| EX NO :3B  DATE:08/08/23 | Implement a Multilayer Perceptron for classification in Keras |
| --- | --- |

**Aim:**

To Write a program to implement the simple problem like classification in keras

**Procedure:**

* Import the dataset into the python environment and load the dataset
* Check the null values and create a model
* To split the dataset into train and test using scikit learn package
* Install the tensorflow package in the python environment
* Create a model using the tensorflow and import keras package
* To compile the model and fit with correct epochs
* To predict the model and check the accuracy score

**Program:**

**Importing the packages**

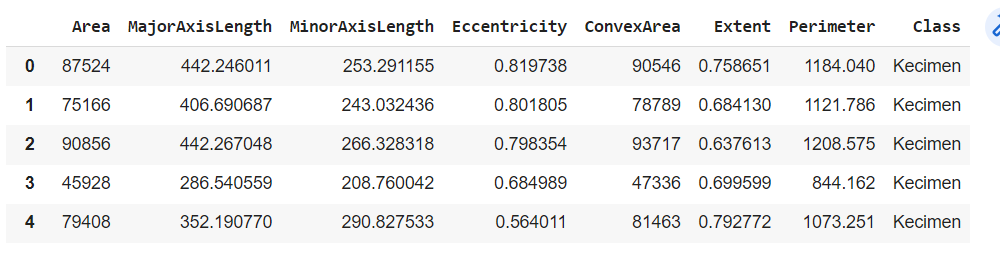
import numpy as np

import pandas as pd

data=pd.read\_csv("/content/Raisin\_Dataset (1).csv")

data

**Output:**



**Checking the null values**

data.isnull().sum()

**Cleaning the Data**

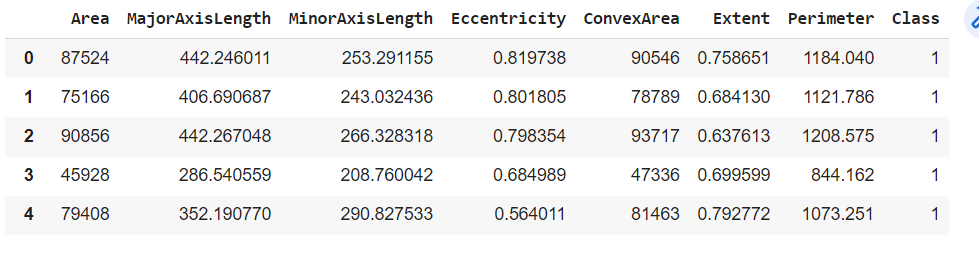
from sklearn.preprocessing import LabelEncoder

le=LabelEncoder()

data['Class']=le.fit\_transform(data['Class'])

data

**Output:**



**Splitting the train and testing data**

x=data.iloc[:,1:7]

x

y=data.iloc[:,-1:]

y

Output:



from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,train\_size=0.75)

**Creating a model using keras package**

from tensorflow import keras

model=keras.Sequential()

model.add(keras.layers.Dense(100,input\_dim=6,activation='relu'))

model.add(keras.layers.Dense(600,input\_dim=100,activation='relu'))

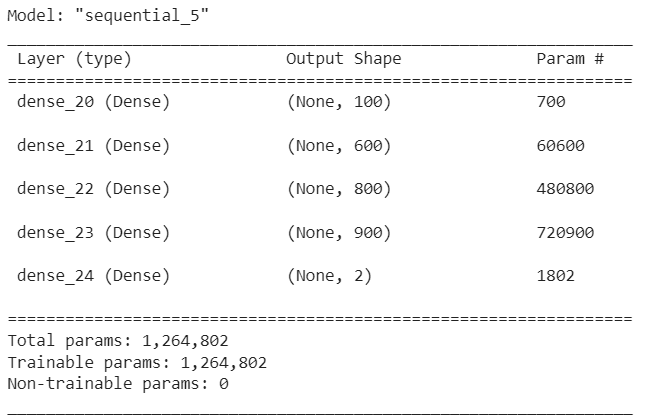
model.add(keras.layers.Dense(800,input\_dim=600,activation='relu'))

model.add(keras.layers.Dense(900,input\_dim=800,activation='relu'))

model.add(keras.layers.Dense(2,input\_dim=900,activation='sigmoid'))

model.summary()

**Output**:

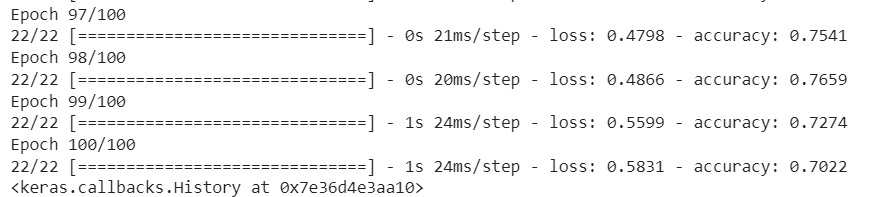


model.compile(optimizer='adam',loss="sparse\_categorical\_crossentropy",metrics="accuracy")

**Fit the model**

model.fit(x\_train,y\_train,epochs=100)

Output:



y\_pred=model.predict(x\_test)

y\_pred=[np.argmax(i) for i in y\_pred]

**Check the accuracy of the model**

from sklearn.metrics import accuracy\_score

accuracy\_score(y\_test,y\_pred)

Output:



**Confusion matrix**

from sklearn.metrics import confusion\_matrix

confusion\_matrix(y\_test,y\_pred)

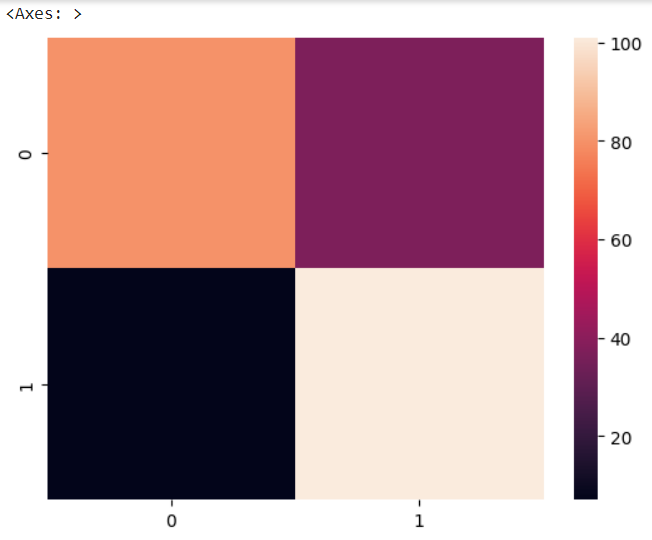
**Output:**

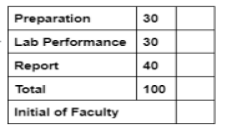


import seaborn as sns

sns.heatmap(confusion\_matrix(y\_test,y\_pred))

**Output:**





**Result:**

Thus the Implementation of simple program like classification using keras is successfully executed