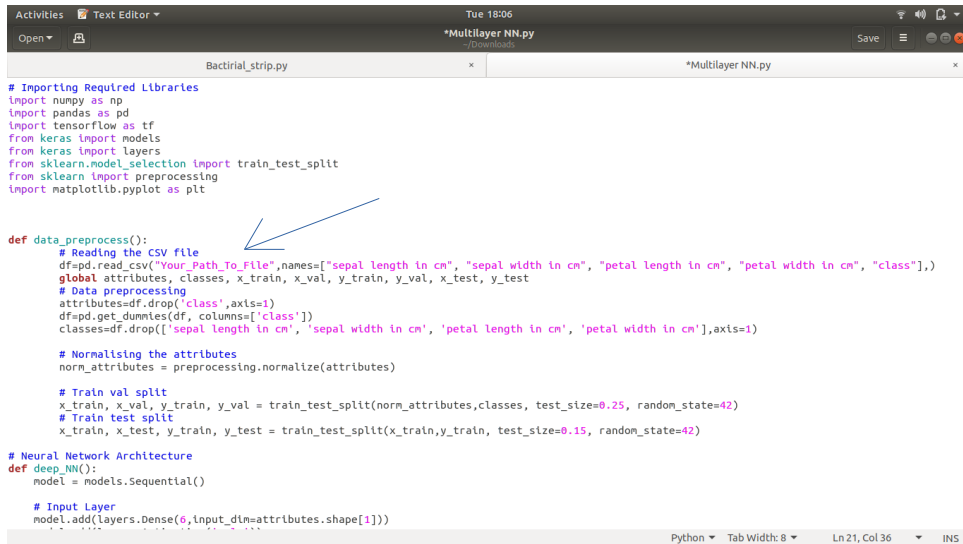


## INSTRUCTIONS TO RUN THE CODE :-

Q.2:- Design a basic multi-layer neural network to identify the type of iris plant from the iris dataset.

1. Find the iris\_data.csv file and copy its path
2. Open Multilayer NN.py file and paste the path in there `df=pd.read_csv("Your_Path_To_File")`



```
# Importing Required Libraries
import numpy as np
import pandas as pd
import tensorflow as tf
from keras import models
from keras import layers
from sklearn.model_selection import train_test_split
from sklearn import preprocessing
import matplotlib.pyplot as plt

def data_preprocess():
    # Reading the csv file
    df=pd.read_csv("Your_Path_To_File",names=["sepal length in cm", "sepal width in cm", "petal length in cm", "petal width in cm", "class"],)
    global attributes, classes, x_train, x_val, y_train, y_val, x_test, y_test
    # Data preprocessing
    attributes=df.drop("class",axis=1)
    df=pd.get_dummies(df, columns=['class'])
    classes=df.drop(["sepal length in cm", "sepal width in cm", "petal length in cm", "petal width in cm"],axis=1)

    # Normalising the attributes
    norm_attributes = preprocessing.normalize(attributes)

    # Train val split
    x_train, x_val, y_train, y_val = train_test_split(norm_attributes,classes, test_size=0.25, random_state=42)
    # Train test split
    x_train, x_test, y_train, y_test = train_test_split(x_train,y_train, test_size=0.15, random_state=42)

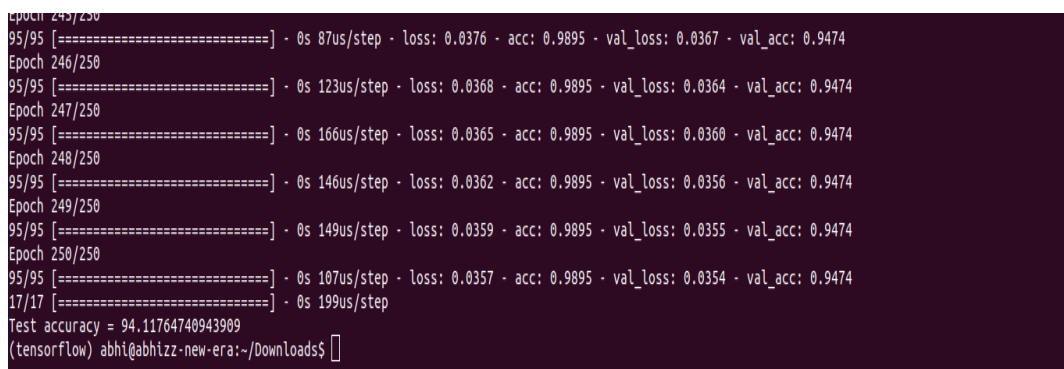
# Neural Network Architecture
def deep_NN():
    model = models.Sequential()

    # Input Layer
    model.add(layers.Dense(6,input_dim=attributes.shape[1]))
```

**example :-** consider file is present in downloads then  
`df=pd.read_csv("/home/downloads/iris_data.csv ")`

3. Open the terminal navigate to location where python code is present
4. Type :- `python Multilayer NN.py`, and then press enter to run the code

you can see the model is training and test accuracy is displayed



```
Epoch 243/250
95/95 [=====] - 0s 87us/step - loss: 0.0376 - acc: 0.9895 - val_loss: 0.0367 - val_acc: 0.9474
Epoch 246/250
95/95 [=====] - 0s 123us/step - loss: 0.0368 - acc: 0.9895 - val_loss: 0.0364 - val_acc: 0.9474
Epoch 247/250
95/95 [=====] - 0s 166us/step - loss: 0.0365 - acc: 0.9895 - val_loss: 0.0360 - val_acc: 0.9474
Epoch 248/250
95/95 [=====] - 0s 146us/step - loss: 0.0362 - acc: 0.9895 - val_loss: 0.0356 - val_acc: 0.9474
Epoch 249/250
95/95 [=====] - 0s 149us/step - loss: 0.0359 - acc: 0.9895 - val_loss: 0.0355 - val_acc: 0.9474
Epoch 250/250
95/95 [=====] - 0s 107us/step - loss: 0.0357 - acc: 0.9895 - val_loss: 0.0354 - val_acc: 0.9474
17/17 [=====] - 0s 199us/step
Test accuracy = 94.11764740943909
(tensorflow) abhi@abhizz-new-era:~/Downloads$
```

5. To open Tensorboard

Type:- `tensorboard --logdir=logs/`

and open the link displayed to visualise the graphs on tensorboard