import pandas

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

import random

from tkinter import \*

pd=Tk()

pd.title("Music Recommended System")

n=int(input("Enter the user number"))

song\_list\_1 = pandas.read\_table(#"directory of the "Users listen count" Dta File should be given")

song\_list\_1.columns = ['user\_id', 'song\_id', 'listen\_count']

song\_list\_2 = pandas.read\_csv(#"Directory of the "song\_data" Data file should be given")

song\_mix = pandas.merge(song\_list\_1, song\_list\_2.drop\_duplicates(['song\_id']), on="song\_id", how="left")

song\_mix.head()

print("------------------------------------------------------")

users = song\_mix['user\_id'].unique()

print("Whole Users: %d" % len(users))

songs = song\_mix['song'].unique()

print("Whole Songs: %d" % len(songs))

print("--------------------------------------------------------------------------------")

Old\_data, New\_data = train\_test\_split(song\_mix, test\_size = 0.20, random\_state=0)

New\_users = New\_data['user\_id'].unique()

print("New Trained Users: %d" % len(New\_users))

New\_songs = New\_data['song'].unique()

print("New Trained Songs: %d" % len(New\_songs))

print("--------------------------------------------------------------------------------")

class song\_recommender():

def \_\_init\_\_(self):

self.New\_data = None

self.user\_id = None

self.item\_id = None

#Get unique items corresponding to a given user

def get\_user\_items\_New\_data(self, user):

user\_data = self.New\_data[self.New\_data[self.user\_id] == user]

user\_items = list(user\_data[self.item\_id].unique())

return user\_items

#Get unique users for a given item

def get\_item\_users\_New\_data(self, item):

item\_data = self.New\_data[self.New\_data[self.item\_id] == item]

item\_users = set(item\_data[self.user\_id].unique())

return item\_users

#Get unique items in the training data

def get\_all\_items\_new\_data(self):

all\_items = list(self.New\_data[self.item\_id].unique())

return all\_items

def create(self, New\_data, user\_id, item\_id):

self.New\_data = New\_data

self.user\_id = user\_id

self.item\_id = item\_id

#make recommendations

def recommend(self, user):

# Gets all unique songs for this user

user\_songs = self.get\_user\_items\_New\_data(user)

print("No. of unique songs for the user: %d" % len(user\_songs))

# Gets all unique items (songs) in the training data

all\_songs = self.get\_all\_items\_new\_data()

print("No. of unique songs in the training set: %d" % len(all\_songs))

def users\_for\_the\_given\_item(self,item):

# Gets all unique users (songs) in the training data

for i in item:

unique\_users = self.get\_item\_users\_New\_data(i)

l=Label(pd,text=i+" : %d People Listen Daily"%len(unique\_users))

l.pack(side=BOTTOM)

is\_model = song\_recommender()

is\_model.create(New\_data, 'user\_id', 'song')

#Print the songs for the user in New Data

user\_id = New\_users[n]

user\_items = is\_model.get\_user\_items\_New\_data(user\_id)

print("User-id: %s:" % user\_id)

print("--------------------------------------------------------------------------------")

#Using Tkinter Function for Displaying the User's Songs

v=Label(pd,text="Recommended Songs for the given user",font="Helvetica 16 bold italic")

v.pack()

q=Label(pd,text="------------------------------------------------------------------------------------")

q.pack()

scrollbar = Scrollbar(pd)

scrollbar.pack( side = RIGHT, fill = Y )

list1=Listbox(pd,width=60,yscrollcommand=scrollbar.set)

for i in range(len(user\_items)):

list1.insert(END,user\_items[i])

list1.pack()

scrollbar.config(command=list1.yview)

q=Label(pd,text="------------------------------------------------------------------------------------")

q.pack()

pop=[]

for i in range(0,5):

pop.append(random.choice(New\_songs))

u=Label(pd,text="Know More Songs",font="Helvetica 13 bold italic")

u.pack()

q=Label(pd,text="------------------------------------------------------------------------------------")

q.pack()

is\_model.users\_for\_the\_given\_item(pop)

#Recommend Songs for the user using Personal Songs played by the user

is\_model.recommend(user\_id)

print("--------------------------------------------------------------------------------")