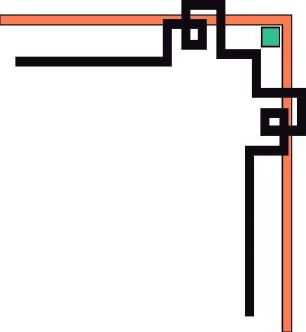
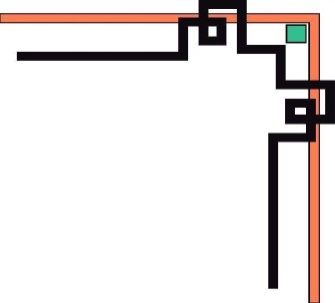
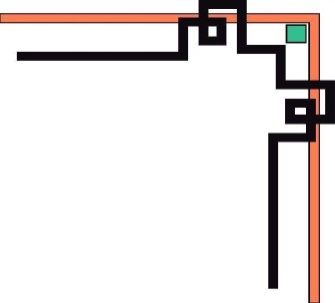
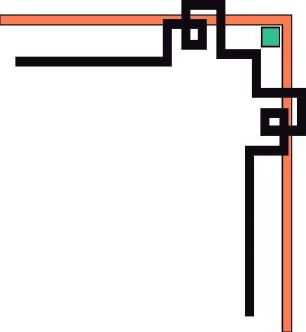
**BỘ GIÁO DỤC VÀ ĐÀO TẠO**



**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

**KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**

**🕯✡🕮🕮✡🕯**

****

**BÁO CÁO ĐỒ ÁN 2**

**GVHD: cô Trần Lê Minh Sang**

**Sinh viên thực hiện:**

Nguyễn Tiến Đạt 16110048

Đỗ Quốc Hùng 16110097

***TP. Hồ Chí Minh, tháng 5 năm 2019***

# NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

Giáo viên hướng dẫn

(ký và ghi họ tên)

# NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN

Giáo viên phản biện

(ký và ghi họ tên)

**MỤC LỤC**

[Phần I. Giới thiệu tổng quan 8](#_Toc528151277)

[I. Tổng quan 8](#_Toc528151278)

[II. Cài đặt 9](#_Toc528151279)

[Phần II. Cấu trúc phân tầng 1](#_Toc528151282)0

[I. Tầng DTO 1](#_Toc528151283)0

[1. Chức năng 1](#_Toc528151284)0

[2. Bảng hàm và lớp 1](#_Toc528151285)1

3. Mã nguồn...............................................................................................................

[II. Tầng DAO 1](#_Toc528151286)2

[1. Chức năng 1](#_Toc528151287)2

[2. Bảng hàm và lớp 1](#_Toc528151288)4

3. Mã nguồn..............................................................................................................

III. Tầng BUS..........................................................................................

1. KNN.........................................................................

a. Bảng hàm..............................................................................

b. Mã nguồn..................................................................................................

c. Đánh giá

2. SVM kernel.........................................................................

a. Bảng hàm.......................................................................

b. Mã nguồn....................................................................

c. Đánh giá

IV. Tầng APP.......................................................................

1. Chức năng..........................................................

2. Bảng hàm....................................................................

3. Mã nguồn.............................................................................

[*Phần III. Đánh giá.............................................................................................. 1*](#_Toc528151289)

I. Đánh giá quá trình thực hiện..................................................................

1. Thuận lợi..........................................................................

2. Khó khăn................................................................................

[Phần III. Phân công và tài liệu tham khảo 1](#_Toc528151289)5

**DANH MỤC CÁC HÌNH**

**DANH MỤC CÁC BẢNG**

**LỜI MỞ ĐẦU**

Hiện nay, sự phát triển của Công nghệ Thông tin ở nước ta đang bước vào thời kì mới với việc triển khai rộng rãi các ứng dụng tin học cho các tổ chức và xã hội. Không ai còn nghi ngờ gì về vai trò của Công nghệ Thông tin trong đời sống, trong khoa học kỹ thuật, kinh doanh, cũng như trong mọi mặt của xã hội, ngay cả đối với một cá nhân.

Tuy nhiên trong thời điểm hiện tại, Công nghệ Thông tin mới chỉ bước đầu được ứng dụng trong đời sống nói chung và giáo dục nói riêng. Việc sử dụng tài liệu điện tử trong dạy và học chưa thực sự phổ biến, chưa mang lại hiệu quả cao thì những quyển sách, giáo trình … vẫn có vai trò vô cùng quan trọng. Đối với những trường lớn, việc quản lý một hệ thống thư viện với hàng ngàn đầu sách và quản lý việc mượn – trả sách của hàng ngàn sinh viên là vô cùng phức tạp. Vì vậy chúng em đã nhận nghiên cứu đề tài:

“Xây dựng chương trình quản lý thư viện”

Chương trình được viết bằng ngôn ngữ Visual Studio C# với cơ sở dữ liệu SQL Server.

Chúng em xin chân thành cảm ơn thầy giáo Nguyễn Minh Đạo đã giúp chúng em hoàn thành đề tài này.

# PHẦN I. GIỚI THIỆU TỔNG QUAN

1. **Tổng quan**

Đồ án được cài đặt chia thành 4 tầng khác nhau nhằm đảm bảo tính linh động cũng như dễ kiểm soát lỗi, nâng cấp và bảo trì. Ở các tầng chỉ có liên kết theo thứ tự như sau App <—> BUS <—> DAO <—> DTO :

• DTO: Nhiệm vụ là tầng chính kết nối đọc và ghi xuống dataset và trả lên đối tượng dữ liệu cho DAO.

• DAO: Nhiệm vụ là biến đổi xử lý dữ liệu và xử lý dữ liệu cho phù hợp với thuật toán và cung cấp đối tượng dữ liệu lên tầng BUS.

• BUS: Là tầng chính xử lý các thuật toán, cung cấp các hàm vẽ và kết quả xử lý các thuật toán lên tầng App.

• APP: Là tầng giao tiếp trực tiếp với người dùng cuối

1. **Cài đặt**

Yêu cầu chương trình hoặc thư viên cần có trên:

• Python3

• Sklearn

• Pandas

# PHẦN II. CẤU TRÚC PHÂN TẦNG

1. **Tầng DTO**

**1.Chức năng**

Trong phân lớp ở tầng này nhiệm vụ là đọc dataset và chuyển nó thành đối tượng dữ liệu và nó được thực thi trong file DataAdapter.py

**2. Bảng hàm và lớp**

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Tên hàm | Mục đích của hàm | File lưu trữ |
| 1 | getFeature1 | Lấy dataset theo từng cột của đối tượng đầu vào và xử lý, cung cấp cho hàm multiprocessing | DataAdapter.py |
| 2 | getFeature2 | Lấy dataset theo từng cột của đối tượng đầu vào và xử lý, cung cấp cho hàm multiprocessing | DataAdapter.py |
| 3 | multiprocessing | Nhiệm vụ là xử lý song song 4 cột cùng 1 lúc để tạo ra đối tượng data cho class dataset | DataAdapter.py |
| 4 | loaiTru | Loại bỏ các ký tự không cần thiết | DataAdapter.py |
| 5 | locKiTuDacBiet | Lọc ra các ký tự đặc biệt liên quan đến cảm xúc | DataAdapter.py |
| 6 | lamSachChuoi | Lọc các lý tự đặc biệt, cung cấp cho hàm ExCSV | DataAdapter.py |
| 7 | ExCSV | Tạo file CSV | DataAdapter.py |
| 8 | dataset | Trả về đối tượng dữ liệu dataset cho tầng DAO sử dụng | DataAdapter.py |

**3. Mã nguồn**

**a. Kết nối dataset**

csv1=pd.read\_csv('../DTO/dataset/CSV/data1.csv',encoding='utf-8')

csv2=pd.read\_csv('../DTO/dataset/CSV/data2.csv',encoding='utf-8')

**b. Hàm getFeature1**

def getFeature1(name):

return (csv1[name].values).tolist()

**c. Hàm getFeature2**

def getFeature2(name):

return (csv2[name].values).tolist()

**d. Hàm multiprocessing**

def multiprocessing(k):

t=['text','label']

pool = ThreadPool(2)

if k==1:

data=pool.map(getFeature1,t)

elif k==2:

data=pool.map(getFeature2,t)

return data

**e. Hàm loaiTru**

for i in s:

if i =='.' or i=='?' or i==',' or i=='0' or i==' ' or i=='1' or i=='2' or i=='3' or i=='4' or i=='5' or i=='6' or i=='7'or i=='8' or i=='9' or i==')'or i=='('or i=='"'or i=='='or i=='>'or i=='<'or i=='&'or i=='^'or i=='\*'or i=='%'or i=='#'or i=='$'or i==''or i=='@'or i==':' or i==';'or i=='-'or i=='+'or i=='^'or i=='&'or i=='|'or i=='{'or i=='}':

return 0

return 1

**f. Hàm locKiTuDacBiet**

s1=""

s2=""

if s!='href' and s!='class' and s!='hashtag-link' and s!= '\n' and s!='thì' and s!='là'and s!='ở'and s!='đi'and s!='tao'and s!='mày'and s!='cây'and s!='đến'and s!='vừng'and s!='bán'and s!='đồ ăn'and s!='Đồ ăn'and s!='cơm\_chiên'and s!='vô'and s!='cách'and s!='đây'and s!='Vị\_trí'and s!='bánh\_bao'and s!='Kem'and s!='từ'and s!='ngoài'and s!='vô'and s!='của'and s!='xe'and s!='thứ'and s!='hôm'and s!='đó'and s!='kho'and s!='quẹt'and s!='buổi\_sáng'and s!='Xe\_đẩy'and s!='decor'and s!='i'and s!='o'and s!='đươ'and s!='c'and s!='n'and s!='cu'and s!='\_' and s!='service'and s!='Menu' and s!='bad'and s!='ㅠ'and s!='bill'and s!='Matcha'and s!='green'and s!='almond'and s!='chocolate'and s!='PERFECT'and s!='kpop'and s!='SG'and s!='upstair'and s!='driving'and s!='in'and s!='to'and s!='check'and s!='say'and s!='ran':

for i in range(len(s)):

if s[i]!='!' and s[i] !='/'and s[i] !='.'and s[i] !="'" :#and s[i]!='['and s[i]!=']'and s[i]!='|'and s[i]!='{'and s[i]!='}'and s[i]!=';'and s[i]!=','

s1+=s[i]

s1.rstrip("\n")

s3=ViPosTagger.postagging(ViTokenizer.tokenize(u"%s"%s1))

for i in range(len(s3[0])):

if s3[1][i]!='N'and s3[1][i]!='Np'and s3[1][i]!='P' and s3[1][i]!='E'and s3[1][i]!='T'and s3[1][i]!='L'and s3[1][i]!='M':

s2+=s3[0][i]+" "

return s2

**g. Hàm lamSachChuoi**

s=""

k=text.split(" ")

for j in k:

if loaiTru(j)==1:

s+=locKiTuDacBiet(j)+" "

return s

**h. Hàm ExCSV**

dt=[]

for i in range(1,kichCoTapTrain): #35473

try:

file=open('../DTO/dataset/pos/%s.txt'%i,"r")

dt.append([lamSachChuoi(lamSachChuoi(file.read())),1])

file.close()

except:

a=2

for i in range(1,kichCoTapTest):#29246

try:

file=open('../DTO/dataset/neg/%s.txt'%i,"r")

dt.append([lamSachChuoi(lamSachChuoi(file.read())),0])

file.close()

except:

a=2

dt=np.array(dt)

df = pd.DataFrame({'text':dt[:,0],'label':dt[:,1]})

df.to\_csv('../DTO/dataset/CSV/data2.csv',encoding='utf-8',index=False)

**i. Lớp dataset**

class dataset:

def \_\_init\_\_(seft,k):

data=multiprocessing(k)

if k==1:

dt1=[]

dt2=[]

for i in range(len(data[0])):

if lamSachChuoi(data[0][i])!=None:

dt1.append(lamSachChuoi(data[0][i]))

dt2.append(data[1][i])

seft.text=dt1

seft.label=dt2

elif k==2:

seft.text=data[0]

seft.label=data[1]

1. **Tầng DAO**

**1.Chức năng**

Mục đích chính ở tầng DAO là biến đổi dữ liệu ban đầu thành các tập dữ liệu phù hợp với thuật toán sử dụng

**2. Bảng hàm và lớp**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên hàm, lớp** | **Mục đích của hàm, lớp** | **File lưu trữ** |
| **1** | demTu | Đếm tổng số từ trong 1 câu | DataProcessing.py |
| **2** | bag\_of\_words | Tạo từ điển,  tạo vector lưu trữ số lần xuất hiện của từ trong từ điển ứng với mỗi câu và vector lưu trữ các từ không xuất hiện trong câu | DataProcessing.py |
| **3** | tinhTF | Tính tần số xuất hiện của 1 từ trong 1 văn bản | DataProcessing.py |
| **4** | tinhIDF | Tính tần số nghịch của 1 từ trong văn bản | DataProcessing.py |
| **5** | tinhTF\_IDF | Tính tần số 1 từ xuất hiện trong văn bản | DataProcessing.py |
| **6** | locTF\_IDF | Loại bỏ các từ có tần số xuất hiện thấp hơn 0.01 | DataProcessing.py |
| **7** | xuLyTF\_IDF | Sắp xếp lại bộ từ điển sau khi loại bỏ các từ có tần số xuất hiện thấp | DataProcessing.py |
| **8** | tienXuLy | Xử lý dữ liệu theo bộ từ điển mới | DataProcessing.py |
| **9** | chuyenSangSo | Gán số, ánh xạ vào cột dataset tương ứng | DataProcessing.py |
| **10** | dataSet | Xử lý, rút chiều dataset | DataProcessing.py |
| **11** | label | Gán label | DataProcessing.py |

**3. Mã nguồn**

**a. Hàm demTu**

def demTu(dt):

k =dt.split(" ")

dem=0

for i in k:

if i !=" ":

dem+=1

return dem

**b. Hàm bag\_of\_words**

def bag\_of\_words(dt):

tuDien=[]

s=[]

s1=[]

# cat chuoi tung cau noi lai

for i in range(len(dt)):

k=dt[i].split(" ")

for j in k:

if j!=' ':

s.append(j)

#loc trung

for i in range(len(s)):

if s[i] not in s1:

s1.append(s[i])

#bo tu dien

for i in range(len(dt)):

array=[]

k=dt[i].split(" ")

for j in s1:

dem=0

if j not in k:

array.append([j,0])

else:

for h in k:

if j ==h:

dem+=1

array.append([j,dem])

tuDien.append(np.array(array))

return tuDien

**c. Hàm tinhTF**

array=[]

tongTu=demTu(dt)

for tu, chiSo in tuDien:

if tu!=' ':

array.append([tu,float(chiSo)/tongTu])

return np.array(array)

**d. Hàm tinhIDF**

def tinhIDF(dt,tuDien):

array=[]

tongCau=len(dt)

dem=0

for tu, chiSo in tuDien[0]:

for cau in dt:

if tu in cau.split():

dem+=1

if(dem>0):

array.append([tu,float(tongCau)/dem])

dem=0

return np.array(array)

**e. Hàm tinhTF\_IDF**

def tinhTF\_IDF(tuDienTF,tuDienIDF):

array=[]

for tu\_tf, chiSo\_tf in tuDienTF:

for tu\_idf, chiSo\_idf in tuDienIDF:

if tu\_tf==tu\_idf:

array.append([tu\_tf,float(chiSo\_tf)\*float(chiSo\_idf)])

return np.array(array)

**f. Hàm locTF\_IDF**

def locTF\_IDF(tuDienTF\_IDF):

array=[]

for tu, chiSo in tuDienTF\_IDF:

if(float(chiSo)>0.01):

array.append([tu,chiSo])

return np.array(array)

**g. Hàm xuLyTF\_IDF**

def xuLyTF\_IDF(data):

dt=data.text

tuDien=bag\_of\_words(dt)

tuDienTF=tuDien

tuDienIDF=[]

tuDienTF\_IDF=[]

#tinhTF

tuDienTF=tuDien

for i in range(len(dt)):

tuDienTF[i]=tinhTF(dt[i],tuDien[i])

#tinhIDF

tuDienIDF=tinhIDF(dt,tuDien)

#tinhTF\_IDF

for i in range(len(tuDienTF)):

tuDienTF\_IDF.append(tinhTF\_IDF(tuDienTF[i],tuDienIDF))

for i in range(len(tuDienTF\_IDF)):

tuDienTF\_IDF[i]=(locTF\_IDF(tuDienTF\_IDF[i]))

return np.array(tuDienTF\_IDF)

**h. Hàm tienXuLy**

s=xuLyTF\_IDF(data)

array1=[]

array2=[]

for cau in s:

for tu, chiSo in cau:

array1.append(tu)

for i in array1:

if i not in array2:

array2.append(i)

return np.array(array2)

**i. Hàm chuyenSangSo**

def chuyenSangSo(dt,tuDien):

k=dt.split()

a=[]

for i in tuDien:

a.append(0)

a=np.array(a)

for i in k:

for j in range(len(tuDien)):

if i==tuDien[j]:

a[j]=1

return a

**j. Hàm dataSet**

def dataSet(k):

da.ExCSV()

data=[]

if k==1:

data=da.dataset(1)

elif k==2:

data=da.dataset(2)

dt=data.text

dataset=[]

tuDien=tienXuLy(data)

for i in range(len(dt)):

dataset.append(chuyenSangSo(dt[i],tuDien))

#rut chieu du lieu

dataset=np.array(dataset)

sc = StandardScaler()

X = sc.fit\_transform(dataset)

pca = PCA(n\_components=3)

X\_pca = pca.fit\_transform(X)

os.system('cls' if os.name == 'nt' else 'clear')

return X\_pca

**k. Hàm label**

def label(k):

data=[]

if k==1:

data=da.dataset(1)

elif k==2:

data=da.dataset(2)

return data.label

1. **Tầng BUS**

**1.KNN**

**a. Bảng hàm**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên hàm** | **Mục đích của hàm** | **File luu trữ** |
| **1** | Knn | Train data, tính độ chính xác htoe giải thuật KNN | knn.py |

**b. Mã nguồn**

def Knn(k):

data=[]

label=[]

if k==1:

data=dp.dataSet(1)

label=dp.label(1)

elif k==2:

data=dp.dataSet(2)

label=dp.label(2)

x\_train, x\_test, y\_train, y\_test=train\_test\_split(data,label,

test\_size=0.2,random\_state=1)

clf=KNeighborsClassifier(n\_neighbors=3).fit(x\_train,y\_train)

precision= precision\_score(y\_test,clf.predict(x\_test), average='weighted')

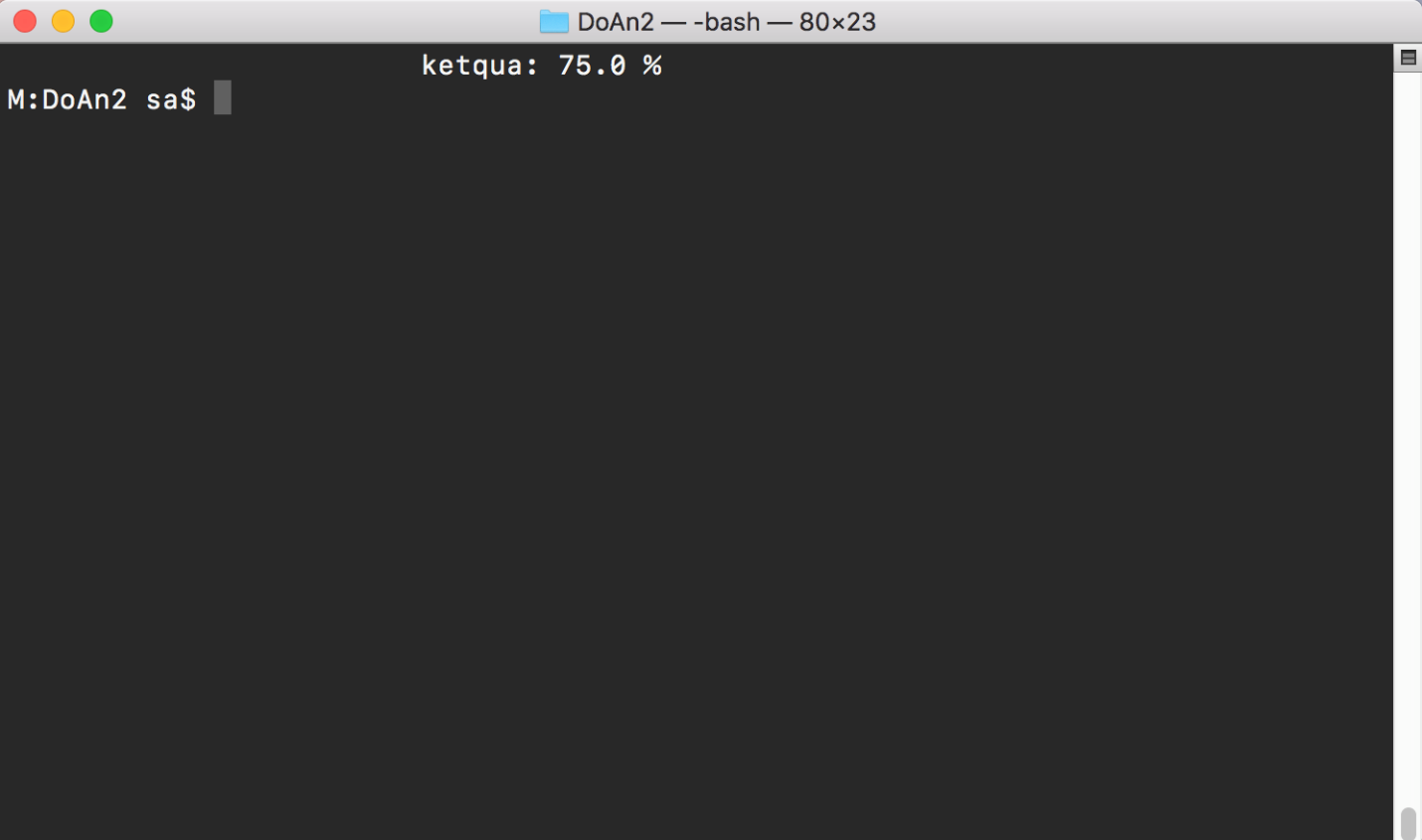
recall= recall\_score(y\_test,clf.predict(x\_test), average='weighted')

F=(2\*precision\*recall)/(precision+recall)

return F

**c. Đánh giá**

Kết quả khi thực thi với 50 câu theo knn là 75%



**2. SVM kernel**

**a. Bảng hàm**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên hàm** | **Mục đích của hàm** | **File lưu trữ** |
| **1** | Svmker | Train data, chạy gridsearch tìm ra best param và best score | svm kernel.py |

**b. Mã nguồn**

def Svmker():

x\_train, x\_test, y\_train, y\_test=train\_test\_split(

dp.dataSet(2),

dp.label(2),

test\_size=0.1,random\_state=1)

grid = [{'kernel': ['rbf'], 'gamma': [1e-2, 1e-4],

'C': [1, 10, 100]}]

grid = GridSearchCV(SVC(), grid, cv=5)

grid.fit(x\_train, y\_train)

print("The best parameters are %s with a score of %0.2f"% (grid.best\_params\_, grid.best\_score\_))

**c. Đánh giá**

1. **Tầng APP**

**1.Chức năng**

Là tầng cấu hình, cài đặt, biên dịch và gọi nghiệp vụ xử lý của tầng BUS hiển thị kết quả ra ngoài như giao diện trực quan hay command line interface

**2. Bảng hàm**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên hàm** | **Chức năng của hàm** | **File lưu trữ** |
| **1** | KNN | Gọi thực thi hàm Knn ở tầng BUS và in ra kết quả | Main.py |
|  |  |  |  |

**3. Mã nguồn**

**a. Hàm KNN**

def KNN(k):

a=knn.Knn(k) \*100

s="\t\t\tketqua: "+str(a)+" %"

print(s)

# PHẦN III. ĐÁNH GIÁ

## Đánh giá quá trình thực hiện

### Thuận lợi

* Nhiều công nghệ hỗ trợ cho việc phát triển đồ án
* Nhiều thư viện hỗ trợ trong python
* Nguồn tài liệu phong phú

### Khó khăn

* Ít kinh nghiệm trong việc xứ lý dữ liệu, áp dụng thuật toán xứ lý
* Chưa tận dụng tối ưu chức năng các thư viện, công nghệ

# PHẦN IV. PHÂN CÔNG VÀ TÀI LIỆU THAM KHẢO

|  |  |  |
| --- | --- | --- |
| Tên SV | Đóng góp | Công việc |
| Nguyễn Tiến Đạt | 50% | Tìm hiểu, góp ý, tìm kiếm dataset, code và báo cáo |
| Đỗ Quốc Hùng | 50% | Tìm hiểu, góp ý, tìm kiếm dataset, code và báo cáo |