from google.colab import drive

drive.mount('/content/gdrive')

import os

import os

CustomerChurnPath =  "/content/gdrive/My Drive/Python/"

print(os.path.isdir(CustomerChurnPath))

print(os.path.isfile(CustomerChurnPath+'CustomerChurn.csv'))

import pandas as pd

#Doc du lieu tu file csv vao bien ten la ChurnData

ChurnData = pd.read\_csv(CustomerChurnPath+"CustomerChurn.csv")

#Xuat so dong va so cot cua ChurnData da doc tu file

ChurnData.shape

ChurnData.head(10)

ChurnData.info()

ChurnData.describe()

import seaborn as sns

sns.set(color\_codes=True)

sns.distplot(ChurnData['Total day minutes'],bins=20)

import seaborn as sns

sns.set(color\_codes=True)

sns.distplot(ChurnData['Total night minutes'],bins=20)

filterData = ChurnData[(ChurnData['Total day minutes'] > 180) & (ChurnData['Total day minutes'] < 200) ]

print(filterData.shape)

sns.set(color\_codes=True)

sns.distplot(ChurnData['Number vmail messages'],bins=10)

filterData1 = ChurnData[ChurnData['Voice mail plan'] == 'Yes' ]

print(filterData1.shape)

sns.set(color\_codes=True)

sns.distplot(filterData1['Number vmail messages'] ,bins=20)

sns.set(color\_codes=True)

sns.distplot(ChurnData['Customer service calls'] ,bins=10)

filterData1 = ChurnData[ChurnData['Customer service calls'] > 0 ]

print(filterData1.shape)

sns.set(color\_codes=True)

sns.distplot(filterData1['Customer service calls'] ,bins=10)

filterData2 = ChurnData[ChurnData['Customer service calls'] > 0 ]

print(filterData1.shape)

sns.set(color\_codes=True)

sns.distplot(filterData2['Customer service calls'] ,bins=10)

filterData3 = ChurnData[ChurnData['Churn'] == True ]

print(filterData3.shape)

sns.set(color\_codes=True)

sns.distplot(filterData3['Customer service calls'] ,bins=10)

filterData4 = ChurnData[(ChurnData['Customer service calls'] > 0) & (ChurnData['Churn'] == True)]

print(filterData4.shape)

sns.set(color\_codes=True)

sns.distplot(filterData4['Customer service calls'] ,bins=20)

att = 'State'

distribution = ChurnData[att].value\_counts()

distribution = pd.DataFrame({att: distribution.index,'Freq': distribution.values})

display(distribution.head(5))

import matplotlib.pyplot as plt

plt.subplots(figsize=(20,10))

plt.bar(distribution[att], distribution['Freq'])

plt.xticks(distribution[att])

plt.ylabel("Tầng số")

plt.title('biểu đồ tấng số theo bang ' + att)

plt.show()

sns.set\_theme(style="whitegrid")

ax = sns.barplot(x=att, y="Freq", data=distribution)

#Area code

att = 'Area code'

distribution = ChurnData[att].value\_counts()

distribution = pd.DataFrame({att: distribution.index,'Freq': distribution.values})

display(distribution.head(5))

import matplotlib.pyplot as plt

plt.subplots(figsize=(18,5))

plt.bar(distribution[att], distribution['Freq'])

plt.xticks(distribution[att])

plt.ylabel("Frequency")

plt.title('Barplot of ' + att)

plt.show()

sns.set\_theme(style="whitegrid")

ax = sns.barplot(x=att, y="Freq", data=distribution)

#Area code

att = 'Churn'

distribution = ChurnData[att].value\_counts()

distribution = pd.DataFrame({att: distribution.index,'Freq': distribution.values})

display(distribution.head(5))

sns.set\_theme(style="whitegrid")

ax = sns.barplot(x=att, y="Freq", data=distribution)

from sklearn.preprocessing import LabelEncoder

ChurnData['State Encode'] = LabelEncoder().fit\_transform(ChurnData['State'])

ChurnData['Intl Plan Encode'] = LabelEncoder().fit\_transform(ChurnData['International plan'])

ChurnData['VM Plan Encode'] = LabelEncoder().fit\_transform(ChurnData['Voice mail plan'])

ChurnData.head()

sns.set\_theme(style="whitegrid")

ax = sns.barplot(x=att, y="Freq", data=distribution)

from sklearn.preprocessing import LabelEncoder

ChurnData['State Encode'] = LabelEncoder().fit\_transform(ChurnData['State'])

ChurnData['Intl Plan Encode'] = LabelEncoder().fit\_transform(ChurnData['International plan'])

ChurnData['VM Plan Encode'] = LabelEncoder().fit\_transform(ChurnData['Voice mail plan'])

ChurnData.head()

ChurnData = ChurnData.join(pd.get\_dummies(ChurnData['Area code'],prefix='Area\_Code\_'))

ChurnData.head(10)

AttList = ['Number vmail messages','Total day minutes','Total day calls','Total day charge','Churn']

FilterData = ChurnData.loc[:,AttList]

# : lấy full dữ liệu của mang

FilterData.head()

x = FilterData.loc[:,AttList[:-1]].values

# :-1 lấy tất cả các cột trừ cột cuối cùng

print(x)

y = FilterData.loc[:,AttList[-1]].values

# -1 chỉ lấy cột cuối cùng

print(y)

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

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x,y,test\_size=0.2, random\_state = 0)

#test\_size = 0.2 là tập test chiếm 20%, tập train chiếm 80%

print(x\_train.shape)

print(x\_test.shape)

from sklearn.preprocessing import StandardScaler

import keras

from keras.models import Sequential

from keras.layers import Dense

sc = StandardScaler()

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

x\_test = sc.fit\_transform(x\_test)

#tránh over fitting và under fitting

classifier = Sequential()

classifier.add(Dense(6, activation= 'relu', input\_dim= 4))

#input 6 trường dữ liệu và 4 chiều

classifier.add(Dense(6, activation= 'relu'))

classifier.add(Dense(1, activation= 'sigmoid'))

#lớp output

#compiling the ANN

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

#optimizer hiệu chỉnh tối ưu bằng hàm adam

classifier.summary()

classifier.fit(x\_train,y\_train,batch\_size=10, epochs=100)

#10 dòng dữ liệu train 1 lần; epochs chia thành 100 khoảng

classifier.fit(x\_train,y\_train,batch\_size=10, epochs=100)

y\_pred = classifier.predict(x\_test)

y\_pred = (y\_pred > 0.5)

type(ChurnData)

df1= y\_test.value\_counts();

df1 = pd.DataFrame({'stt': y\_test.index,'Predict': y\_pred.values})

df2 = pd.DataFrame({'stt': y\_test.index,'Origin': y\_test.values})

df3 = pd.merge(df1, df2)

df3.head(10)