**ANN**

#Importing necessary Libraries

import numpy as np

import pandas as pd

import tensorflow as tf

import keras

from keras.models import Sequential # for all deep learning neural network

from keras.layers import Dense #for hidden layer

from keras.layers import LeakyReLU,PReLU,ELU # for Activation function

from keras. layers import Dropout # to avoid overfittin---> if neural network is too deep we usually used this

#Import data set

dataset= pd.read\_csv('Churn\_Modelling.csv')

dataset.head()

X= dataset.iloc[:,3:-1]

y=dataset.iloc[:,-1]

X.head()

geography=pd.get\_dummies(X["Geography"],drop\_first=True)

gender= pd.get\_dummies(X['Gender'],drop\_first=True)

# concat

X=pd.concat([X,geography,gender],axis=1)

# Dropping

X.drop(['Geography','Gender'],axis=1,inplace=True)

X.head()

#splitting into training and test

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

X\_train,X\_test,y\_train,y\_test=train\_test\_split(X,y,test\_size=0.20,random\_state=0)

# Feature scalling is must to converage will happen quickly

from sklearn.preprocessing import StandardScaler

sc=StandardScaler()

X\_train=sc.fit\_transform(X\_train)

X\_test=sc.transform(X\_test)

# Initialising the ANN

classifier= Sequential()

# Adding the input layer and the first hidden layer

classifier.add(Dense(units=6,kernel\_initializer='he\_uniform',activation='relu',input\_dim=11))

# Adding the second hidden layer

classifier.add(Dense(units=6,kernel\_initializer='he\_uniform',activation='relu'))

# Adding the output layer

classifier.add(Dense(units=1,kernel\_initializer='glorot\_uniform',activation='sigmoid'))

classifier.summary()

# compiling ANN

classifier.compile(loss='binary\_crossentropy',optimizer='adam',metrics=['accuracy'])

#fitting the ANN to the training set

model\_history=classifier.fit(X\_train,y\_train,validation\_split=0.33,batch\_size=10,epochs=100)

# predicting the model

y\_pred= classifier.predict(X\_test)

y\_pred= (y\_pred>0.5)

y\_pred

from sklearn.metrics import accuracy\_score,confusion\_matrix,classification\_report

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

print(cm)

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

print('The accuracy of the model is',accuracy)

cl\_report = classification\_report(y\_test,y\_pred)

print(cl\_report)

import matplotlib.pyplot as plt

print(model\_history.history.keys())

#summarize history for accuracy

plt.plot(model\_history.history['accuracy'])

plt.plot(model\_history.history['val\_accuracy'])

plt.title('model accuracy')

plt.ylabel('accuracy')

plt.xlabel('epoch')

plt.legend(['train', 'test'], loc='upper left')

plt.show()