**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

****

**ĐỒ ÁN HỌC PHẦN**

TÊN HỌC PHẦN: **Lập trình Python (PYPRO)**

MÃ SỐ LỚP HP: **IPPA233277\_03**

Tên chủ đề : **LẬP TRÌNH PYTHON PHÂN TÍCH DOANH SỐ BẢN LẺ TRỰC TUYẾN CỦA MỘT CÔNG TY ANH QUỐC (01/12/2010 – 09/12/2011)**

**Họ tên sinh viên: Cao Đăng Duy [08]**

**Mã số sinh viên: 21110397**

**Lớp: 211103 [CQ]**

**Ngày nộp: 17/05/2023**

**TP.HCM, ngày .... Tháng .... năm 2023**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

**ĐỀ TÀI ĐỒ ÁN HỌC PHẦN**

TÊN HỌC PHẦN: **Lập trình Python (PYPRO)**

MÃ SỐ LỚP HP: **IPPA233277\_03**

Tên đề tài: **LẬP TRÌNH PYTHON PHÂN TÍCH DOANH SỐ BẢN LẺ TRỰC TUYẾN CỦA MỘT CÔNG TY ANH QUỐC (01/12/2010 – 09/12/2011)**

**Giảng viên giảng dạy: VÕ XUÂN THỂ**

**Họ tên sinh viên: Cao Đăng Duy [08]**

**Mã số sinh viên: 21110397**

**Lớp: 211103 [CQ]**

**TÊN CÁC FILE SẢN PHẨM ĐỀ TÀI:**

**Tên sản phẩm đề tài: GxttHoTen\_DAHP.PyPro\_Topic.rar**

**Tập tin mã nguồn: G308CDD\_OnlineRetail.py**

**Tập dữ liệu thực nghiệm: G308CDD\_DAHP\_OnlineRetail.csv**

**Giới thiệu thư mục bài làm: G308CDD\_DAHP.Intro.docx**

**Tập các thư viện sử dụng:** # Speech: speech\_recognition, gtts, playsound

# GUI: tkinter

# EDA: pandas, numpy, scipy, sklearn

**NHẬN XÉT CỦA GIẢNG VIÊN**

**………………………………………………………..**

**………………………………………………………..**

**Ngày……./……../202…..**

**Giảng viên: Ký tên**

**TP.HCM, ngày 10 tháng 05 năm 2023**

**LỜI CẢM ƠN**

# DANH MỤC CÁC TỪ VIẾT TẮT

|  |  |
| --- | --- |
| CSDL hoặc DB | Cơ sở dữ liệu: DataBase |
| NSD = Users = Account | Người Sử Dụng = là quyền làm việc trên WebApp được đảm bảo bởi tối thiểu là 2 yếu tố: tên đăng nhập (UserName) và mật khẩu (Password) |
|  |  |
| AI | Artificial intelligence: Trí tuệ nhân tạo |
| EDA | Exploratory Data Analysis: Phân tích Khám phá Dữ liệu (Còn gọi là: phân tích dữ liệu thăm dò) |
| GUI | Graphical User Interface: Giao diện đồ hoạ |
| PYPRO | Lập trình Python |

**DANH MỤC CÁC THUẬT NGỮ**

|  |  |
| --- | --- |
| Giao diện người dùng:  User-Interface | Là hệ thống các màn hình giao tiếp cho phép người sử dụng tương tác với các thành phần phần mềm trong HTTT, thường là 1 trong các dong: Win-form, Web-form, Mobile-Form. |
| Tài khoản (Account) | Là một quyền làm việc trên HTTT được cấp phát cho một cá nhân thông qua tên tài khoản (username) và mật khẩu (password). |
| Trợ lý ảo: Voice Assistant | Là các hệ thống có khả năng “nghe” và “nói” với con người, nhờ đó hỗ trợ con người trong một số chức năng như một “trợ lý”. |

**MỤC LỤC**

[**LỜI CẢM ƠN** 3](#_Toc129347035)

[DANH MỤC CÁC TỪ VIẾT TẮT 4](#_Toc129347036)

[**DANH MỤC CÁC THUẬT NGỮ** 4](#_Toc129347037)

[Chương 1: GIỚI THIỆU VỀ ĐỒ ÁN HỌC PHẦN 8](#_Toc129347038)

[1.1. Tổng quan về ĐỒ ÁN HỌC PHẦN 8](#_Toc129347039)

[1.2. Nội dung chuyên môn chính của ĐỒ ÁN HỌC PHẦN 8](#_Toc129347040)

[1.3. Công cụ và nền tảng kỹ thuật thực hiện ĐỒ ÁN HỌC PHẦN 8](#_Toc129347041)

[1.4. Bố cục của báo cáo 9](#_Toc129347042)

[Chương 2: CƠ SỞ LÝ THUYẾT CỦA HỌC PHẦN 10](#_Toc129347043)

[2.1. GIỚI THIỆU VỀ LẬP TRÌNH PYTHON 10](#_Toc129347044)

[2.1.1. Giới thiệu chung 10](#_Toc129347045)

[2.1.2. Cài đặt “bộ thảo chương” (soạn thảo chương trình IDE) và “bộ dịch” 10](#_Toc129347046)

[2.1.3. Giới thiệu các bộ tương tác lập trình 10](#_Toc129347047)

[2.1.4. Cấu hình mô trường tương tác lập trình 10](#_Toc129347048)

[2.2. TỔNG QUAN VỀ NGÔN NGỮ LẬP TRÌNH PYTHON 10](#_Toc129347049)

[2.2.1. Danh hiệu (Identifier) 10](#_Toc129347050)

[2.2.2. Về viết mã lệnh lập trình (Code) 10](#_Toc129347051)

[2.2.3. Về chú giải (Comment) 10](#_Toc129347052)

[2.2.4. Các phép toán 10](#_Toc129347053)

[2.2.5. Chuỗi 10](#_Toc129347054)

[2.2.6. Lệnh 10](#_Toc129347055)

[2.2.7. Định nghĩa hàm 10](#_Toc129347056)

[2.2.8. Danh sách 10](#_Toc129347057)

[2.2.9. Bộ dữ liệu: Tuple 10](#_Toc129347058)

[2.2.10. Tập hợp: Set 10](#_Toc129347059)

[2.2.11. Từ điển: Dictionary 10](#_Toc129347060)

[2.2.12. Module 10](#_Toc129347061)

[2.2.13. Package 10](#_Toc129347062)

[2.2.14. Lớp 10](#_Toc129347063)

[2.2.15. Xử lý ngoại lệ 10](#_Toc129347064)

[2.3. GIỚI THIỆU CÁC THƯ VIỆN VÀ CÔNG CỤ CĂN BẢN 10](#_Toc129347065)

[2.3.1. Nạp thư viện 10](#_Toc129347066)

[2.3.2. Pandas 10](#_Toc129347067)

[2.3.3. Tensorflow 10](#_Toc129347068)

[2.3.4. NumPy 10](#_Toc129347069)

[2.3.5. SCIPY 10](#_Toc129347070)

[2.3.6. Theano 10](#_Toc129347071)

[2.3.7. Matplotlib 10](#_Toc129347072)

[2.3.8. scikit-learn = sklearn 10](#_Toc129347073)

[2.3.9. Keras 11](#_Toc129347074)

[2.3.10. NLTK 11](#_Toc129347075)

[2.3.11. PyTorch 11](#_Toc129347076)

[2.3.12. LightGBM 11](#_Toc129347077)

[2.3.13. Eli5 11](#_Toc129347078)

[2.4. LẬP TRÌNH PYTHON XỬ LÝ GIỌNG NÓI (Trợ lý ảo: Voice Assistant) 11](#_Toc129347079)

[2.4.1. Giới thiệu chủ đề 11](#_Toc129347080)

[2.4.2. Tài liệu và nguồn thực nghiệm 11](#_Toc129347081)

[2.4.3. Giới thiệu Các thư viện Python liên quan 11](#_Toc129347082)

[2.4.4. Giới thiệu bài toán 11](#_Toc129347083)

[2.4.5. Giới thiệu code tham khảo 11](#_Toc129347084)

[2.5. KỸ THUẬT LẬP TRÌNH ĐỆ QUY: THÁP HÀ NỘI (HANOI TOWER) 11](#_Toc129347085)

[2.6. LẬP TRÌNH PHÂN TÍCH KHÁM PHÁ (EDA) [Thăm dò] 11](#_Toc129347086)

[Chương 3: PHÂN TÍCH VÀ XÁC ĐINH CÁC CƠ SỞ KỸ THUẬT 12](#_Toc129347087)

[3.1. GIỚI THIỆU CHUNG VỀ CHỦ ĐỀ 12](#_Toc129347088)

[3.1.1. Mô tả yêu câu của chủ đề 12](#_Toc129347089)

[3.1.2. Phân tích yều cầu của chủ đề 12](#_Toc129347090)

[3.2. CÁC NỀN TẢNG KỸ THUẬT 12](#_Toc129347091)

[3.2.1. Hệ thống thư viện sử dụng 12](#_Toc129347092)

[3.2.2. Tập dữ liệu thực nghiệm 13](#_Toc129347093)

[3.2.3. Mô tả giải thuật 13](#_Toc129347094)

[3.3. LẬP TRÌNH XỬ GIỌNG NÓI (TRỢ LÝ ẢO: VOICE ASSISTANT) 14](#_Toc129347095)

[3.3.1. Giới thiệu chung về chủ đề 14](#_Toc129347096)

[3.3.1.1. Mô tả yêu cầu của chủ đề 14](#_Toc129347097)

[3.3.1.2. Phân tích yều cầu của chủ đề 14](#_Toc129347098)

[3.3.2. Các nền tảng kỹ thuật 15](#_Toc129347099)

[3.3.2.1. Nền tảng kỹ thuật cơ bản 15](#_Toc129347100)

[3.3.2.2. Hệ thống thư việc sử dụng 15](#_Toc129347101)

[3.3.3. Thiết kế giao diện 15](#_Toc129347102)

[3.3.4. Mã lệnh lập trình (python) 15](#_Toc129347103)

[Chương 4: LẬP TRÌNH PYTHON PHÂN TÍCH ........................................ 17](#_Toc129347104)

[4.1. Thiết kế giao diện 17](#_Toc129347105)

[4.2. Mã lệnh lập trình (python) 17](#_Toc129347106)

[Chương 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN CỦA ĐỒ ÁN HỌC PHẦN 21](#_Toc129347107)

[5.1. Kết luận 21](#_Toc129347108)

[5.1.1. Những kết quả đạt được 21](#_Toc129347109)

[5.1.2. Hạn chế 21](#_Toc129347110)

[5.2. Hướng phát triển 21](#_Toc129347111)

[5.2.1. Hướng khắc phục các hạn chế 21](#_Toc129347112)

[5.2.2. Hướng mở rộng ĐỒ ÁN HỌC PHẦN 21](#_Toc129347113)

[DANH MỤC TÀI LIỆU THAM KHẢO 22](#_Toc129347114)

# GIỚI THIỆU VỀ ĐỒ ÁN HỌC PHẦN

## Tổng quan về ĐỒ ÁN HỌC PHẦN

Đồ án Học phần Lập trình Python **LẬP TRÌNH PYTHON PHÂN TÍCH DOANH SỐ BẢN LẺ TRỰC TUYẾN CỦA MỘT CÔNG TY ANH QUỐC (01/12/2010 – 09/12/2011)** vận dụng kiến thức và kỹ năng về Lập trình Python để xây dựng giao diện đồ hoạ (GUI) cho phép: tương tác với người dùng bằng giọng nói (Trợ lý ảo: Voice Assistant) và phân tích thăm dò (EDA: Epolore Data Analysing) đối với tập dữ liệu thực nghiệm về …………………..

## Nội dung chuyên môn chính của ĐỒ ÁN HỌC PHẦN

Bao gồm 4 nội dung lập trình chính dùng ngôn ngữ lập trình Python:

+ Trợ lý ảo

+ Phân tích dữ liệu khám phá || thăm dò

+ Giao diện đồ họa dạng WinForm

+ Game

## Công cụ và nền tảng kỹ thuật thực hiện ĐỒ ÁN HỌC PHẦN

+ Ngôn ngữ lập trình **Python**

+ Công cụ lập trình và biên dịch: **Visual Studio Code**

+ Các thư viện chính:

import tkinter as cdd\_tk

from tkinter import messagebox

import numpy as np

import speech\_recognition as sr

from tkinter import ttk

from gtts import gTTS

import playsound

import os

import pandas as pd

from scipy import stats

from matplotlib.backends.backend\_tkagg import FigureCanvasTkAgg

from matplotlib.figure import Figure

from sklearn import preprocessing

from sklearn.feature\_selection import SelectKBest, chi2, f\_classif

from sklearn.preprocessing import Binarizer

import cv2

import tkinter as tk

from tkinter import \*

from tkinter import filedialog

from tkinter import messagebox as msg

import os

import subprocess

import plotly.graph\_objs as go

import plotly.io as pio

import matplotlib.pyplot as plt

import io

from sklearn.preprocessing import LabelEncoder

## Bố cục của báo cáo

Báo cáo gồm những nội dung như sau:

Chương 1: Giới thiệu đồ án của Học phần

Chương 2: Các cơ sở lý thuyết của Học phần phục việc thực hiện đề tài

Chương 3: Phân tích và xác đinh các cơ sở kỹ thuật thực hiện đề tài

Chương 4: Lập trình xây dựng sản phầm đề lài: **LẬP TRÌNH PYTHON PHÂN TÍCH DOANH SỐ BẢN LẺ TRỰC TUYẾN CỦA MỘT CÔNG TY ANH QUỐC (01/12/2010 – 09/12/2011)**

Chương 5 Tổng kết các kết quản đạt được và còn hạn chế của đồ án, đồng thời đề xuất hướng khắc phục hạn chế và phát triển Đồ án.

# CƠ SỞ LÝ THUYẾT CỦA HỌC PHẦN

## GIỚI THIỆU VỀ LẬP TRÌNH PYTHON

### Giới thiệu chung

### Cài đặt “bộ thảo chương” (soạn thảo chương trình IDE[[1]](#footnote-1)) và “bộ dịch”

### Giới thiệu các bộ tương tác lập trình

### Cấu hình mô trường tương tác lập trình

## TỔNG QUAN VỀ NGÔN NGỮ LẬP TRÌNH PYTHON

### Danh hiệu (Identifier)

### Về viết mã lệnh lập trình (Code)

### Về chú giải (Comment)

### Các phép toán

### Chuỗi

### Lệnh

### Định nghĩa hàm

### Danh sách

### Bộ dữ liệu: Tuple

### Tập hợp: Set

### Từ điển: Dictionary

### Module

### Package

### Lớp

### Xử lý ngoại lệ

## GIỚI THIỆU CÁC THƯ VIỆN VÀ CÔNG CỤ CĂN BẢN

### Nạp thư viện

### Pandas

### Tensorflow

### NumPy

### SCIPY

### Theano

### Matplotlib

### scikit-learn = sklearn

### Keras

### NLTK

### PyTorch

### LightGBM

### Eli5

## LẬP TRÌNH PYTHON XỬ LÝ GIỌNG NÓI (Trợ lý ảo: Voice Assistant)

### Giới thiệu chủ đề

### Tài liệu và nguồn thực nghiệm

### Giới thiệu Các thư viện Python liên quan

### Giới thiệu bài toán

### Giới thiệu code tham khảo

**[1] Nghe tiếng Việt => Text**

**[2] Text => đọc tiếng Việt**

## KỸ THUẬT LẬP TRÌNH ĐỆ QUY: THÁP HÀ NỘI (HANOI TOWER)

## LẬP TRÌNH PHÂN TÍCH KHÁM PHÁ (EDA) [Thăm dò]

1. **PHÂN TÍCH VÀ XÁC ĐINH CÁC CƠ SỞ KỸ THUẬT**

## GIỚI THIỆU CHUNG VỀ CHỦ ĐỀ

### Mô tả yêu câu của chủ đề

Vận dụng kỹ thuật lập trình EDA kết hợp với GUI và Voice Assistant cho phép dùng tập dữ liệu thực nghiệm về doanh số bán hàng của công ty để dự đoán và tư vấn các nhà quản lý hoạch định kế hoạch kinh doanh của công ty.

Dữ liệu thực nghiệm gồm doanh số bán hàng trong quá khứ của hệ thống siêu thị:

G308CDD\_DAHP\_OnlineRetail.csv : dạng file Excel .csv

. Số records: 541910

. Số thuộc tính input: 8 để đoán thuộc tính thứ ???: target variable:

= “RainTomorrow”[Y|N]

### Phân tích yều cầu của chủ đề

+ Chủ đề lập trình python này là cơ sở của các ứng dụng về DataWarehouse, Bigdata, Data Analysis, Data Sciences, . . .

+ Đây là xử lý nền tảng cho các nội dung chuyên môn (nêu trên) chủ yếu là giai đoạn tiền xử lý dữ liệu đầu vào (Input Preporcessing), như: xử lý NULL, lọc dữ liệu, tích hợp dữ liệu (Integration),…

**Một số trường hợp tiền xử lý dữ liệu đầu vào (input) của các hệ thống EDA**

[1] Xử lý NULL là ……………

[2] Xử lý dữ liệu quá khác biệt = “ngoại lệ” (isolated: ….) là……………

## CÁC NỀN TẢNG KỸ THUẬT

### Hệ thống thư viện sử dụng

**Bước 1: Nhập các thư viện cần thiết**

# AI/weatherAUS.py

# For linear algebra (Thư viện về Đại số tuyến tính tính)

import numpy as np

# For data processing (Thư viện xử lý dữ liệu)

import pandas as pd

### Tập dữ liệu thực nghiệm

**Bước 2: Tải tập dữ liệu**

#Load the data set (Nạp tập dữ liệu)

self.data = pd.read\_csv(

            ".\G308CDD\_DAHP\_OnlineRetail.csv", encoding="ISO-8859-1"

        )

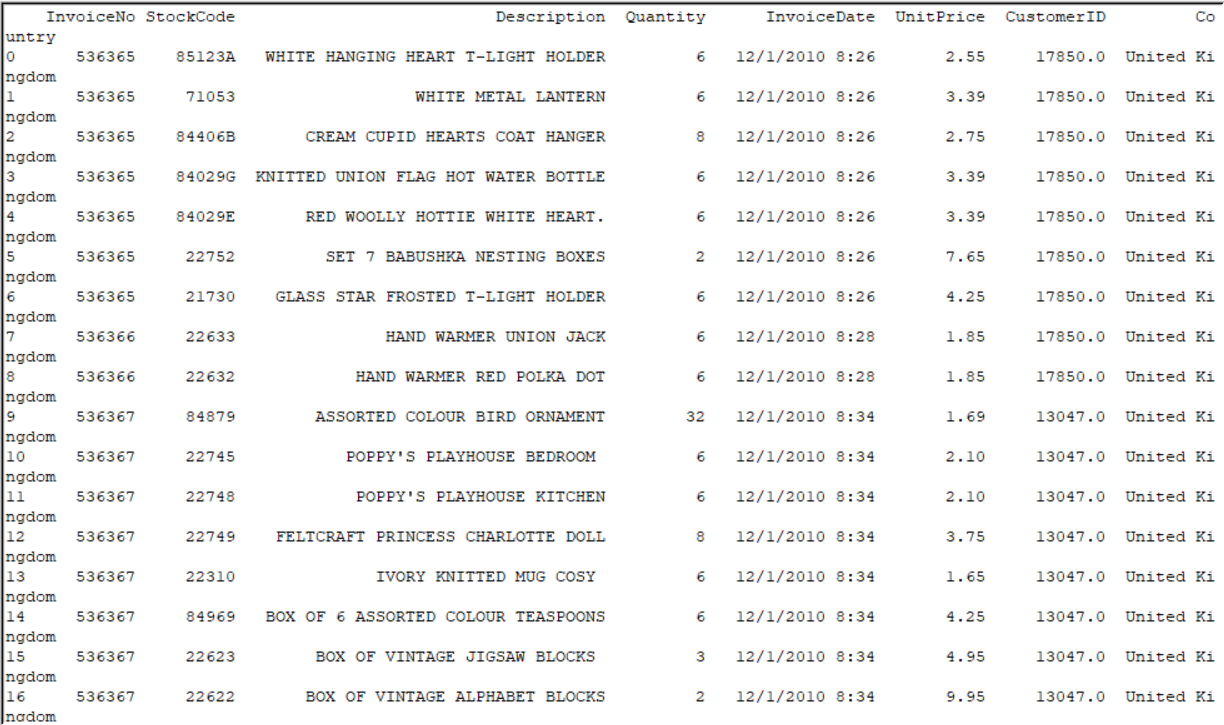
#Display the shape of the data set (Vẽ biểu đồ tập dữ liệu)

print('Độ lớn của bảng [frame] dữ liệu thời tiết:',df.shape)

#Display data (Hiển thị dữ liệu dạng mảng 5 dòng đầu)

print(df[0:5])

Chạy sẽ hiển thị kết quả:

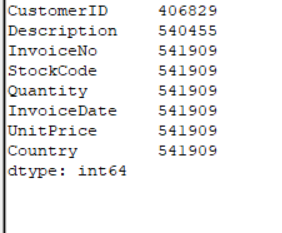


### Mô tả giải thuật

**Bước 3: Xử lý dữ liệu**

# Checking for null values (Kiểm tra giá trị null = đếm số dòng có dữ liệu ứng từng thuộc tính)

**Kết quả:**



## LẬP TRÌNH XỬ GIỌNG NÓI (TRỢ LÝ ẢO: VOICE ASSISTANT)

## Giới thiệu chung về chủ đề

### Mô tả yêu cầu của chủ đề

+ Lập trình App xử lý "lời nới" (Speech): Ứng dụng “Trợ lý ảo” (Voice Assistant)

+ Gồm 2 phần (Part) cơ bản sau

1. Nghe tiếng Việt => Text: ……………….

2. Trả lời bằng tiếng Việt : Text => Nói tiếng Việt: ……………….

### Phân tích yều cầu của chủ đề

+ Chủ đề lập trình python này là cơ sở của các ứng dụng về AI, ML, Deep ML.

+ Các hệ thống này thường có 3 phần :

Part1: Speech….: chính là part1 của chủ đề này

Part2: Các kỹ thuật AI

Part2: Voice….: chính là part2 của chủ đề này

## Các nền tảng kỹ thuật

### Nền tảng kỹ thuật cơ bản

### Hệ thống thư việc sử dụng

B1: Cài đặt thư viện

conda install …. Các thư viện sau:

------------------gốc hướng dẫn / dùng python trong visual (pip3)

pip3 install SpeechRecognition

*pip3 install SpeechRecognition pydub # Long audio source*

pip3 install pyaudio

B2 : Nạp thư viện vào App

1. import tkinter as cdd\_tk
2. from tkinter import messagebox
3. import numpy as np
4. import speech\_recognition as sr
5. from tkinter import ttk
6. from gtts import gTTS
7. import playsound
8. import os
9. import pandas as pd
10. from scipy import stats
11. from matplotlib.backends.backend\_tkagg import FigureCanvasTkAgg
12. from matplotlib.figure import Figure
13. from sklearn import preprocessing
14. from sklearn.feature\_selection import SelectKBest, chi2, f\_classif
15. from sklearn.preprocessing import Binarizer
16. import cv2
17. import tkinter as tk
18. from tkinter import \*
19. from tkinter import filedialog
20. from tkinter import messagebox as msg
21. import os
22. import subprocess
23. import plotly.graph\_objs as go
24. import plotly.offline as pyoff
25. import plotly.io as pio
26. import matplotlib.pyplot as plt
27. import io
28. import plotly.express as px

## Thiết kế giao diện

+ Xử lý trong cửa sổ Console Output của Spyder

## Mã lệnh lập trình (python)

**FULL CODE**

# -\*- coding: utf-8 -\*-

"""

Created on Sat Jul 16 19:59:48 2023

@author: VOXUAN

"""

# B1: NẠP THƯ VIỆN

import speech\_recognition as sr

from gtts import gTTS

import playsound

# B2: CHỌN PHƯƠNG ÁN NHẬP ÂM THANH TỪ MICROPHONE

r = sr.Recognizer()

# B3: XỬ LÝ NHẬN DIỆN MIC

#PHẦN 1: IN RA TEXT THEO LỜI NÓI = Nghe tiếng Việt => Text

with sr.Microphone() as Source:

#hiệu chỉnh mic để chuẩn bị nói

print("Hieu chinh nhieu trươc khi noi!")

r.adjust\_for\_ambient\_noise(Source, duration=1)

#nhận lời nói của người dùng từ MIc mặc định lưu dữ liệu âm thanh vào audio\_data

print("Nói tiếng Việt đi, sau 5s sẽ in ra văn bản!")

audio\_data = r.record(Source, duration = 5)

#In ra văn bản text

print("KẾT QUẢ NHẬN DIỆN ..................")

#chuyển lời nói thành văn bản

try:

text = r.recognize\_google(audio\_data,language="vi")

except:

text = "Quý vị nói gì nghe không rõ...!"

#in kết quả ra

print("Quý vị đã nói là : ",format(text))

#PHẦN 2: XUẤT RA LỜI NÓI THEO VĂN BẢN ĐÃ NHẬP = Trả lời bằng tiếng Việt : Text => Nói tiếng Việt

# ĐỌC CHỮ VIỆT => NÓI RA TIẾNG VIỆT

def VXT(t):

vx = gTTS(text=t, lang = 'vi')

xuan = '06VXT.mp3'

vx.save(xuan)

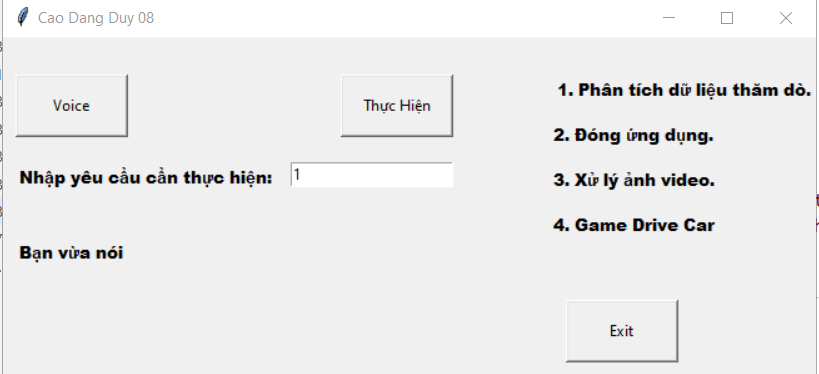
playsound.playsound(xuan)

VXT(text)

#VXT("số thứ tự 06, họ tên: VÕ XUÂN THỂ là giảng viên viên của TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT THÀNH PHỐ HỒ CHÍ MINH")

1. **LẬP TRÌNH PYTHON PHÂN TÍCH DOANH SỐ BẢN LẺ TRỰC TUYẾN**

## Thiết kế giao diện



## Mã lệnh lập trình (python)

import tkinter as cdd\_tk

from tkinter import messagebox

import numpy as np

import speech\_recognition as sr

from tkinter import ttk

from gtts import gTTS

import playsound

import os

import pandas as pd

from scipy import stats

from matplotlib.backends.backend\_tkagg import FigureCanvasTkAgg

from matplotlib.figure import Figure

from sklearn import preprocessing

from sklearn.feature\_selection import SelectKBest, chi2, f\_classif

from sklearn.preprocessing import Binarizer

import cv2

import tkinter as tk

from tkinter import \*

from tkinter import filedialog

from tkinter import messagebox as msg

import os

import subprocess

import plotly.graph\_objs as go

import plotly.io as pio

import matplotlib.pyplot as plt

import io

from sklearn.preprocessing import LabelEncoder

class Window(cdd\_tk.Tk):

    def \_\_init\_\_(self):

        super().\_\_init\_\_()

        self.CDD\_FILE = os.path.abspath("") + "\CDD.mp3"

        self.title("Cao Dang Duy 08")

        self.geometry("650x270")

        self.radio = cdd\_tk.IntVar()

        self.btnExit = cdd\_tk.Button(self, text="Exit", command=self.on\_closing)

        self.btnExit.place(x=450, y=210, w=90, h=50)

        self.lb1 = cdd\_tk.Label(

            self, text="Nhập yêu cầu cần thực hiện: ", font=("Arial Black", 10)

        )

        self.lb1.place(x=10, y=100)

        self.txtIn = cdd\_tk.Entry(self)

        self.txtIn.place(x=230, y=100, w=130, h=20)

        self.button = cdd\_tk.Button(self, text="Voice", command=self.new\_window\_voice)

        self.button.place(x=10, y=30, width=90, height=50)

        self.btnEDA = cdd\_tk.Button(self, text="Thực Hiện", command=self.ThucHien\_text)

        self.btnEDA.place(x=270, y=30, w=90, h=50)

        self.lbVoice = cdd\_tk.Label(self, text="Bạn vừa nói", font=("Arial Black", 10))

        self.lbVoice.place(x=10, y=160)

        self.lbtxt = cdd\_tk.Label(

            self,

            text="\t  1. Phân tích dữ liệu thăm dò.\n\n2. Đóng ứng dụng.\n\n3. Xử lý ảnh video.\n\n4. Game Drive Car",

            font=("Arial Black", 10),

        )

        self.lbtxt.place(x=360, y=30)

        self.dict = {

            1: self.new\_window\_panel,

            2: self.on\_closing,

            3: self.Cut\_frame,

            4: self.game,

        }

        self.number = 0

        self.data = None

        self.dropcolum = None

    def game(self):

        subprocess.run(["python", "G3B14EX7\_Game\_08CaoDangDuy.py"])

    def set\_up(self):

        self.child = child\_window(self)

        self.child.mainloop()

    def new\_window\_voice(self):

        form1 = cdd\_tk.Toplevel()

        form1.title(

            "08 CAO ĐĂNG DUY, 211103C\_HCMUTE, ĐỒ ÁN HỌC PHẦN: LẬP TRÌNH PYTHON, T5.2023"

        )

        form1.geometry("400x400")

        form1.resizable(cdd\_tk.FALSE, cdd\_tk.FALSE)

        label1 = cdd\_tk.Label(

            form1,

            text="Chọn ngôn ngữ muốn nói: ",

            relief=cdd\_tk.SUNKEN,

            font=("Arial Black", 10),

            borderwidth=3,

            width=25,

            height=2,

            anchor="center",

        )

        radio1 = cdd\_tk.Radiobutton(

            form1,

            text="Tiếng Anh",

            variable=self.radio,

            value=1,

            font=("Arial Black", 14),

        )

        radio2 = cdd\_tk.Radiobutton(

            form1,

            text="Tiếng Việt",

            variable=self.radio,

            value=2,

            font=("Arial Black", 14),

        )

        btnnoi = cdd\_tk.Button(form1, text="Bắt đầu nói:", command=self.Noi)

        btnphat = cdd\_tk.Button(form1, text="Phát lại", command=self.Phat)

        label1.place(x=95, y=70)

        radio1.place(x=120, y=130)

        radio2.place(x=120, y=160)

        btnnoi.place(x=150, y=230)

        btnphat.place(x=162, y=270)

    def new\_window\_panel(self):

        self.data = pd.read\_csv(

            ".\G308CDD\_DAHP\_OnlineRetail.csv", encoding="ISO-8859-1"

        )

        self.data.head(1000)

        panel = new\_panel(self)

        panel.mainloop()

    def Noi(self):

        rCDD = sr.Recognizer()

        with sr.Microphone() as Source:

            if self.radio.get() == 1:

                lang = "en"

            elif self.radio.get() == 2:

                lang = "vi"

            messagebox.showinfo("Thông báo", "Bấm OK để bắt đầu, trong 5s")

            file\_audio\_data = rCDD.record(Source, duration=5)

            try:

                vCDDlenh = rCDD.recognize\_google(file\_audio\_data, language=lang)

                if vCDDlenh == "hài" or vCDDlenh == "hi" or vCDDlenh == "ai":

                    self.on\_closing

                if vCDDlenh == "một" or vCDDlenh == "mốt":

                    self.new\_window\_panel

                self.number = self.convert(vCDDlenh)

                self.lbVoice.config(text=f"Bạn vừa nói là: {self.number}")

            except:

                self.lbVoice.config(text="Bạn nói gì tôi nghe không rõ....!")

            if os.path.isfile(f"G337CDD.mp3"):

                os.remove(f"G337CDD.mp3")

            self.CDD\_FILE = "G337CDD.mp3"

            CDD\_Text = gTTS(text=vCDDlenh, lang=lang)

            CDD\_Text.save(self.CDD\_FILE)

            self.Phat()

            self.ThucThi()

    def Phat(self):

        playsound.playsound(self.CDD\_FILE)

    def eda\_click(self):

        result = self.txtIn.get("1.0", cdd\_tk.END)

        if result == "":

            result = self.lbVoice.cget("text")

        self.lbEDA.config(text=result)

        self.txtIn.delete("1.0", cdd\_tk.END)

    def on\_closing(self):

        if cdd\_tk.messagebox.askokcancel("Quit", "Do you want to quit?"):

            self.destroy()

    def convert(self, text):

        t = text.upper()

        if t == "1":

            number = 1

        elif t == "2":

            number = 2

        elif t == "3":

            number = 3

        elif t == "4":

            number = 4

        elif t == "5":

            number = 5

        elif t == "6":

            number = 6

        return number

    def Cut\_frame(self):

        cutframe = Cut\_Frame(self)

        cutframe.mainloop()

    def ThucThi(self):

        self.dict[self.number]()

    def ThucHien\_text(self):

        self.number = self.convert(self.txtIn.get())

        self.dict[self.number]()

class child\_window(cdd\_tk.Toplevel):

    def \_\_init\_\_(self, parent):

        super().\_\_init\_\_(parent)

        self.title("XỬ LÝ  & CHỌN CÁC GIÁ TRỊ CẦN XỬ LÝ")

        self.geometry("800x500")

        self.resizable(cdd\_tk.FALSE, cdd\_tk.FALSE)

        cdd\_tk.Label(self, text="Các thuộc tính").place(x=15, y=15)

        self.txtSource = cdd\_tk.Entry(self, width=30)  # Entry = cho nhập DL vào

        self.txtSource.place(x=120, y=15)

        self.btnAdd = cdd\_tk.Button(self, text="Add", width=10, command=self.insert)

        self.btnAdd.place(x=320, y=10)

        self.direction\_var = cdd\_tk.StringVar(value="Thuận")

        self.forward\_radio = cdd\_tk.Radiobutton(

            self, text="Thuận", variable=self.direction\_var, value="Thuận"

        )

        self.backward\_radio = cdd\_tk.Radiobutton(

            self, text="Nghịch", variable=self.direction\_var, value="Nghịch"

        )

        self.forward\_radio.place(x=400, y=10)

        self.backward\_radio.place(x=500, y=10)

        self.listbox1 = cdd\_tk.Listbox(

            self, height=25, width=40, font="Consolas 8", selectmode=cdd\_tk.EXTENDED

        )

        self.listbox1.bind(

            "<Button-3>", self.ShowPopupMenuA

        )  # <Button-3> : đăng ký sự kiện cho chuột phải của

        self.listbox1.place(x=15, y=50)

        self.listbox2 = cdd\_tk.Listbox(

            self, height=25, width=40, font="Consolas 8", selectmode=cdd\_tk.EXTENDED

        )

        self.listbox2.place(x=450, y=50)

        self.listbox2.bind("<Button-3>", self.ShowPopupMenuB)

        cdd\_tk.Label(self, text="Số lượng").place(x=15, y=420)

        self.lblSoLuong = cdd\_tk.Label(

            self,

            relief=cdd\_tk.SUNKEN,

            font="Times 8",

            borderwidth=3,

            width=15,

            height=1,

        )

        self.lblSoLuong.place(x=100, y=420)

        self.btn1 = cdd\_tk.Button(

            self, text="Copy to left", width=10, command=self.CopyToLeft

        )

        self.btn2 = cdd\_tk.Button(

            self, text="Copy to right", width=10, command=self.CopyToRight

        )

        self.btn3 = cdd\_tk.Button(

            self, text="Move to left", width=10, command=self.MoveToLeft

        )

        self.btn4 = cdd\_tk.Button(

            self, text="Move to right", width=10, command=self.MoveToRight

        )

        self.btn5 = cdd\_tk.Button(self, text="Delete", width=10, command=self.Delete)

        self.btn1.place(x=320, y=70)

        self.btn2.place(x=320, y=110)

        self.btn3.place(x=320, y=150)

        self.btn4.place(x=320, y=190)

        self.btn5.place(x=320, y=230)

        self.btn\_dropcolum = cdd\_tk.Button(

            self, text="Xác nhận", command=lambda: self.valueWasChose(parent)

        )

        self.btn\_dropcolum.place(x=700, y=420)

        self.data = parent.data.columns.tolist()

    def on\_closing(self):

        if cdd\_tk.messagebox.askokcancel(

            "Đóng chương trình", "Bạn chắc chắn muốn thoát?"

        ):

            self.destroy()

    def insert(self):

        for dt in self.data:

            self.listbox1.insert(cdd\_tk.END, dt)

    def total(self):

        dem = 0

        a = self.txtSource.get().strip()

        kq = a.ljust(20)

        if a != "":

            self.listbox1.insert(cdd\_tk.END, kq)

        # Đếm số dòng trong listbox

        dem = self.listbox1.size()

        # Điền thông tin vào label

        self.lblSoLuong.configure(text=dem)

        self.txtSource.delete(0, cdd\_tk.END)

    def valueWasChose(self, parent):

        all\_items = []

        for i in range(self.listbox2.size()):

            all\_items.append(self.listbox2.get(i))

        parent.dropcolumn = all\_items

        # print(parent.dropcolumn)

    def CopyToRight(self):

        direction = self.direction\_var.get()

        if direction == "Thuận":

            selection = self.listbox1.curselection()

            selection = sorted(selection)  # sắp xếp theo thứ tự tăng dần

            self.listbox2.insert(cdd\_tk.END, \*[self.listbox1.get(i) for i in selection])

        else:

            selection = self.listbox1.curselection()

            selection = sorted(selection, reverse=True)  # sắp xếp theo thứ tự giảm dần

            self.listbox2.insert(0, \*[self.listbox1.get(i) for i in selection])

        self.listbox1.select\_clear(0, cdd\_tk.END)  # b

        self.total()

    def MoveToRight(self):

        direction = self.direction\_var.get()

        selection = self.listbox1.curselection()

        selection = sorted(selection, reverse=(direction == "Nghịch"))

        if selection:

            values = [self.listbox1.get(i) for i in selection]

            self.listbox2.insert(cdd\_tk.END, \*values)

        if direction == "Thuận":

            for i in reversed(selection):

                self.listbox1.delete(i)

        else:

            for i in selection:

                self.listbox1.delete(i)

        self.listbox1.selection\_clear(0, cdd\_tk.END)

        self.total()

    def Delete(self):

        selection = self.listbox1.curselection()

        if len(selection) > 0:

            for i in reversed(selection):

                self.listbox1.delete(i)

        selection2 = self.listbox2.curselection()

        if len(selection2) > 0:

            for i in reversed(selection2):

                self.listbox2.delete(i)

        self.total()

    def CopyToLeft(self):

        direction = self.direction\_var.get()

        if direction == "Thuận":

            selection = self.listbox2.curselection()

            selection = sorted(selection)

            if len(selection) > 0:

                values = [self.listbox2.get(i) for i in selection]

                self.listbox1.insert(cdd\_tk.END, \*values)

        else:

            selection = self.listbox2.curselection()

            selection = sorted(selection, reverse=True)

            if len(selection) > 0:

                values = [self.listbox2.get(i) for i in selection]

                self.listbox1.insert(cdd\_tk.END, \*values)

        self.total()

    def MoveToLeft(self):

        direction = self.direction\_var.get()

        selection = self.listbox2.curselection()

        selection = sorted(selection, reverse=(direction == "Nghịch"))

        if selection:

            values = [self.listbox2.get(i) for i in selection]

            self.listbox1.insert(cdd\_tk.END, \*values)

            if direction == "Thuận":

                for i in reversed(selection):

                    self.listbox2.delete(i)

            else:

                for i in selection:

                    self.listbox2.delete(i)

        self.listbox2.selection\_clear(0, cdd\_tk.END)

        self.total()

    def ShowPopupMenuA(self, e):

        if self.listbox1.size() > 0:

            popMenu = cdd\_tk.Menu(self.listbox1, tearoff=cdd\_tk.FALSE)

            popMenu.add\_command(label="Copy To Right", command=self.CopyToRight)

            popMenu.add\_command(label="Move To Right", command=self.MoveToRight)

            popMenu.add\_command(label="Delete", command=self.Delete)

            popMenu.tk\_popup(

                e.x\_root, e.y\_root

            )  # phải thiết lập x\_root, y\_root để showpopup

    def ShowPopupMenuB(self, e):

        if self.listbox1.size() > 0:

            popMenu = cdd\_tk.Menu(self.listbox2, tearoff=cdd\_tk.FALSE)

            popMenu.add\_command(label="Copy To Left", command=self.CopyToLeft)

            popMenu.add\_command(label="Move To Left", command=self.MoveToLeft)

            popMenu.add\_command(label="Delete", command=self.Delete)

            popMenu.tk\_popup(

                e.x\_root, e.y\_root

            )  # phải thiết lập x\_root, y\_root để showpopup

class new\_panel(cdd\_tk.Toplevel):

    def \_\_init\_\_(self, parent):

        super().\_\_init\_\_(parent)

        self.title("EDA")

        self.geometry("1200x700")

        self.resizable(cdd\_tk.FALSE, cdd\_tk.FALSE)

        self.data = pd.read\_csv(

            ".\G308CDD\_DAHP\_OnlineRetail.csv", encoding="ISO-8859-1"

        )

        parent.data = self.data

        self.panel = cdd\_tk.Frame(self, bg="white", bd=2, relief=cdd\_tk.SUNKEN)

        self.panel.place(x=10, y=10, w=980, h=580)

        self.btn = cdd\_tk.Button(

            self, text="Drop Column", command=lambda: self.EAD\_column(parent), width=10

        )

        self.btn.place(x=1052, y=20)

        self.label = cdd\_tk.Label(self, text="Chọn số cột lượng cột NULL:")

        self.label.place(x=1030, y=60)

        self.btn\_data = cdd\_tk.Button(

            self, text="Thực hiện", command=lambda: self.EDA\_data(parent), width=10

        )

        self.btn\_data.place(x=1052, y=550)

        self.text = cdd\_tk.Text(self.panel)

        self.text.pack(fill=cdd\_tk.BOTH, expand=True)

        self.flag = 0

        self.spin = cdd\_tk.Spinbox(self, from\_=1, to=8, width=10)

        self.spin.place(x=1052, y=100)

        self.btn\_row = cdd\_tk.Button(

            self, text="Drop Row Null", command=lambda: self.EAD\_row(parent), width=11

        )

        self.btn\_row.place(x=1052, y=140)

        self.btn\_dothi = cdd\_tk.Button(

            self, text="Vẽ z-score", command=self.EDA\_vedothi, width=10

        )

        self.btn\_dothi.place(x=1052, y=180)

        self.btn\_thaythe = cdd\_tk.Button(

            self, text="Thay thế", command=lambda: self.EDA\_thaythe(parent), width=10

        )

        self.btn\_thaythe.place(x=1052, y=220)

        self.btn\_roirac = cdd\_tk.Button(

            self, text="Rời rạc", command=self.EDA\_roiRac, width=10

        )

        self.btn\_roirac.place(x=1052, y=260)

        self.method\_label = cdd\_tk.Label(self, text="Select method:", width=10)

        self.method\_label.place(x=1052, y=300)

        method\_list = ["Chi2", "f\_classif"]

        self.method = ttk.Combobox(self, values=method\_list, state="readonly", width=10)

        self.method.place(x=1052, y=340)

        self.spin2 = cdd\_tk.Spinbox(self, from\_=1, to=8, width=10)

        self.spin2.place(x=1052, y=380)

        self.btn\_phantich = cdd\_tk.Button(

            self, text="PT Thăm dò", command=self.EDA\_Thamdo, width=10

        )

        self.btn\_phantich.place(x=1052, y=420)

        self.method.current(0)

        self.btn\_dauvao = cdd\_tk.Button(

            self, text="Lấy đầu vào", command=lambda: self.EDA\_DauVao(parent), width=10

        )

        self.btn\_dauvao.place(x=1052, y=460)

        self.btn\_csv = cdd\_tk.Button(self, text="Show", width=10, command=self.show\_csv)

        self.btn\_csv.place(x=10, y=600)

        self.btn\_reset = cdd\_tk.Button(self, text="Reset", command=self.reset, width=10)

        self.btn\_reset.place(x=120, y=600)

        self.btn\_vedothi = cdd\_tk.Button(

            self, text="Vẽ Đồ thị", width=10, command=lambda: self.VeDoThi(parent)

        )

        self.btn\_vedothi.place(x=230, y=600)

    def show\_csv(self):

        df = self.data.head(1000)

        self.text.delete("1.0", cdd\_tk.END)

        self.text.insert(cdd\_tk.END, df.to\_string())

        self.panel.update()

    def reset(self):

        self.text.delete("1.0", cdd\_tk.END)

        self.text.destroy()

        self.text = cdd\_tk.Text(self.panel)

        self.text.pack(fill=cdd\_tk.BOTH, expand=True)

    def VeDoThi(self, parent):

        form = Do\_Thi\_CDD(parent)

        form.mainloop()

    def EDA\_Thamdo(self):

        text = self.method.get()

        if text == "Chi2":

            self.flag = 6

        else:

            self.flag = 7

        self.k = int(self.spin2.get())

    def EAD\_row(self, parent):

        self.flag = 2

        child = child\_window(parent)

        child.mainloop()

    def EAD\_column(self, parent):

        self.flag = 1

        child = child\_window(parent)

        child.mainloop()

    def EDA\_vedothi(self):

        self.flag = 3

    def EDA\_thaythe(self, parent):

        self.flag = 4

        child = child\_window(parent)

        child.mainloop()

    def EDA\_roiRac(self):

        self.flag = 5

    def Lay\_zcore(self, text):

        self.z = int(text.get())

        print(self.z)

    def EDA\_DauVao(self, parent):

        self.flag = 8

        child = child\_window(parent)

        child.mainloop()

    def EDA\_data(self, parent):

        if self.flag == 0:

            self.text.insert(

                cdd\_tk.END, "Kich thuoc cua DataFrame: {}\n".format(self.data.shape)

            )

            self.text.insert(cdd\_tk.END, f"{self.data.head(5)}\n")

            self.text.insert(cdd\_tk.END, str(self.data.count().sort\_values()) + "\n")

        if self.flag == 1:

            self.data = self.data.drop(columns=parent.dropcolumn, axis=1)

            self.text.insert(

                cdd\_tk.END,

                "Kich thuoc cua DataFrame sau khi drop cot duoc chon: {}\n".format(

                    self.data.shape

                ),

            )

            self.text.insert(cdd\_tk.END, str(self.data.count().sort\_values()) + "\n")

        if self.flag == 2:

            # value = int(self.spin.get())

            # self.data = self.data.dropna(thresh=value)

            self.data = self.data.dropna(subset=parent.dropcolumn)

            self.text.insert(

                cdd\_tk.END,

                "Kich thuoc cua DataFrame sau khi drop cac dong Null theo cac cot da chon: {}\n".format(

                    self.data.shape

                ),

            )

            self.text.insert(cdd\_tk.END, str(self.data.count().sort\_values()) + "\n")

            self.text.insert(cdd\_tk.END, "Tính toán Z core \n")

            z = np.abs(stats.zscore(self.data.\_get\_numeric\_data()))

            self.zcore = z

            self.text.insert(cdd\_tk.END, "MA TRAN Z-SCORE:\n")

            self.text.insert(cdd\_tk.END, str(z) + "\n")

        if self.flag == 3:

            # Tính toán z-score cho các cột "Quantity" và "UnitPrice"

            z\_scores = np.abs(stats.zscore(self.data[["Quantity", "UnitPrice"]]))

            # Vẽ biểu đồ z-score

            fig, ax = plt.subplots()

            ax.boxplot(z\_scores, labels=["Quantity", "UnitPrice"])

            ax.set\_ylabel("Z-Score")

            ax.set\_title("Biểu đồ Z-Score trước khi xử lý ngoại lệ")

            plt.show()

            # Lọc dữ liệu theo điều kiện Z-Score

            self.data = self.data[(z\_scores < 3).all(axis=1)]

            z\_scores = np.abs(stats.zscore(self.data[["Quantity", "UnitPrice"]]))

            # Vẽ biểu đồ z-score

            fig, ax = plt.subplots()

            ax.boxplot(z\_scores, labels=["Quantity", "UnitPrice"])

            ax.set\_ylabel("Z-Score")

            ax.set\_title("Biểu đồ Z-Score sau khi xử lý ngoại lệ")

            plt.show()

            self.text.insert(

                cdd\_tk.END,

                "Kich thuoc cua DataFrame sau khi xu li cac gia tri ngoai le: {}\n".format(

                    self.data.shape

                ),

            )

        if self.flag == 4:

            mean\_value = self.data[parent.dropcolumn].mean()

            self.data[parent.dropcolumn] = self.data[parent.dropcolumn].fillna(

                mean\_value

            )

            self.text.insert(

                cdd\_tk.END,

                "Kich thuoc cua DataFrame sau khi thay the cac dong null bang gia tri trung binh: {}\n".format(

                    self.data.shape

                ),

            )

            self.text.insert(cdd\_tk.END, str(self.data.count().sort\_values()) + "\n")

        if self.flag == 5:

            self.data = self.data.drop(

                [

                    "Description",

                    "Country",

                    "InvoiceDate",

                    "CustomerID",

                    "StockCode",

                    "InvoiceNo",

                ],

                axis=1,

            )

            self.text.insert(cdd\_tk.END, "Chuẩn hóa rời rạc \n")

            rr = preprocessing.MinMaxScaler()  # xác định thang đo rr.fit(df)

            rr.fit(self.data)

            self.data = pd.DataFrame(

                rr.transform(self.data),

                index=self.data.index,

                columns=self.data.columns,

            )

            self.text.insert(cdd\_tk.END, "\n")

            self.text.insert(cdd\_tk.END, self.data.iloc[4:10])

            self.text.insert(cdd\_tk.END, "\n")

            self.text.insert(

                cdd\_tk.END,

                f"Độ lớn của bảng [frame] dữ liệu SAU KHI CHUẨN HÓA DL: {self.data.shape}\n",

            )

        if self.flag == 6:

            a = self.data.loc[:, self.data.columns != "Quantity"]

            b = self.data[["Quantity"]]

            binarizer = Binarizer(threshold=0.5)

            b = binarizer.fit\_transform(b)

            selector = SelectKBest(chi2, k=self.k)

            selector.fit(a, b)

            a\_new = selector.transform(a)

            selected\_columns = a.columns[selector.get\_support(indices=True)]

            self.text.insert("end", "Các đặc trưng quan trọng: \n")

            for col in selected\_columns:

                self.text.insert("end", "-{}\n".format(col))

            self.text.insert(cdd\_tk.END, str(a\_new))

            self.text.insert(cdd\_tk.END, "\n")

            self.data = self.data[selected\_columns]

            parent.data = self.data

        if self.flag == 7:

            a = self.data.loc[:, self.data.columns != "Quantity"]

            b = self.data["Quantity"].ravel()

            selector = SelectKBest(f\_classif, k=self.k)

            selector.fit(a, b)

            a\_new = selector.transform(a)

            selected\_columns = a.columns[selector.get\_support(indices=True)]

            self.text.insert("end", "Các đặc trưng quan trọng: \n")

            for col in selected\_columns:

                self.text.insert("end", "-{}\n".format(col))

            self.text.insert(cdd\_tk.END, str(a\_new))

            self.text.insert(cdd\_tk.END, "\n")

            self.data = self.data[selected\_columns]

            parent.data = self.data

        if self.flag == 8:

            self.text.insert("end", "Các thuộc tính đầu vào: \n")

            self.text.insert(cdd\_tk.END, f"{str(parent.dropcolumn)} \n")

            for col in parent.dropcolumn:

                self.text.insert("end", "-{}\n".format(self.data[[col]]))

class Cut\_Frame(cdd\_tk.Toplevel):

    def \_\_init\_\_(self, parent):

        super().\_\_init\_\_(parent)

        self.title("CDD Cut Frame")

        self.geometry("380x350")

        self.resizable(cdd\_tk.FALSE, cdd\_tk.FALSE)

        self.buttonFolder = cdd\_tk.Button(

            self, text="open", width=10, command=self.open\_file

        )

        self.buttonFolder.place(x=140, y=20)

        self.lblFolder = cdd\_tk.Label(

            self, text="Mở Folder", width=10, font="Arial 13", anchor=cdd\_tk.CENTER

        )

        self.lblFolder.place(x=10, y=20)

        self.fileName = None

        self.txtFolderName = cdd\_tk.Label(self, width=30, text="", relief=SUNKEN)

        self.txtFolderName.place(x=140, y=70)

        self.LableName = cdd\_tk.Label(

            self, width=10, text="Ten folder", font="Arial 13", anchor=cdd\_tk.CENTER

        )

        self.LableName.place(x=10, y=70)

        self.LableCatFrame = cdd\_tk.Label(

            self,

            text="Cắt khi video đang chạy: ",

            font="Arial 13",

        )

        self.LableCatFrame.place(x=15, y=120)

        self.btnXulyAnh = cdd\_tk.Button(

            self, text="Xử lý", width=10, command=self.capture\_frames

        )

        self.btnXulyAnh.place(x=240, y=120)

        self.btnCatYMuon = cdd\_tk.Button(

            self, text="Cắt ảnh", width=10, command=self.cut\_frame

        )

        self.LableCatFrameTime = cdd\_tk.Label(

            self,

            text="Cắt khi video tại thời điểm: ",

            font="Arial 13",

        )

        self.LableCatFrameTime.place(x=15, y=250)

        self.btnCatYMuon.place(x=110, y=300)

        self.spinbox = cdd\_tk.Spinbox(self, from\_=0, to=100, width=5)

        self.spinbox.place(x=20, y=303)

        self.lablespin = cdd\_tk.Label(self, text="giây")

        self.lablespin.place(x=70, y=300)

        self.entr = cdd\_tk.Entry(self, width=10, relief=cdd\_tk.SUNKEN)

        self.btnfd = cdd\_tk.Button(

            self, text="Chọn đường dẫn", command=self.Chon\_duong\_Dan

        )

        self.labelfd = cdd\_tk.Label(self, text="Nhập tên")

        self.labelfd.place(x=10, y=180)

        self.entr.place(x=70, y=180)

        self.btnfd.place(x=160, y=180)

        # ...

    def Get\_value(self):

        self.value = int(self.spinbox.get())

    def cut\_frame(self):

        self.Get\_value()

        cap = cv2.VideoCapture(self.fileName)

        fps = cap.get(cv2.CAP\_PROP\_FPS)

        total\_frames = int(cap.get(cv2.CAP\_PROP\_FRAME\_COUNT))

        time\_to\_cut = self.value  # Giây

        frame\_to\_cut = int(time\_to\_cut \* fps)

        cap.set(cv2.CAP\_PROP\_POS\_FRAMES, frame\_to\_cut)

        ret, frame = cap.read()

        if not ret:

            print("Không thể đọc frame.")

            return

        cv2.imshow("Frame", frame)

        cv2.imwrite(self.savepath, frame)

        cv2.waitKey(0)

        cv2.destroyAllWindows()

    def open\_file(self):

        file\_path = filedialog.askopenfilename(

            title="Mở file",

            filetypes=(("mp4 file(.mp4)", "\*.mp4"), ("mov file(.mov)", "\*.mov")),

        )

        self.fileName = file\_path

        self.txtFolderName.config(text=self.fileName)

    def Chon\_duong\_Dan(self):

        self.savepath = filedialog.askdirectory(title="Chọn folder lưu ảnh")

        self.savepath += f"/{self.entr.get()}.jpg"

        cdd\_tk.Label(self, text=self.savepath).place(x=10, y=230)

    def capture\_frames(self):

        # mở cửa sổ chọn file

        file\_path = self.fileName

        # kiểm tra file có tồn tại không

        if not file\_path:

            return

        # khởi tạo đối tượng VideoCapture

        cap = cv2.VideoCapture(file\_path)

        # khởi tạo biến lưu frame số

        frame\_count = 0

        # đọc frame từ video

        while True:

            # Đọc frame tiếp theo

            ret, frame = cap.read()

            # Kiểm tra nếu đọc hết video

            if not ret:

                break

            # Hiển thị frame

            cv2.imshow("frame", frame)

            # Chờ sự kiện từ bàn phím

            key = cv2.waitKey(25)

            # Kiểm tra nếu nhấn space

            if key == ord(" "):

                # Hiển thị hộp thoại để chọn vị trí và đặt tên file

                root = tk.Tk()

                root.withdraw()

                save\_path = filedialog.asksaveasfilename(

                    defaultextension=".jpg", filetypes=[("JPEG files", "\*.jpg")]

                )

                root.destroy()

                if save\_path:

                    # Lưu frame vào file

                    cv2.imwrite(save\_path, frame)

                    frame\_count += 1

            # Kiểm tra nếu nhấn phím 'q'

            if cv2.waitKey(1) & 0xFF == ord("q"):

                break

        cap.release()

        cv2.destroyAllWindows()

class Do\_Thi\_CDD(cdd\_tk.Toplevel):

    def \_\_init\_\_(self, parent):

        super().\_\_init\_\_(parent)

        self.title("Biểu đồ")

        self.geometry("300x300")

        self.frame = cdd\_tk.Frame(self, width=400, height=400, relief=cdd\_tk.SUNKEN)

        self.frame.place(x=10, y=10)

        self.button = cdd\_tk.Button(

            self, text="Vẽ biểu đồ theo quốc gia", width=20, command=self.VeDoThi

        )

        self.button.place(x=10, y=10)

        self.button1 = cdd\_tk.Button(

            self, text="Vẽ biểu đồ theo từng tháng", width=20, command=self.VeDoThi2

        )

        self.button1.place(x=10, y=40)

        self.dict\_tinh = {

            "United Kingdom",

            "France",

            "Australia",

            "Portugal",

        }

        self.combobox = ttk.Combobox(self, values=list(self.dict\_tinh), width=10)

        self.label1 = cdd\_tk.Label(self, text="Chọn quốc gia")

        self.label1.place(x=200, y=10)

        self.combobox.place(x=200, y=50)

        self.lbl1 = cdd\_tk.Label(

            self, text="Thống kê số lượng các sản phẩm bán chạy nhất"

        )

        self.lbl1.place(x=10, y=80)

        self.btn1 = cdd\_tk.Button(self, text="Vẽ", command=self.veDoThi3, width=5)

        self.btn1.place(x=10, y=100)

        self.lbl2 = cdd\_tk.Label(

            self, text="Thống kê số lượng các sản phẩm bán chạy nhất"

        )

        self.lbl2.place(x=10, y=140)

        self.btn2 = cdd\_tk.Button(self, text="Vẽ", command=self.veDoThi4, width=5)

        self.btn2.place(x=10, y=160)

        self.data = pd.read\_csv("G308CDD\_DAHP\_OnlineRetail.csv", encoding="ISO-8859-1")

    def VeDoThi(self):

        country = self.combobox.get()

        tx\_country = self.data.query(f"Country == '{country}'").reset\_index(drop=True)

        # create monthly active customers dataframe

        tx\_country["InvoiceDate"] = pd.to\_datetime(tx\_country["InvoiceDate"])

        tx\_country["InvoiceYearMonth"] = tx\_country["InvoiceDate"].map(

            lambda date: 100 \* date.year + date.month

        )

        tx\_monthly\_active = (

            tx\_country.groupby("InvoiceYearMonth")["CustomerID"].nunique().reset\_index()

        )

        # create bar plot of monthly active customers

        plot\_data = [

            go.Bar(

                x=tx\_monthly\_active["InvoiceYearMonth"],

                y=tx\_monthly\_active["CustomerID"],

            )

        ]

        plot\_layout = go.Layout(

            xaxis={"type": "category"}, title=f"Monthly Active Customers in {country}"

        )

        fig = go.Figure(data=plot\_data, layout=plot\_layout)

        img\_bytes = pio.to\_image(fig, format="png")

        img = plt.imread(io.BytesIO(img\_bytes))

        plt.imshow(img)

        plt.show()

    def VeDoThi2(self):

        self.data["InvoiceDate"] = pd.to\_datetime(self.data["InvoiceDate"])

        self.data["InvoiceYearMonth"] = self.data["InvoiceDate"].map(

            lambda date: 100 \* date.year + date.month

        )

        self.data["Revenue"] = self.data["UnitPrice"] \* self.data["Quantity"]

        tx\_revenue = (

            self.data.groupby(["InvoiceYearMonth"])["Revenue"].sum().reset\_index()

        )

        self.data["InvoiceDate"] = pd.to\_datetime(self.data["InvoiceDate"])

        self.data["InvoiceYearMonth"] = self.data["InvoiceDate"].map(

            lambda date: 100 \* date.year + date.month

        )

        self.data["Revenue"] = self.data["UnitPrice"] \* self.data["Quantity"]

        tx\_revenue = (

            self.data.groupby(["InvoiceYearMonth"])["Revenue"].sum().reset\_index()

        )

        tx\_revenue

        plot\_data = [

            go.Scatter(

                x=tx\_revenue["InvoiceYearMonth"],

                y=tx\_revenue["Revenue"],

            )

        ]

        plot\_layout = go.Layout(xaxis={"type": "category"}, title="Montly Revenue")

        fig = go.Figure(data=plot\_data, layout=plot\_layout)

        img\_bytes = pio.to\_image(fig, format="png")

        # Display the image using plt.imshow()

        img = plt.imread(io.BytesIO(img\_bytes))

        plt.imshow(img)

        plt.show()

    def veDoThi3(self):

        top\_products = self.data["Description"].value\_counts().head(10)

        plt.figure(figsize=(10, 6))

        top\_products.plot(kind="bar")

        plt.xlabel("Product Description")

        plt.ylabel("Quantity Sold")

        plt.title("Top 10 Best-selling Products")

        plt.show()

    def veDoThi4(self):

        revenue\_by\_country = self.data.groupby("Country")["Quantity", "UnitPrice"].sum()

        plt.figure(figsize=(10, 6))

        revenue\_by\_country["UnitPrice"].plot(kind="pie", autopct="%1.1f%%")

        plt.axis("equal")

        plt.title("Revenue Distribution by Country")

        plt.legend(revenue\_by\_country.index)

        plt.show()

myWindow = Window()

myWindow.mainloop()

1. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN CỦA ĐỒ ÁN HỌC PHẦN

## Kết luận

## Những kết quả đạt được

ĐỒ ÁN HỌC PHẦN đã thực hiện việc Lập trình Python …………..; bao gồm:

+ Giao diện đồ họa: GUI

+ Xử lý giọng nói trợ lý ảo: Voice Assistant

+ Phân tích khám phá trên tập dữ liệu thực nghiệm: …………………….

## Hạn chế

Vì thời gian có hạn nên chưa liên kết voice assistant để ra lệnh phân tích khám phá bằng lời nói.

## Hướng phát triển

## Hướng khắc phục các hạn chế

Đồ án sẽ tiếp tục thực hiện iên kết voice assistant để ra lệnh phân tích khám phá bằng lời nói

## Hướng mở rộng ĐỒ ÁN HỌC PHẦN

Sẽ tiếp tục mở rộng sản phẩm đề tài với các ứng dụng thực tiễn.

# DANH MỤC TÀI LIỆU THAM KHẢO

1. Võ Xuân Thể (2023), Tài liệu học tập Lập trình Python, Trường Đại học Sư phạm kỹ thuật Tp.HCM, Tài liệu lưu hành nội bộ được cung cấp theo từng buổi học
2. Võ Xuân Thể (2023), Bài tập thực hành Lập trình Python, Trường Đại học Sư phạm kỹ thuật Tp.HCM, Tài liệu lưu hành nội bộ được cung cấp theo từng buổi học
3. Vijaykumar Ummadisetty 9 (2017): https://www.kaggle.com/datasets/vijayuv/onlineretail

1. IDE (Integrated Development Environment) là môi trường tích hợp dùng để viết code để phát triển ứng dụng [↑](#footnote-ref-1)