from tkinter import messagebox

from tkinter import \*

from tkinter import simpledialog

import tkinter

from tkinter import filedialog

from tkinter.filedialog import askopenfilename

import pandas as pd #pandas to read and explore dataset

import numpy as np

from sklearn.preprocessing import LabelEncoder

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

from sklearn.preprocessing import StandardScaler

from sklearn\_extra.cluster import KMedoids

import matplotlib.pyplot as plt

import gower

from sklearn.metrics import silhouette\_score

from sklearn.decomposition import PCA

main = tkinter.Tk()

main.title("Shopping Hard or Hardly Shopping: Revealing Consumer Segments Using Clickstream Data") #designing main screen

main.geometry("1300x1200")

global filename, X, dataset, cluster, sc, centroids, labels, unique, count

def uploadDataset():

global filename, dataset

filename = filedialog.askopenfilename(initialdir="Dataset")

text.delete('1.0', END)

text.insert(END,filename+" Dataset Loaded\n\n")

dataset = pd.read\_csv(filename, sep=";", nrows=1000)

text.insert(END,str(dataset)+"\n\n")

text.insert(END,"Dataset Descriptive Analysis\n\n")

text.insert(END,str(dataset.describe()))

def Preprocessing():

global X, dataset, sc

text.delete('1.0', END)

dataset.fillna(0, inplace = True)

le = LabelEncoder()

dataset['page 2 (clothing model)'] = pd.Series(le.fit\_transform(dataset['page 2 (clothing model)'].astype(str)))#encode all str columns to numeric

X = dataset.values

sc = StandardScaler()

X = sc.fit\_transform(X)

text.insert(END,"Dataset Cleaning & Normalization Completed\n\n")

text.insert(END, str(X))

def runGower():

global X

text.delete('1.0', END)

X = gower.gower\_matrix(X)

text.insert(END,"Gower Distance Matrix Conversion Completed\n\n")

text.insert(END,"Gower Values = "+str(X))

def runKmedoid():

global X, centroids, unique, labels, count, dataset

text.delete('1.0', END)

pca = PCA(2)

X = pca.fit\_transform(X)

km = KMedoids(n\_clusters=6, method='pam')

km.fit(X)

labels = km.labels\_

dataset['cluster'] = labels

centroids = km.cluster\_centers\_

unique, count = np.unique(labels, return\_counts = True)

text.insert(END,"Kmedoid PAM Clustering Completed. Below are the cluster based segmented customers from click stream\n\n")

for i in range(len(unique)):

text.insert(END,"Cluster No : "+str(i)+" Total Segmented Customers : "+str(count[i])+"\n")

text.update\_idletasks()

score = []

cls = [2, 4, 6, 8, 10]

for i in range(len(cls)):

km = KMedoids(n\_clusters=cls[i], method='pam')

km.fit(X)

predict = km.labels\_

silhouette\_coefficients = silhouette\_score(X, predict)

score.append(silhouette\_coefficients)

plt.grid(True)

plt.xlabel('Number of Clusters')

plt.ylabel('Silhouette Score')

plt.plot(cls, score, 'ro-', color = 'green')

plt.legend(['Silhouette Score'], loc='upper left')

plt.title('Num Clusters Vs Silhouette Score Graph')

plt.show()

def calculateRevenue():

global dataset, unique, count

text.delete('1.0', END)

output = []

for i in range(len(unique)):

revenue = dataset.loc[dataset['cluster'] == unique[i]]

revenue = revenue['order'].sum() / count[i]

page\_visit = dataset.loc[dataset['cluster'] == unique[i]]

page\_visit = page\_visit['session ID'].sum() / count[i]

total\_page = dataset.loc[dataset['cluster'] == unique[i]]

total\_page = total\_page['page'].sum() / count[i]

output.append([unique[i], revenue, page\_visit, total\_page])

data = pd.DataFrame(output, columns=['Cluster No', 'Average Revenue/Orders', 'Average Page Visit', 'Average Total Pages Browse'])

text.insert(END, str(data))

def ExtensionSegment():

global dataset, unique, count, X, centroids, labels

plt.figure(figsize=(7, 7))

for cls in unique:

plt.scatter(X[labels == cls, 0], X[labels == cls, 1], label=cls)

plt.scatter(centroids[:,0],centroids[:,1],marker='x',s=169,linewidths=3,color='k',zorder=10)

plt.legend()

plt.title("Extension Customer Segmeentation via Visualization")

plt.show()

font = ('times', 16, 'bold')

title = Label(main, text='Shopping Hard or Hardly Shopping: Revealing Consumer Segments Using Clickstream Data')

title.config(bg='deep sky blue', fg='white')

title.config(font=font)

title.config(height=3, width=120)

title.place(x=0,y=5)

font1 = ('times', 12, 'bold')

text=Text(main,height=25,width=150)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=50,y=110)

text.config(font=font1)

font1 = ('times', 13, 'bold')

uploadButton = Button(main, text="Upload Click Stream Dataset", command=uploadDataset)

uploadButton.place(x=50,y=600)

uploadButton.config(font=font1)

processButton = Button(main, text="Preprocess Dataset", command=Preprocessing)

processButton.place(x=360,y=600)

processButton.config(font=font1)

gowerButton = Button(main, text="Convert Dataset to Gower Matrix", command=runGower)

gowerButton.place(x=680,y=600)

gowerButton.config(font=font1)

kmedoidButton = Button(main, text="Run PAM Based K-Medoid", command=runKmedoid)

kmedoidButton.place(x=50,y=650)

kmedoidButton.config(font=font1)

revenueButton = Button(main, text="Calculate Cluster Based Revenue", command=calculateRevenue)

revenueButton.place(x=360,y=650)

revenueButton.config(font=font1)

extensionButton = Button(main, text="Extension Customer Segment Graph", command=ExtensionSegment)

extensionButton.place(x=680,y=650)

extensionButton.config(font=font1)

main.config(bg='LightSteelBlue3')

main.mainloop()