

ĐỒ ÁN

**XÂY DỰNG MÔ HÌNH ĐÁNH
GIÁ ĐIỆN THOẠI QUA BÌNH
LUẬN CỦA TRANG
THEGIOIDIDONG.COM**

Thành viên

Trần Đức Thắng - 1712761

Lê Quang Quí - 1712710



I. Trình bày bài toán

1. Đầu vào:

- Dataset được chuẩn bị sẵn từ khoảng 18.000 câu comment của người Việt đánh giá các sản phẩm điện thoại khi mua hàng:
 - 0: Tốt, tích cực hoặc trung gian.
 - 1: Không tốt, tiêu cực.
- Các comment trên trang này sẽ được scrape về, sử dụng thư viện selenium để scrape.

2. Đầu ra:

- Dựa vào tính toán số comment tốt/xấu của sản phẩm để đưa ra lời khuyên cho khách hàng.

```
time.sleep(5)
for elem in elems:
    list_web.append(elem.get_attribute("href"))
```

```
for i in list_web:
    if('/dtdd/' in i):
        print(i)
```

```
https://www.thegioididong.com/dtdd/samsung-galaxy-m51
https://www.thegioididong.com/dtdd/iphone-12-128gb
https://www.thegioididong.com/dtdd/oneplus-nord-n10-5g
https://www.thegioididong.com/dtdd/realme-c17
https://www.thegioididong.com/dtdd/vivo
https://www.thegioididong.com/dtdd/nokia-34
https://www.thegioididong.com/dtdd/samsung-galaxy-note-20-ultra
https://www.thegioididong.com/dtdd/oppo-a92
https://www.thegioididong.com/dtdd/iphone-12-mini
https://www.thegioididong.com/dtdd/oppo-reno4
https://www.thegioididong.com/dtdd/iphone-12
https://www.thegioididong.com/dtdd/samsung-galaxy-a51-8gb
https://www.thegioididong.com/dtdd/samsung-galaxy-note-20-ultra
https://www.thegioididong.com/dtdd/samsung-galaxy-a51-8gb
https://www.thegioididong.com/dtdd/realme-c15
```

II. Craw comment bằng selenium

Craw danh sách link các điện thoại có trong web.

II. Crawl comment bằng selenium

- Thư viện Selenium WebDriver cung cấp tất cả các Webdriver implementations. Ở đây sử dụng Webdriver implementations là Chrome.

```
from selenium import webdriver
import time
list_web = []
browser = webdriver.Chrome(executable_path='chromedriver.exe')
browser.get("https://www.thegioididong.com/dtdd#i:20")
```

- Dùng phương thức Get để chuyển hướng đến trang ***https://www.thegioididong.com/dtdd#i:20***

II. Crawl comment bằng selenium

- Lấy danh sách Link sản phẩm trên trang thế giới di động.
 - Dùng phương thức `find_elements_by_xpath` để tìm element chứa link sản phẩm.

```
elems = browser.find_elements_by_xpath("//a[@href]")
time.sleep(5)
for elem in elems:
    list_web.append(elem.get_attribute("href"))
```

```
▼ <a href="/dtdd/samsung-galaxy-m51" class=" version2020 large" data-s=
"0"> == $0
```

II. Crawl comment bằng selenium

- Code

```
from requests_html import HTMLSession, HTML
from bs4 import BeautifulSoup
import math
import re

link = open('laptop.csv', 'r')

for i in link:
    star = []
    session = HTMLSession()
    r = session.get(i.rstrip() + '/danh-gia')
    data = r.html.html
    soup = BeautifulSoup(data, "html.parser")

    for t in soup.find_all('strong'):
        if(t.text.isdigit()):
            star.append(t.text)

    n = 5
    for j in star:
        a = open('comment{}.txt'.format(n), 'a', encoding='utf-8')
        a.write('Name: ' + i.split('https://www.thegioididong.com/dtdd/')[1].replace('-', ' ').title() + '\n')
        page = math.ceil(int(j)/50)
        for k in range(page):
            session = HTMLSession()
            r = session.get(i.rstrip() + '/danh-gia?s={}&p={}'.format(n, k+1))
            print(i.rstrip() + '/danh-gia?s={}&p={}'.format(n, k+1))
            data = r.html.html
            soup = BeautifulSoup(data, "html.parser")
            for item in soup.find_all('ul', {"class" : "ratingLst"}):
                cmt = item.select('i')
                for count in range(len(cmt)):
                    if(count > 2 and cmt[count-2].text == '' and cmt[count].text != ''):
                        a.write(cmt[count].text.rstrip() + '\n' + '\n')

        a.close()
        del a
        n = n - 1

link.close()
```

III. Khám phá dữ liệu

- Đọc dữ liệu từ file

Do trong các comment đôi khi có dấu xuống dòng ("\n") vì vậy nếu đọc file theo các thông thường từ các thư viện của pandas sẽ dẫn đến dữ liệu sai và thiếu, vì vậy cần viết hàm đọc file mới.

```
# Load data
def load_data_model(filepath, n, star):
    data = []
    with open(filepath, encoding="utf8") as fp:
        line = fp.readline()
        if line.strip().split(':')[0] == 'Name':
            name = line.strip().split(':')[1]
            temp = ''
            cnt = 1
            while line:
                #print("Line {}: {}".format(cnt, line.strip()))
                if line.strip().split(':')[0] == 'Name':
                    name = line.strip().split(':')[1]
                    line = fp.readline()
                elif line.strip() == '':
                    data.append([name, temp, n, star])
                    temp = ''
                else:
                    temp += ' ' + line.strip()
                    line = fp.readline()
            return pd.DataFrame(data, columns=['model', 'Text', 'Sentiment', 'Star'])
```

III. Khám phá dữ liệu

- Số lượng bản ghi và ý nghĩa trường dữ liệu

```
num_rows, num_cols = df.shape  
print("Dữ liệu có {} dòng, {} cột.".format(num_rows, num_cols))
```

Dữ liệu có 34346 dòng, 4 cột.

- model : Tên điện thoại
- Text : Nhận xét về điện thoại
- Sentiment : Đánh giá (tốt/xấu)
- Star : Số sao

IV. Tiền xử lí dữ liệu

- Dữ liệu ở dạng thô là các câu comment chứa:
 - Dấu câu
 - Các tiếng rời rạc
 - Các từ lặp lại nhiều lần nhưng không quan trọng
- Cần thực hiện:
 - ❖ Loại bỏ dấu câu
 - ❖ Ghép các chữ rời rạc thành từ có nghĩa
 - ❖ Tính tần số xuất hiện và độ quan trọng của từ

❖ Loại bỏ dấu câu

```
def standardize_data(row):  
    # Xóa dấu chấm, phẩy, hỏi ở cuối câu  
    row = re.sub(r"[\.,\?]+$-", "", row)  
    # Xóa tất cả dấu chấm, phẩy, chấm phẩy, chấm than, ... trong câu  
    row = row.replace(",", " ").replace(".", " ") \\  
        .replace(";", " ").replace("<";", " ") \\  
        .replace(":", " ").replace("<";", " ") \\  
        .replace("'", " ").replace("<";", " ") \\  
        .replace("!", " ").replace("<";", " ") \\  
        .replace("-", " ").replace("<";", " ")  
    # xóa số trong câu  
    #row = re.sub(r'[0-9]+', '', row)  
    row = row.strip()  
    return row  
  
def tokenizer(row):  
    return word_tokenize(row, format="text")
```

- Regex (Regular Expression): Module re trong python.

❖ Ghép các chữ rời rạc thành từ có nghĩa

- Tokenize: Thư viện underthesea – chuyên về xử lý ngôn ngữ Tiếng Việt

```
def tokenizer(row):  
    return word_tokenize(row, format="text")
```

- Hàm word_tokenize() sẽ thực hiện việc ghép các tiếng phù hợp để tạo các từ có nghĩa trong Tiếng Việt

https://underthesea.readthedocs.io/en/v1.1.8/_modules/underthesea/word_tokenize.html

❖ Tần số xuất hiện và độ quan trọng của từ

- Thư viện sklearn, hàm TfidfVectorizer()

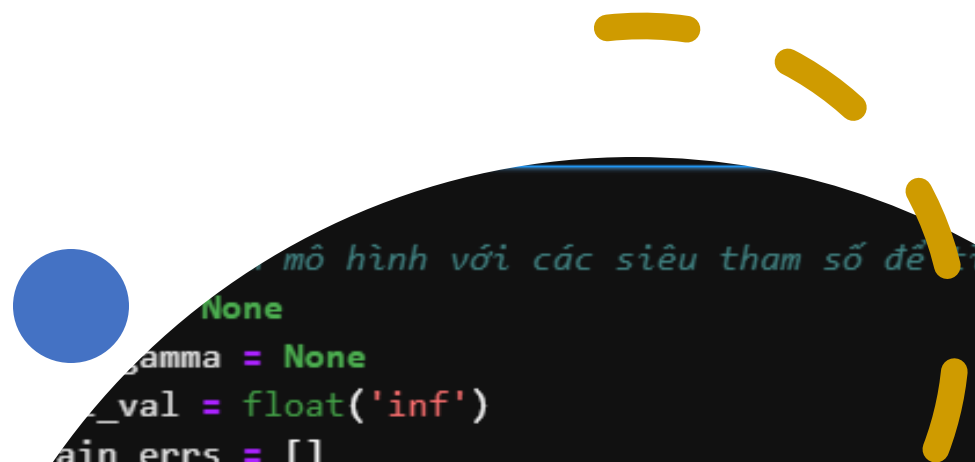
```
def embedding(X_train, X_test=None):  
    global emb  
    emb = TfidfVectorizer(min_df=5, max_df=0.8, max_features=3000, sublinear_tf=True)  
    emb.fit(X_train)  
  
    # Save pkl file  
    joblib.dump(emb, 'tfidf.pkl')  
    X_train = emb.transform(X_train)  
    #if X_test:  
    X_test = emb.transform(X_test)  
    return X_train, X_test
```

- => Chuyển phần văn bản nhập vào hàm (các câu comment ở trên) thành dạng ma trận để đưa vào model.

V. huấn luyện mô hình

1. Huấn luyện bằng mô hình máy học

- Model sử dụng: SVC
 - Thử nghiệm mô hình với các siêu tham số để tìm ra bộ siêu tham số tối ưu.
 - Thời gian train mất khoảng 25 phút (do cấu hình máy yếu có thể nhanh hơn nếu train bằng colab)



```
gamma = None
gamma_val = float('inf')
train_errs = []
val_errs = []
C = [0.001, 0.01, 0.1, 1, 3, 5, 10]
G = ['scale', 'auto']
#Train and save model
for c in C:
    for gamma in G:
        model = svm.SVC(kernel='rbf', C = c, gamma=gamma)
        model.fit(X_train,y_train)
        train_errs.append(model.score(X_train,y_train))
        val_errs.append(model.score(X_val, y_val))
        if val_errs[-1] == max(val_errs):
            best_C = c
            best_gamma = gamma
            best_val = val_errs[-1]
```

all time: 25min 54s

2. Trực quan kết quả

- Tham số C khi huấn luyện ở tập train sẽ có sự tăng dần khi C tăng dần nhưng tập val sẽ tăng đến 1 ngưỡng rồi giảm xuống vì khi C càng tăng mô hình càng fit vào dữ liệu train sẽ dẫn đến hiện tượng overfitting làm giảm độ chính xác khi transform tập validation.



```
# Huấn luyện lại mô hình trên toàn bộ tập dữ liệu train
# standardize and tokenizer
train_final = train.Text.apply(standardize_data)
train_final = train.Text.apply(tokenizer)

# Embedding
X_train_final = embedding_(train_final)
y_train_final = train.Sentiment

# train and save model
model = svm.SVC(kernel='rbf', C = best_C, gamma=best_gamma)
model.fit(X_train_final, y_train_final)
joblib.dump(model, 'saved_model.pkl')
```

3. Huấn luyện lại mô hình trên toàn bộ tập dữ liệu train

```
['saved_model.pkl']
```

4. Kết quả trên tập test

- Tỷ lệ dự đoán chính xác là 0.831973898858075



5. Đánh giá

- Để đánh giá điện thoại là có đáng mua hay không, nhóm đánh giá theo công thức là:
 - *Điện thoại đáng mua là điện thoại có tỉ lệ comment tốt trên 80%.*



VI. KÌ VỌNG TRONG TƯƠNG LAI



Tập dữ liệu này được crawl từ trang thegioididong.com, nhóm đánh giá các comment trên trang web này là trung thực và không có các comment giả để tăng khả năng bán hàng.



Nên khi áp dụng mô hình này vào các comment trên các nền tảng bán hàng online khác như tiki hay lazada thì kết quả sẽ không cho được tỉ lệ dự đoán chính xác vì trên các nền tảng này chứa khá nhiều comment giả để tăng uy tín.



Do đó: **Kì vọng của nhóm là có thể tạo được mô hình học có thể phân biệt comment thật/ comment giả, từ đó có thể áp dụng cho nhiều nền tảng bán hàng.**

VII. NHÌN LẠI QUÁ TRÌNH LÀM ĐỒ ÁN

- **Khó khăn:**

- việc lựa chọn được một dataset phù hợp và đưa ra chủ đề là khá tốn thời gian, chiếm khoảng 2/3 tổng thời gian làm đồ án
- Vì tập data này là khá nhỏ (18018 dòng, 4 cột) nên việc mô hình học đạt tỉ lệ cao trên 90% là khá khó, vì khó khăn áp dụng các thuật toán deep learning

VIII. TÀI LIỆU THAM KHẢO

- Tham khảo các tài liệu từ các notebook có sẵn trên Kaggle
- <https://gate.ac.uk/sale/nle-svm/svm-ie.pdf>
- <http://aurelieherbelot.net/resources/slides/teaching/SVMs.pdf>



CẢM ƠN MỌI
NGƯỜI ĐÃ
THEO DÕI