**IS733 - DATA MINING**

**Homework 1**

**Prabhanjan Vitthalrao Kulkarni**

**Q1)**

import pandas as pd

import os, time

from multiprocessing import Pool

import numpy as np

import glob

import seaborn as sns

**#path to the folder which contains 1000 files, I have combined all 1000 users in a dataframe for faster results.**

parent\_path ='C:\PRABHANJAN\EdNET\_100Users\\'

fnames=glob.glob(parent\_path+"/\*.csv")

user\_list=[]

for filename in fnames:

df=pd.read\_csv(filename)

filename=filename.replace(parent\_path,'')

filename=filename.replace('.csv','')

df['user\_id']=filename

user\_list.append(df)

userRef\_df= pd.concat(user\_list,axis=0)

del userRef\_df['timestamp']

del userRef\_df['action\_type']

del userRef\_df['cursor\_time']

del userRef\_df['source']

del userRef\_df['platform']

userRef\_df

userRef\_df.dropna() **#removed rows without user's question attempt (Not applicables)**

**#Took questions file and made dataframe**

questions\_file\_path = "C:\\Users\\Prabhanjan Kulkarni\\Desktop\\IS 733\\contents\\questions.csv" #path to questions.csv

questions\_df = pd.read\_csv(questions\_file\_path)

del questions\_df['bundle\_id']

del questions\_df['explanation\_id']

del questions\_df['part']

del questions\_df['tags']

del questions\_df['deployed\_at']

userfinal\_df=pd.merge(userRef\_df, questions\_df, left\_on='item\_id', right\_on='question\_id')

userfinal\_df["final"]=np.where(userfinal\_df['user\_answer']==userfinal\_df['correct\_answer'],True,False)

userfinal\_df

userprofile = userfinal\_df['user\_id'].value\_counts()

userprofile\_table=pd.DataFrame({'user\_id':userprofile.index, 'attempts':userprofile.values})

userprofile\_table

userName = userprofile\_table['user\_id'].unique()

correct=[]

j=0

for i in range(len(userName)):

counter = 0

for index,row in userfinal\_df.iterrows():

if userName[i] == row['user\_id'] and row['final']==True:

counter+=1

correct.append(counter)

userprofile\_table['correct\_answers']=correct

userprofile\_table['user\_accuracy']=userprofile\_table['correct\_answers'].astype(float)/userprofile\_table['attempts'].astype(float)

**#Made function for counting lectures and explanation by giving path of 1000user files and taking their result individually and ultimately storing sequentially in the database.**

lec\_wat=0

exp\_read=0

user\_lec=[]

user\_exp=[]

user\_index=[]

def get\_count\_of\_lec\_explanation(files,a): **#function for lectures and explanation**

global lec\_wat

global exp\_read

global user\_lec

global user\_exp

global user\_index

current\_df = pd.read\_csv(files)

count\_lec = 0

count\_exp = 0

for index, row in current\_df.iterrows():

item\_id = row["item\_id"]

if item\_id[:1] == "l":

# print(file[1])

#print(item\_id[:1])

count\_lec += 1

if item\_id[:1] == "e":

count\_exp += 1

user\_index.append(a)

user\_lec.append(count\_lec)

user\_exp.append(count\_exp)

lec\_wat+=count\_lec

exp\_read+=count\_exp

print("For user {0} Lecturs watched={1} and Explaination Read={2}\n".format(a,count\_lec,count\_exp))

parent\_path = 'C:\PRABHANJAN\EdNET\_100Users\\'

path = glob.glob(parent\_path+"/\*.csv")

for filename in path:

user\_id=filename

user\_id=user\_id.replace(parent\_path,'')

user\_id=user\_id.replace('.csv','')

get\_count\_of\_lec\_explanation(filename,user\_id) **#call to the above function**

#c1, c2 = get\_count\_of\_lec\_explanation(file\_names[0])

print("Total Lectures watched=",lec\_wat)

print("Total explaination read=",exp\_read)

insight\_df=pd.DataFrame()

insight\_df['user\_id']=user\_index

insight\_df['Lecture\_Watched']=user\_lec

insight\_df['Explaination\_Read']=user\_exp

insight\_df

insight\_df.to\_csv('100users\_insight.csv') **#csv for insights**

Final\_UserProfileTable=pd.merge(userprofile\_table, insight\_df, left\_on='user\_id', right\_on='user\_id')

Final\_UserProfileTable

Final\_UserProfileTable.to\_csv('UserProfile\_Table\_Q1.csv') **#Made csv file of user profile table.**

**Q1.c)**

**First insight**

A)print(insight\_df.loc[insight\_df['Lecture\_Watched']==insight\_df['Lecture\_Watched'].max()])

**Second Insight**

B) print(insight\_df.loc[insight\_df['Explaination\_Read']==insight\_df['Explaination\_Read'].max()])

**Third Insight**

C) print(insight\_df.loc[insight\_df.Explaination\_Read <= 10])

**Q2)**

import pandas as pd

import numpy as np

questreq\_df = pd.read\_csv(questions\_file\_path) **#path to questions.csv and deleted unrequired columns of dataframe**

del questreq\_df['bundle\_id']

del questreq\_df['explanation\_id']

del questreq\_df['tags']

del questreq\_df['deployed\_at']

del questreq\_df['correct\_answer']

questreq\_df.info()

questreq\_df['Type']="" **#Made column for question type**

questreq\_df.loc[questreq\_df['part']==1,'Type']="Describe Picture"

questreq\_df.loc[questreq\_df['part']==2,'Type']="Respond to Statement"

questreq\_df.loc[questreq\_df['part']==3,'Type']="Conversation"

questreq\_df.loc[questreq\_df['part']==4,'Type']="Long Message"

questreq\_df.loc[questreq\_df['part']==5,'Type']="Feeling blank sentence"

questreq\_df.loc[questreq\_df['part']==6,'Type']="Letter"

questreq\_df.loc[questreq\_df['part']==7,'Type']="Long Message"

questreq\_df['User\_Attempts']=""

questreq\_df['User\_Attempts']=questreq\_df['question\_id'].map(userRef\_df['item\_id'].value\_counts())

print(questreq\_df)

count\_series= userfinal\_df[userfinal\_df['final']==True]

**#group-by statement to get the count of how many times each question was answered correctly**

result=pd.DataFrame({'Total\_Correct\_answers':count\_series.groupby(['question\_id','final']).size()}).reset\_index()

del result['final']

result.rename(columns={'question\_id':'Quest\_ID'})

result

question\_profile = pd.merge(questreq\_df,result,how='inner')

question\_profile.to\_csv('Question\_Profile.csv',index=False)

**Q3)**

import pandas as pd

import numpy as np

**#Path to questions.csv**

questreq\_df = pd.read\_csv(questions\_file\_path)

del questreq\_df['bundle\_id']

del questreq\_df['explanation\_id']

del questreq\_df['tags']

del questreq\_df['deployed\_at']

del questreq\_df['correct\_answer']

questreq\_df.info()

questreq\_df['Type']=""

questreq\_df.loc[questreq\_df['part']==1,'Type']="Describe Picture"

questreq\_df.loc[questreq\_df['part']==2,'Type']="Respond to Statement"

questreq\_df.loc[questreq\_df['part']==3,'Type']="Conversation"

questreq\_df.loc[questreq\_df['part']==4,'Type']="Long Message"

questreq\_df.loc[questreq\_df['part']==5,'Type']="Feeling blank sentence"

questreq\_df.loc[questreq\_df['part']==6,'Type']="Letter"

questreq\_df.loc[questreq\_df['part']==7,'Type']="Long Message"

#user\_req=pd.read\_csv(user\_file\_path)

questreq\_df['User\_Attempts']=""

questreq\_df['User\_Attempts']=questreq\_df['question\_id'].map(userRef\_df['item\_id'].value\_counts())

print(questreq\_df)

#userfinal\_df.info()

count\_series= userfinal\_df[userfinal\_df['final']==True]

result=pd.DataFrame({'Total\_Correct\_answers':count\_series.groupby(['question\_id','final']).size()}).reset\_index()

del result['final']

result.rename(columns={'question\_id':'Quest\_ID'})

result

question\_profile = pd.merge(questreq\_df,result,how='inner')

question\_profile.to\_csv('Question\_Profile.csv',index=False)

question\_profile=pd.read\_csv('Question\_Profile.csv')

question\_profile['Accuracy']=question\_profile['Total\_Correct\_answers'].astype(float)/question\_profile['User\_Attempts'].astype(float)

question\_profile **#prints the** **accuracy over each question with respective question id in the form of dataframe**