```
# -*- coding: utf-8 -*-
!pip install pyvi
from __future__ import print_function
from sklearn import metrics
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import CountVectorizer,TfidfTransformer
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.svm import LinearSVC
from sklearn.linear_model import LogisticRegression
# from sklearn.neural_network import MLPClassifier
# from sklearn.tree import DecisionTreeClassifier
# from sklearn.naive_bayes import MultinomialNB
# from sklearn.linear_model import SGDClassifier
# from sklearn.neighbors import KNeighborsClassifier
# from sklearn.ensemble import AdaBoostClassifier
# from sklearn.ensemble import RandomForestClassifier
import pandas as pd
from pyvi import ViTokenizer
import re
import string
import codecs
import json
```

```
Collecting pyvi
  Downloading pyvi-0.1.1-py2.py3-none-any.whl (8.5 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 8.5/8.5 MB 34.6 MB/s eta 0:00:00
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-packages (from pyvi) (1.2.2)
Collecting sklearn-crfsuite (from pyvi)
  Downloading sklearn_crfsuite-0.3.6-py2.py3-none-any.whl (12 kB)
Requirement already satisfied: numpy>=1.17.3 in /usr/local/lib/python3.10/dist-packages (from scikit-learn->pyvi) (1.23.5)
Requirement already satisfied: scipy>=1.3.2 in /usr/local/lib/python3.10/dist-packages (from scikit-learn->pyvi) (1.11.4)
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from scikit-learn->pyvi) (1.3.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn->pyvi) (3.2.0)
Collecting python-crfsuite>=0.8.3 (from sklearn-crfsuite->pyvi)
  Downloading python_crfsuite-0.9.9-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (993 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 993.5/993.5 kB 64.2 MB/s eta 0:00:00
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from sklearn-crfsuite->pyvi) (1.16.0)
Requirement already satisfied: tabulate in /usr/local/lib/python3.10/dist-packages (from sklearn-crfsuite->pyvi) (0.9.0)
Requirement already satisfied: tqdm>=2.0 in /usr/local/lib/python3.10/dist-packages (from sklearn-crfsuite->pyvi) (4.66.1)
Installing collected packages: python-crfsuite, sklearn-crfsuite, pyvi
Successfully installed python-crfsuite-0.9.9 pyvi-0.1.1 sklearn-crfsuite-0.3.6
```

tiền sử lý văn bản, chuẩn hóa dữ liệu

```
#Từ điển tích cực, tiêu cực, phủ định
path_nag = '/content/drive/MyDrive/sentiment_analysis/data_clean/VietNamese_sentiment_dictionary/nag.txt'
path_pos = '/content/drive/MyDrive/sentiment_analysis/data_clean/VietNamese_sentiment_dictionary/pos.txt'
path_not = '/content/drive/MyDrive/sentiment_analysis/data_clean/VietNamese_sentiment_dictionary/not.txt'

with codecs.open(path_nag, 'r', encoding='UTF-8') as f:
    nag = f.readlines()
nag_list = [n.replace('\n', '') for n in nag]

with codecs.open(path_pos, 'r', encoding='UTF-8') as f:
    pos = f.readlines()
pos_list = [n.replace('\n', '') for n in pos]
with codecs.open(path_not, 'r', encoding='UTF-8') as f:
    not_ = f.readlines()
not_list = [n.replace('\n', '') for n in not_]

# xác định các kí tự có dấu trong tiếng Việt
VN_CHARS_LOWER = u'ạảãàáâậầấẩẫăắằặẳẵóòọõỏôộổỗơờớợởỡéèẻẹẽêếềệểễúùụủũưựữửừứíìịỉĩýỳỷỵỹđð'
VN_CHARS_UPPER = u'ẠẢÃÀÁÂẬẦẤẨẪĂẮẰẶẲẴÓÒỌÕỎÔỘỔỖƠỜỚỢỞỠÉÈẺẸẼÊỀẾỆỂỄÚÙỤỦŨƯỰỮỬỪỨÍÌỊỈĨÝỲỶỴỸĐĐ'
VN_CHARS = VN_CHARS_LOWER + VN_CHARS_UPPER


#Viết hàm loại bỏ dấu ủa kí tự
def no_marks(s):
    __INTAB = [ch for ch in VN_CHARS]
    __OUTTAB = "a"*17 + "o"*17 + "e"*11 + "u"*11 + "i"*5 + "y"*5 + "d"*2
    __OUTTAB += "A"*17 + "O"*17 + "E"*11 + "U"*11 + "I"*5 + "Y"*5 + "D"*2
    __r = re.compile("|".join(__INTAB))
    __replaces_dict = dict(zip(__INTAB, __OUTTAB))
    result = __r.sub(lambda m: __replaces_dict[m.group(0)], s)
    return result
```

```
#Viết hàm để chuẩn hóa các từ teencode, emoji, sai chính tả, tiếng anh, dấu câu,...
def normalize_text(text):

    #Remove các ký tự kéo dài: vd: đẹpppppppp
    text = re.sub(r'([A-Z])\1+', lambda m: m.group(1).upper(), text, flags=re.IGNORECASE)

    # Chuyển thành chữ thường
    text = text.lower()

    #Chuẩn hóa tiếng Việt, xử lý emoj, chuẩn hóa tiếng Anh, thuật ngữ
    with open('/content/drive/MyDrive/sentiment_analysis/list.json', 'r',encoding='utf-8') as file:
        data = json.load(file)

    replace_list = data

    for k, v in replace_list.items():
        text = text.replace(k, v)

    # chuyển dấu câu thành space
    translator = str.maketrans(string.punctuation, ' ' * len(string.punctuation))
    text = text.translate(translator)

    text = ViTokenizer.tokenize(text)  # Thực hiện tokennize các câu
    texts = text.split()
    len_text = len(texts)

    texts = [t.replace('_', ' ') for t in texts]# bỏ dấu "_" trong các token

 # Xử lý vấn đề phủ định (VD: áo này chẳng đẹp--> áo này notpos)
    for i in range(len_text):
        cp_text = texts[i]
        if cp_text in not_list: # nếu có từ phủ định thì thực hiện đổi thành 'notpos' hoặc 'notnag'
            numb_word = 2 if len_text - i - 1 >= 4 else len_text - i - 1

            for j in range(numb_word):
                if texts[i + j + 1] in pos_list:
                    texts[i] = 'notpos'
                    texts[i + j + 1] = ''

                if texts[i + j + 1] in nag_list:
                    texts[i] = 'notnag'
                    texts[i + j + 1] = ''
        else: #Thêm feature cho những sentiment words vào cuối câu nếu nó không có từ phủ định (áo này đẹp--> áo này đẹp positive)
            if cp_text in pos_list:
                texts.append('positive')
            elif cp_text in nag_list:
                texts.append('nagative')

    text = u' '.join(texts)

    #remove nốt những ký tự thừa thãi như "", kí tự đặc biệt
    text = text.replace(u'"', u' ')
    text = text.replace(u'', u'')
    text = text.replace('','')
    return text


#Text hàm chẩn hóa
print(normalize_text('không đẹp xíu nào'))
print(normalize_text('không xấu nha'))
print(normalize_text(' đẹp quá'))
print(normalize_text('sản phẩm không thể chê vào đâu được, quá tuyệt vời '))
```

```
 notpos  xíu nào
 notnag  nha
 đẹp quá positive
 sản phẩm không thể chê vào đâu được quá tuyệt vờ yêu positive positive positive
```

```
# tạo class với các phương thức để truy cập dữ liệu
class DataSource(object):
    #Hàm đọc dữ liệu từ file và chia  thành các comment riêng biệt của từng khách hàng .
    def _load_raw_data(self, filename, is_train=True):

        a = []
        b = []

        regex = 'train_'
        if not is_train:
            regex = 'test_'

        with open(filename, 'r') as file:
            for line in file:
                if regex in line:
                    b.append(a)
                    a = [line]
                elif line != '\n':
                    a.append(line)
        b.append(a)

        return b[1:]
    # Đối cới một khách hàng tạo ra một dictionary bao goomg ID, lable, và phần comment
    def _create_row(self, sample, is_train=True):

        d = {}
        d['id'] = sample[0].replace('\n', '')
        review = ""

        if is_train:
            for clause in sample[1:-1]:
                review += clause.replace('\n', ' ')
                review = review.replace('.', ' ')

            d['label'] = int(sample[-1].replace('\n', ' '))
        else:
            for clause in sample[1:]:
                review += clause.replace('\n', ' ')
                review = review.replace('.', ' ')


        d['review'] = review

        return d
    # Viết hàm thực hiện kết hợp hai phương thức _load_raw_data và _create_row để chiuyeenr tất cả dữ liệu thabhf một danh sách hoàn ch
    def load_data(self, filename, is_train=True):

        raw_data = self._load_raw_data(filename, is_train)
        lst = []

        for row in raw_data:
            lst.append(self._create_row(row, is_train))

        return lst
    # Hàm thực hiện việc chuẩn hóa dữ liệu
    def transform_to_dataset(self, x_set,y_set):
        X, y = [], []
        for document, topic in zip(list(x_set), list(y_set)):
            document = normalize_text(document)
            X.append(document.strip())
            y.append(topic)
            #Augmentation bằng cách remove dấu tiếng Việt
            X.append(no_marks(document))
            y.append(topic)
        return X, y



##  load và xử lý dữ liệu từ train.crash cho tập train, sau đó nối thêm dữ liệu mới vào dữ liệu huấn luyện

ds = DataSource()
train_data = pd.DataFrame(ds.load_data('/content/drive/MyDrive/sentiment_analysis/data_clean/train.crash'))
new_data = []

#Thêm mẫu bằng cách lấy trong từ điển Sentiment (nag/pos)
for index,row in enumerate(nag_list):
    new_data.append(['pos'+str(index),'0',row])
for index,row in enumerate(nag_list):
    new_data.append(['nag'+str(index),'1',row])

new_data = pd.DataFrame(new_data,columns=list(['id','label','review']))
train_data.append(new_data)
```

```
# load dữ liệu cho tập test từ test.crash
test_data = pd.DataFrame(ds.load_data('/content/drive/MyDrive/sentiment_analysis/data_clean/test.crash', is_train=False))
```

```
<ipython-input-9-9917524a8c0f>:14: FutureWarning: The frame.append method is deprecated and will be removed from pandas in a future
  train_data.append(new_data)
```

```
### Try model MultinomialNB
# from sklearn.naive_bayes import MultinomialNB
# from sklearn.svm import LinearSVC
# classifiers = MultinomialNB(),



###Try model DecisionTreeClassifier
# from sklearn.tree import DecisionTreeClassifier
# classifiers = MultinomialNB(),



# model LinearSVC
classifiers = LinearSVC(fit_intercept = True,multi_class='crammer_singer', C=1),


##chuyển đổi dữ liệu kiểm tra thành dạng phù hợp để kiểm tra mô hình.
X_train, X_test, y_train, y_test = train_test_split(train_data.review, train_data.label, test_size=0.3,random_state=42)
X_train, y_train = ds.transform_to_dataset(X_train,y_train)
X_test, y_test = ds.transform_to_dataset(X_test, y_test)



#THÊM STOPWORD LÀ NHỮNG TỪ KÉM QUAN TRỌNG
stop_ws = [u'rằng',u'thì',u'là',u'mà']


# Thực hiện mô hình phân loại và gán nhãn cho dữ liệu 0: posity, 1 negative
for classifier in classifiers:
    steps = []
    steps.append(('CountVectorizer', CountVectorizer(ngram_range=(1,5),stop_words=stop_ws,max_df=0.5, min_df=5)))
    steps.append(('tfidf', TfidfTransformer(use_idf=False, sublinear_tf = True,norm='l2',smooth_idf=True)))
    steps.append(('classifier', classifier))
    clf = Pipeline(steps)
    clf.fit(X_train, y_train)
    y_pred = clf.predict(X_test)
    report1 = metrics.classification_report(y_test, y_pred, labels=[1,0], digits=3)

X_train, y_train = ds.transform_to_dataset(train_data.review, train_data.label)



#TRAIN OVERFITTING/ERRO ANALYSIS
clf.fit(X_train, y_train)
y_pred = clf.predict(X_train)
report2 = metrics.classification_report(y_train, y_pred, labels=[1,0], digits=3)



#ERRO ANALYSIS
# Cho thấy các trường hợp mà nhãn dự đoán của mô hình (y_pred) không trùng khớp với nhãn thực tế (y_train).
for id,x, y1, y2 in zip(train_data.id, X_train, y_train, y_pred):
    if y1 != y2:
        # CHECK EACH WRONG SAMPLE POSSITIVE/NAGATIVE
        if y1!=1:#0:
            print(id,x, y1, y2)# In ra các trường hợp phân loại sai
```

```
train_004375 hang o khong 0 1
train_004512 gó yêu g yêu ấy bé hơn loạ yêu 22 25k positive positive positive 0 1
train_004513 go yeu g yeu ay be hon loa yeu 22 25k positive positive positive 0 1
train_004656 o notpos  đáy 0 1
train_004657 o notpos  day 0 1
train_004784 t yêu ền nào của đó nó yêu chung tạm được mình không th gì c notpos cá  mặt kính lồ yêu phía trên ô ngày th yêu ết k
train_004785 t yeu en nao cua do no yeu chung tam duoc minh khong th gi c notpos ca  mat kinh lo yeu phia tren o ngay th yeu et k
train_004926 t yêu ền nào của ý positive 0 1
train_004927 t yeu en nao cua y positive 0 1
train_005100 mình đang dung không e 3 gần năm vẫn chạy tốt chỉ hư phần cảm ứng kh yêu không dùng bút nhưng đã đặt cọc không e 7 đ
train_005256 mua duoc vo yeu g yeu a flash 5 khong kem do yeu gold danh g yeu a do yeu nay co g yeu ay de chiu hon notnag  co xat
train_005483 theo mình nghĩ chát đay khôngphả yêu la mục xay ma la thịch heo th yêu pha yêu khôngco mu yêu v yêu của nó nhúng van
train_005626 hang dep nhung co kich cobao can nang len han 3 khong ma mac quan ao van bi chat positive nagative nagative nagative
train_005881 bức hình nào là nhỏ xíu và 950 chưa b gì o là cá yêu đt có camera cù yêu positive positive 0 1
train_005882 buc hinh nao la nho xiu va 950 chua b gi o la ca yeu dt co camera cu yeu positive positive 0 1
train_006321 đã nhận được hàng đặt hàng gh yêu chú lấy màu đen nhưng kh yêu gử yêu lạ yêu gử yêu màu xanh bây g yêu ờ muốn đổ yêu
train_006322 da nhan duoc hang dat hang gh yeu chu lay mau den nhung kh yeu gu yeu la yeu gu yeu mau xanh bay g yeu o muon do yeu
train_006481 samsung làm chủ dây truyền và công nghệ ho tha hồ ra sản phẩm yêu phon muốn làm cá yêu vỏ cũng phả yêu đ yêu thuê hã
train_006482 samsung lam chu day truyen va cong nghe ho tha ho ra san pham yeu phon muon lam ca yeu vo cung pha yeu d yeu thue ha
train_006623 huawe yêu matebo không đầu tư kĩ lưỡng ghê muốn có em nó quá à positive nagative 0 1
train_006782 mo yeu nhan duoc hang chua xem nhung nh yeu `n cach dong go yeu thi khongha yeu long khongb yeu et do cua hang hay c
train_007081 mình nó yêu s6 s 6 egde xanh lục bảo người yêu châu á rất chuộng positive positive 0 1
train_007082 minh no yeu s6 s 6 egde xanh luc bao nguo yeu chau a rat chuong positive positive 0 1
train_007216 pha yeu co otg mo yeu dung duoc nha a yeu khongxa yeu duoc thi dung vao rate 1star nha to yeu cua hang lam nagat yeu
train_007235 sản phẩm g yêu ong trong h yêu `nh positive positive 0 1
train_007256 cua hang phuc vu khach chu dao positive 0 1
train_007340 con e y het con nay nghe bas manh khongre cung mung positive 0 1
train_007560 dung duoc 2 ngay ro yeu thay cung do tham positive positive nagative 0 1
train_007614 ch yeu eu da yeu vua n phan co chan ho yeu rong yeu khong notpos thoa  ma yeu 1 positive positive positive positive
train_007691 g yêu ao hàng quá chậm trog thành phố mà tận 4 ngày nhưng máy thì xà yêu được positive nagative positive positive 0 1
train_007692 g yeu ao hang qua cham trog thah pho ma tan 4 ngay nhung may thi xa yeu duoc positive nagative positive positive 0 1
train_008219 sản phẩm dùng o không một lưu ý nhỏ phần móc tay bằng chun bị sờn chỉ kh yêu sử dụng ngay lần đầu nagative positive
train_008220 san pham dung o khong mot luu y nho phan moc tay bang chun bi son chi kh yeu su dung ngay lan dau nagative positive
train_008272 be nha minh khong hop tac h yeu ch yeu c positive positive 0 1
train_008277 hàng o notpos 0 1
train_008278 hang o notpos  0 1
train_008299 sạc không dây dễ quá mà 0 1
train_008399 mình không đọc rồ yêu khônghay lắm nh yêu ều lúc cứ v yêu ết cá yêu gì ấy notpos l  ên quan đến bình thường s gì cả
train_008400 minh khong doc ro yeu khonghay lam nh yeu eu luc cu v yeu et ca yeu gi ay notpos l  en quan den binh thuong s gi ca
train_008414 tay da chet mu yeu tra xanh sang khoa yeu lam luon san pham y nhu hinh hang duoc cua hang dong go yeu ky co d yeu em
train_008518 mu s yeu eu dthuong positive 0 1
```

```
#Đánh giá mô hình
#CROSS VALIDATION
cross_score = cross_val_score(clf, X_train,y_train, cv=5)


# in kết quả đánh giá
print('DATASET LEN %d'%(len(X_train)))
print('TRAIN 70/30 \n\n',report1)
print('TRAIN OVERFITING\n\n',report2)
print("CROSSVALIDATION 5 FOLDS: %0.4f (+/- %0.4f)" % (cross_score.mean(), cross_score.std() * 2))
```

```
DATASET LEN 32146
TRAIN 70/30

              precision    recall  f1-score   support

           1      0.918     0.939     0.928      4438
           0      0.947     0.929     0.938      5206

    accuracy                          0.933      9644
   macro avg      0.932     0.934     0.933      9644
weighted avg      0.934     0.933     0.933      9644

TRAIN OVERFITING

              precision    recall  f1-score   support

           1      0.973     0.979     0.976     14766
           0      0.982     0.977     0.980     17380

    accuracy                          0.978     32146
   macro avg      0.978     0.978     0.978     32146
weighted avg      0.978     0.978     0.978     32146

CROSSVALIDATION 5 FOLDS: 0.9318 (+/- 0.0101)
```

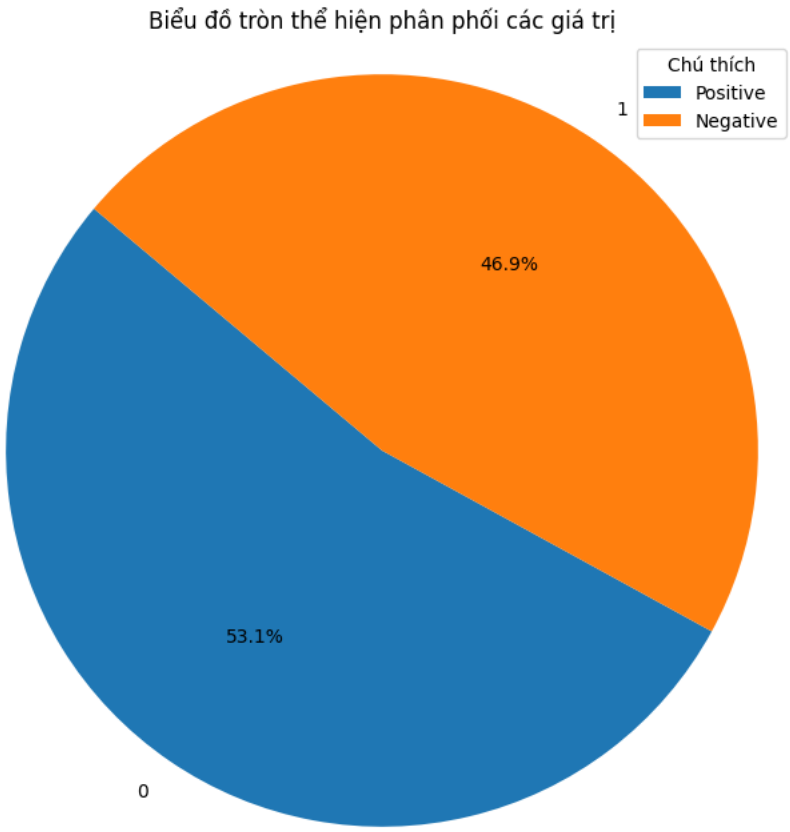
```
##lưu dữ liệu kết quả phân tích vào FILE SUBMIT
test_list = []
for document in test_data.review:
```

```
##phân tích kết quả thu được bằng đồ thị:
import matplotlib.pyplot as plt

# Đọc dữ liệu từ file CSV vào một DataFrame của pandas
data = pd.read_csv('submit.csv')
# Xác định cột dữ liệu bạn muốn sử dụng để vẽ biểu đồ tròn
column_to_plot = 'label'

# Tính toán số lần xuất hiện của từng giá trị trong cột dữ liệu
value_counts = data[column_to_plot].value_counts()

# Vẽ biểu đồ tròn
plt.figure(figsize=(8, 8))
domain_names = ['Positive','Negative']
plt.pie(value_counts, labels=value_counts.index, autopct='%1.1f%%', startangle=140)
plt.axis('equal')  # Đảm bảo biểu đồ tròn có hình dạng hợp lý
plt.title('Biểu đồ tròn thể hiện phân phối các giá trị')
plt.legend(domain_names, title='Chú thích', loc='upper right')
plt.show()
```



Mục mới