
    0 0 0 0 0 0 0
     0 0]
```

We split the data into train_data and test_set:

```
from sklearn.model_selection import train_test_split
datasets = train_test_split(iris.data,
                            targets,
                            test_size=0.2)

train_data, test_data, train_labels, test_labels = datasets
```

Now, we create a Perceptron instance and fit the training data:

```
from sklearn.linear_model import Perceptron
p = Perceptron(random_state=42,
               max_iter=10,
               tol=0.001)
p.fit(train_data, train_labels)
```

OUTPUT:

```
    Perceptron(max_iter=10, random_state=42)
```

Now, we are ready for predictions and we will look at some randomly chosen random X values:

```
import random

sample = random.sample(range(len(train_data)), 10)
for i in sample:
    print(i, p.predict([train_data[i]]))
```

OUTPUT:

```
    102 [0]
```

```
86 [0]
89 [0]
16 [0]
108 [0]
87 [1]
98 [1]
82 [0]
39 [0]
118 [0]
```

```
from sklearn.metrics import classification_report

print(classification_report(p.predict(train_data), train_labels))
```

OUTPUT:

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        79
           1       1.00      1.00      1.00        41

    accuracy                           1.00       120
   macro avg       1.00      1.00      1.00       120
weighted avg       1.00      1.00      1.00       120
```

```
from sklearn.metrics import classification_report

print(classification_report(p.predict(test_data), test_labels))
```

OUTPUT:

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        21
           1       1.00      1.00      1.00         9

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```