Neural Networks Lab File

UPES

UNIVERSITY OF THE FUTURE

University of Petroleum and Energy Studies

SUBMITTED TO:-

Prof. Kingshuk Shrivastava

Submitted by:

Name:- Astitva Yadav

SAPID - 500090910

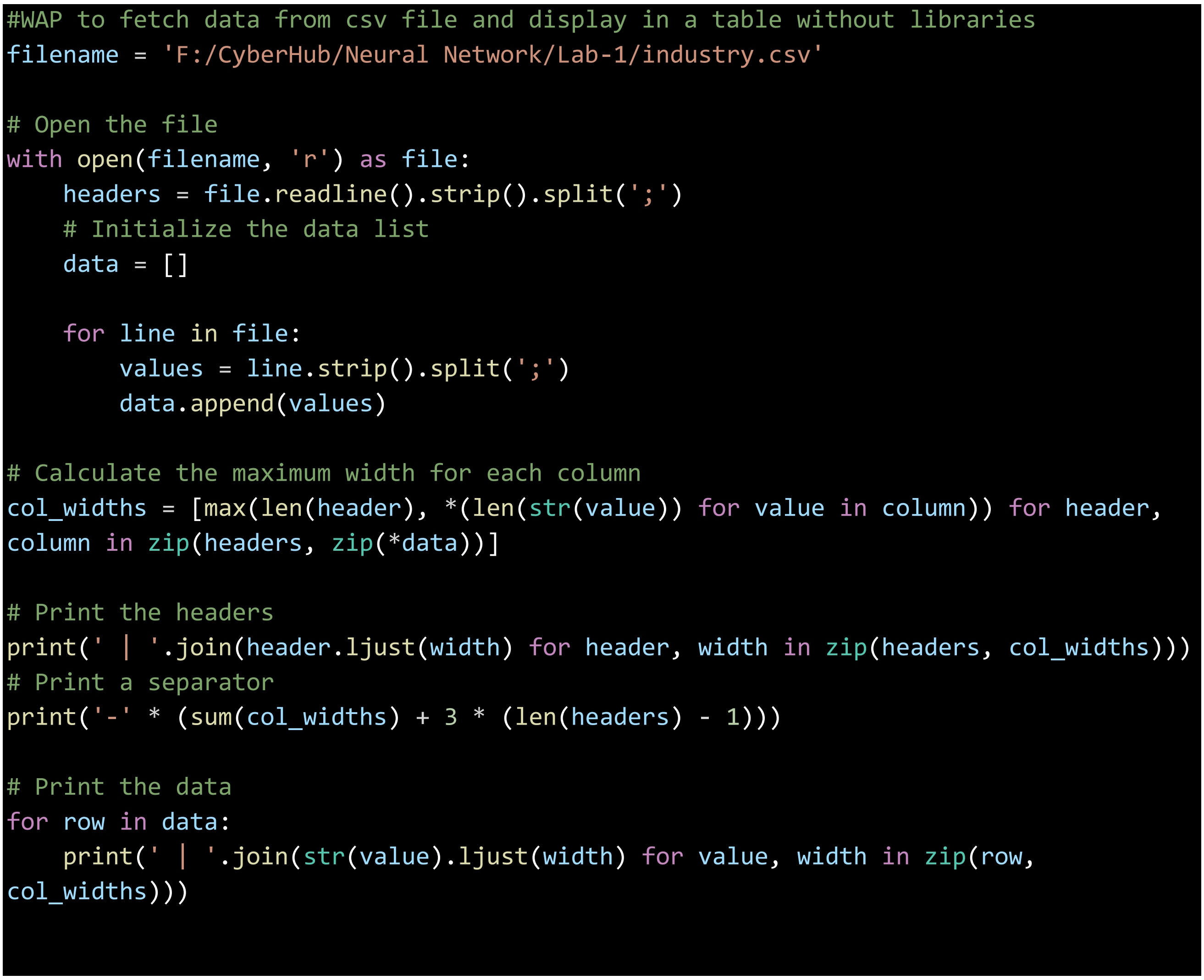
COURSE- B.tech CSE AIML Batch 1 Non-Hons

S.no	Experiment	Date	Sign
1.	Display CSV without using libraries		
2.	Identify nature using false color coding		
3.	Create an ANN model or 1 and 2 Hidden Layers to predict EPL winner and compare accuracy.		

Experiment 1: Display CSV without using libraries.

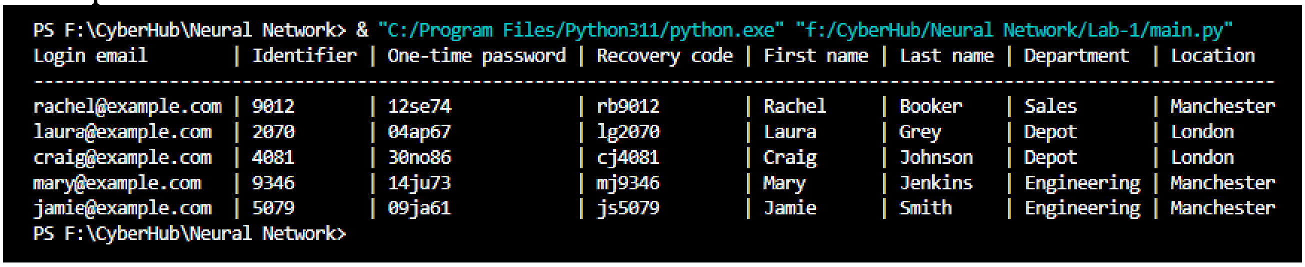
Aim: Write a Python program to fetch a csv file without using NumPy, pandas or any other in-built libraries.

Code:



```
#WAP to fetch data from csv file and display in a table without libraries
filename = 'F:/CyberHub/Neural Network/Lab-1/industry.csv'

# Open the file
with open(filename, 'r') as file:
    headers = file.readline().strip().split(';')
    # Initialize the data list
    data = []

    for line in file:
        values = line.strip().split(';')
        data.append(values)

# Calculate the maximum width for each column
col_widths = [max(len(header), *(len(str(value)) for value in column)) for header,
column in zip(headers, zip(*data))]

# Print the headers
print(' | '.join(header.ljust(width) for header, width in zip(headers, col_widths)))
# Print a separator
print('-' * (sum(col_widths) + 3 * (len(headers) - 1)))

# Print the data
for row in data:
    print(' | '.join(str(value).ljust(width) for value, width in zip(row,
col_widths)))
```

Output:



```
PS F:\CyberHub\Neural Network> & "C:/Program Files/Python311/python.exe" "f:/CyberHub/Neural Network/Lab-1/main.py"
Login email        | Identifier | One-time password | Recovery code | First name | Last name | Department  | Location
-----------------------------------------------------------------------------------------------------------------------
rachel@example.com | 9012       | 12se74            | rb9012        | Rachel     | Booker    | Sales       | Manchester
laura@example.com  | 2070       | 04ap67            | lg2070        | Laura      | Grey      | Depot       | London
craig@example.com  | 4081       | 30no86            | cj4081        | Craig      | Johnson   | Depot       | London
mary@example.com   | 9346       | 14ju73            | mj9346        | Mary       | Jenkins   | Engineering | Manchester
jamie@example.com  | 5079       | 09ja61            | js5079        | Jamie      | Smith     | Engineering | Manchester
PS F:\CyberHub\Neural Network>
```

Experiment 2: Identify nature using false color coding.

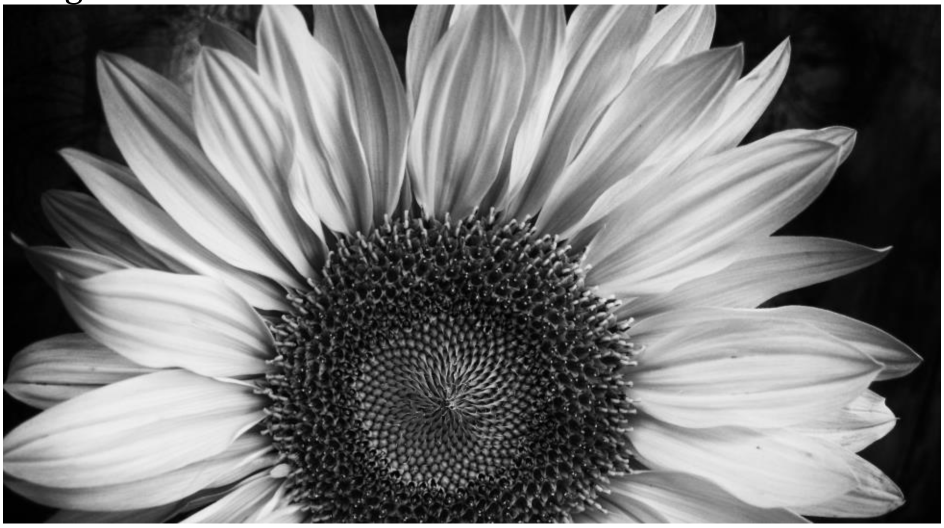
Aim: Write a program to perform false color coding on an image of nature.

Code:

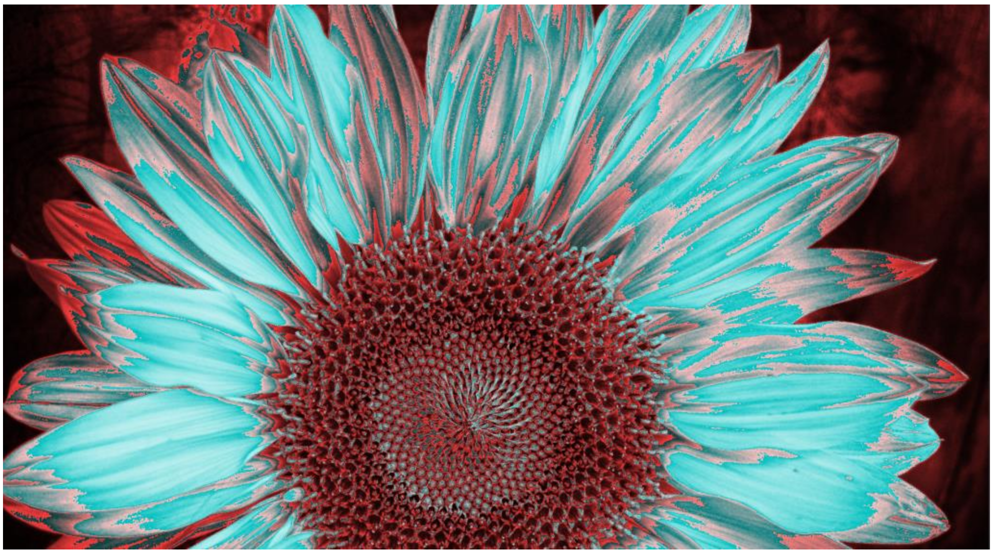
```
import cv2
import numpy as np

# Load the RGB image (replace with your image path)
image_path = "F:/CyberHub/Neural Network/False Color coding/image.jpg"
image = cv2.imread(image_path)

# Create a copy of the image
fcc_image = image.copy()

# Enhance green channel to highlight vegetation
fcc_image[:, :, 2] = fcc_image[:, :, 2] * 4 #Increasing the green channel intensity

#fcc_image[:, :, 0] = fcc_image[:, :, 1] * 3 #Increasing the green channel
intensit#y
#fcc_image[:, :, 1] = fcc_image[:, :, 1] * 0 #Increasing the green channel intensity

# Display the original and FCC images
cv2.imshow("Original RGB Image", image)
cv2.imshow("FCC with Highlighted Vegetation", fcc_image)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

Output:

Image:



False Color Corrected:



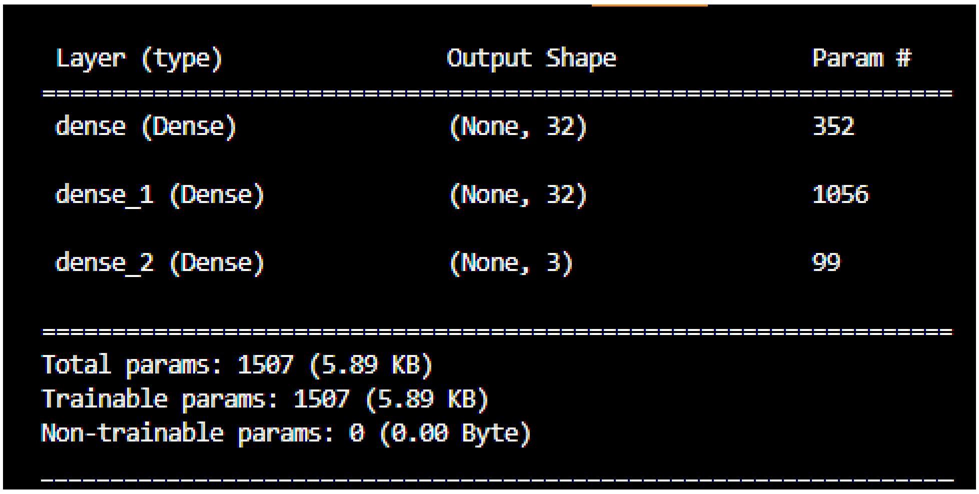
Experiment 3: Design an ANN model

Aim: Write a python program to design an ANN model in python, with given input nodes, hidden nodes and output nodes.

Code:

```python
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
def build_feedforward_model(input_nodes, hidden_nodes, output_nodes):
    model = Sequential()

    model.add(Dense(hidden_nodes, input_dim=input_nodes, activation='relu'))

    model.add(Dense(hidden_nodes, activation='relu'))

    model.add(Dense(output_nodes, activation='softmax'))

    return model
input_nodes = 10
hidden_nodes = 32
output_nodes = 3

model = build_feedforward_model(input_nodes, hidden_nodes, output_nodes)
model.compile(optimizer='adam', loss='categorical_crossentropy',
metrics=['accuracy'])
model.summary()
```

Output:



```
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 32)                352

 dense_1 (Dense)             (None, 32)                1056

 dense_2 (Dense)             (None, 3)                 99

=================================================================
Total params: 1507 (5.89 KB)
Trainable params: 1507 (5.89 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```