Lab 5

BÁO CÁO BÀI THỰC HÀNH SỐ 5
PHÂN LỚP VĂN BẢN

Môn học: Học máy thống kê

Giảng viên hướng dẫn	Trần Quốc Khánh
Sinh viên thực hiện	Phạm Lê Thành Phát (21521262)
Mức độ hoàn thành	Hoàn thành
Thời gian thực hiện	04/06/2023 - 06/06/2023

TRẢ LỜI CÁC CÂU HỎI

1. Giới thiệu bài toán.

Bài toán phân tích cảm xúc theo định nghĩa học máy thì là bài toán phân lớp cảm xúc dựa trên văn bản ngôn ngữ tự nhiên. Đầu vào của bài toán là một câu hay một đoạn văn bản, còn đầu ra

là các giá trị xác suất (điểm số) của N lớp cảm xúc mà ta cần xác định.

Đối với bộ dữ liệu dưới dây chúng ta sử dụng là bộ dữ liệu UIT-VSFC, với mong muốn có Input: câu bình luận của người dung và Output: Nhãn cảm xúc, gồm 1 trong 3 nhãn: tích cực, tiêu cực và trung tính.

2. Phân tích sơ bộ trên bộ dữ liệu.

Ở đây, sau khi nhập dữ liệu và sử dụng 1 số câu lệnh kiểm tra, ta thấy được:

a) Tổng các câu trong bộ dữ liệu: khoảng 15000

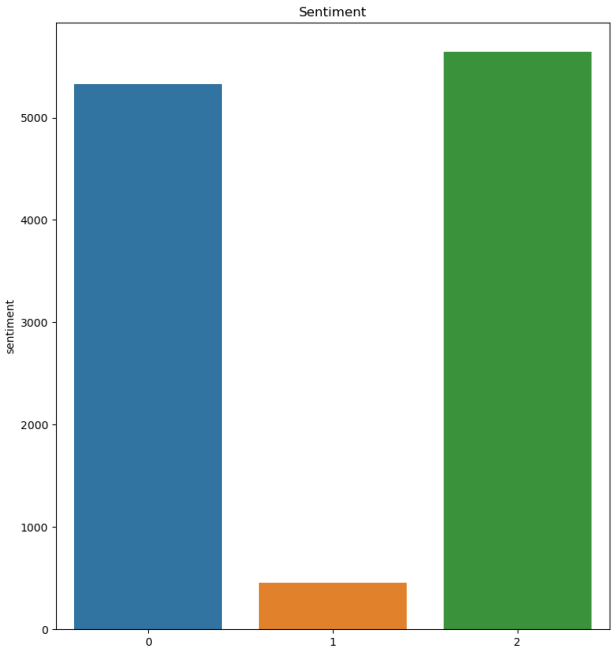
```
df_train = pd.concat([X_train, y_train], axis=1)
df_test = pd.concat([X_test, y_test], axis=1)
df_dev = pd.concat([X_dev, y_dev], axis=1)

print('Size of train: ', df_train.shape)
print('Size of test: ', df_test.shape)
print('Size of dev: ', df_dev.shape)
```

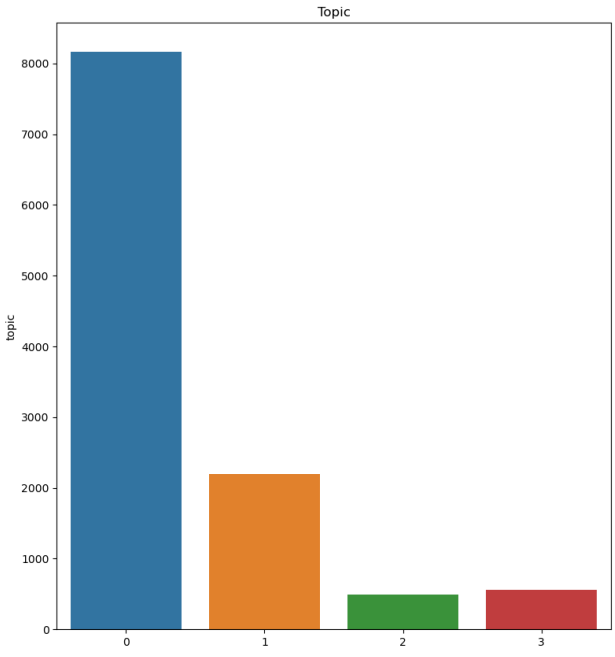
[6] ✓ 0.0s Python

```
Size of train:  (11426, 3)
Size of test:  (3166, 3)
Size of dev:  (1583, 3)
```

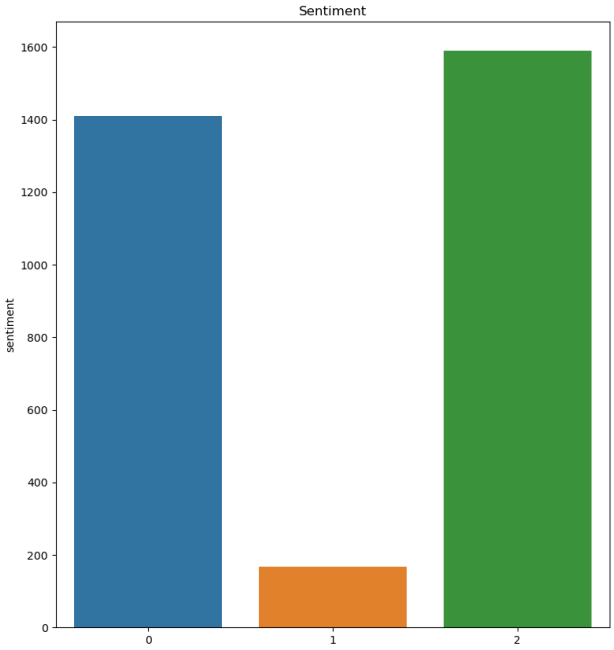
b) Sự phân bố nhãn

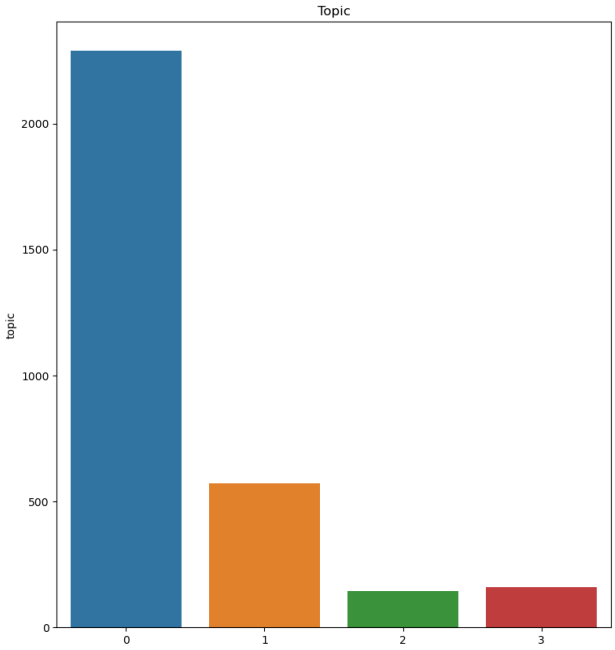
- Tập train:



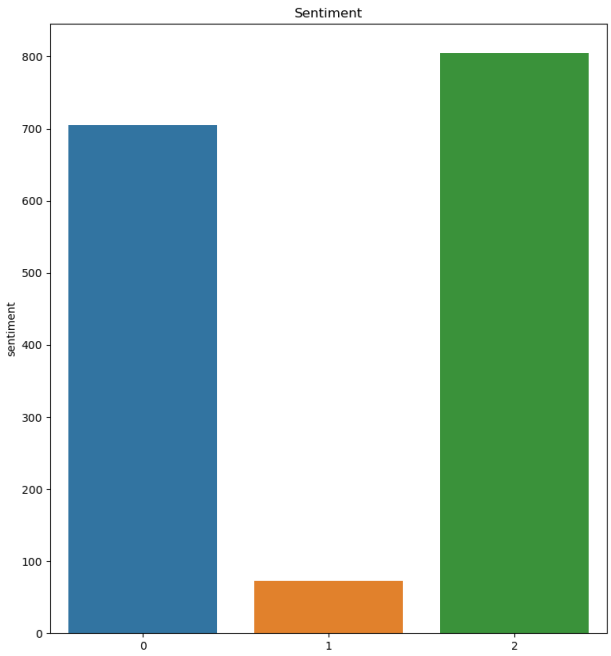


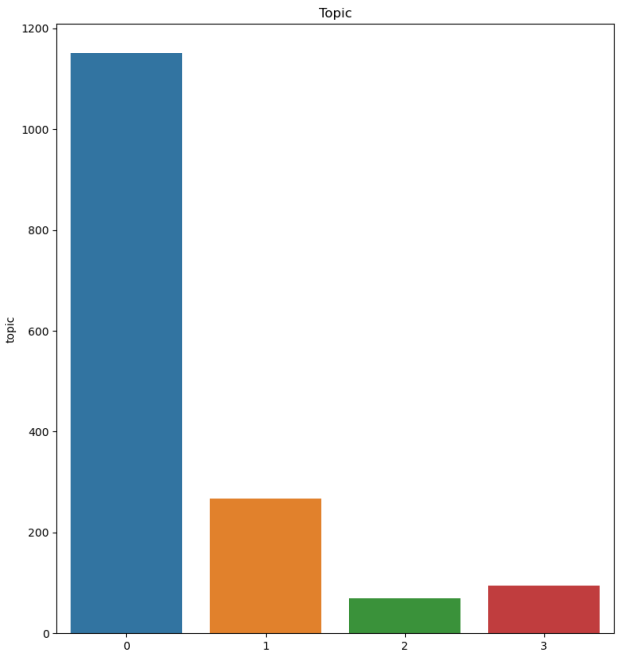
- Tập test





- Tập dev





3. Mô hình áp dụng.

Sử dụng mô hình Logistic Regression và SVM, đây là 2 mô hình cơ bản của học máy. Quá trình huấn luyện mô hình Logistic Regression thường dựa trên phương pháp tối đa hóa hàm likelihood, thông qua các phương pháp tối ưu hóa như Gradient Descent. Mục tiêu là tìm ra các hệ số tối ưu (weights) cho biến đầu vào để mô hình có khả năng dự đoán chính xác. Mô hình Logistic Regression rất phổ biến và được sử dụng rộng rãi trong nhiều lĩnh vực, chẳng hạn như phân loại email spam, dự đoán khả năng mắc bệnh, xác định khả năng chấp nhận của khách hàng, v.v. Còn SVM là mô hình chuyên dụng cho bài toán phân loại 2 lớp, quá

trình huấn luyện mô hình SVM thường bao gồm việc tối ưu hóa một hàm mất mát, như hàm mất mát hinge, nhằm tìm ra siêu phẳng tối ưu. Có nhiều phương pháp tối ưu hóa được sử dụng để giải quyết bài toán tối ưu SVM, bao gồm cả phương pháp tìm kiếm gradient (gradient search) và phương pháp lập trình tuyến tính (linear programming). Mô hình SVM có ưu điểm là độ tin cậy cao trong dự đoán và khả năng xử lý tốt với các bộ dữ liệu lớn. Nó cũng có khả năng xử lý các bài toán phân loại phi tuyến thông qua sử dụng các hàm kernel.

4. Kết quả đạt được.

a) Logistic Regression

Về điểm số (khi chưa tối ưu):

```
accuracy_score:  0.8840808591282375
precision_score:  0.7574990290782253
recall_score:  0.6861116062575138
f1_score:  0.7041924971509287
```

Sau khi sử dụng TfidfVectorizer thay cho CountVectorizer:

```
logreg_tfidf accuracy_score:  0.8809222994314593
logreg_tfidf precision_score:  0.7915430551516596
logreg_tfidf recall_score:  0.6457385241568766
logreg_tfidf f1_score:  0.6514111568382064
```

Sau khi được tách từ và sử dụng TfidfVectorizer:

```
logreg_underthesea accuracy_score:  0.885028427037271
logreg_underthesea precision_score:  0.8076321858498366
logreg_underthesea recall_score:  0.6468660422324822
logreg_underthesea f1_score:  0.6514107474844565
```

Nhận xét: Không có sự thay đổi nhiều về khả năng dự đoán, thậm chí còn bị giảm so với lúc chưa được tối ưu. Cho thấy rằng mô hình Logistic Regression là một mô hình khó tối ưu bằng cách thông thường, tối ưu không chuẩn có thể làm giảm hiệu suất cũng như gây tiêu tốn tài nguyên.

b) SVM

Về điểm số (khi chưa tối ưu):

```
accuracy_score:  0.8859759949463045
precision_score:  0.8134670819952211
recall_score:  0.6388055283302992
f1_score:  0.6358368786162855
```

Sau khi sử dụng TfidfVectorizer thay cho CountVectorizer:

```
svm_tfidf accuracy_score:  0.8897662665824384
svm_tfidf precision_score:  0.8769945392300164
svm_tfidf recall_score:  0.6576409081752131
svm_tfidf f1_score:  0.6680936999205618
```

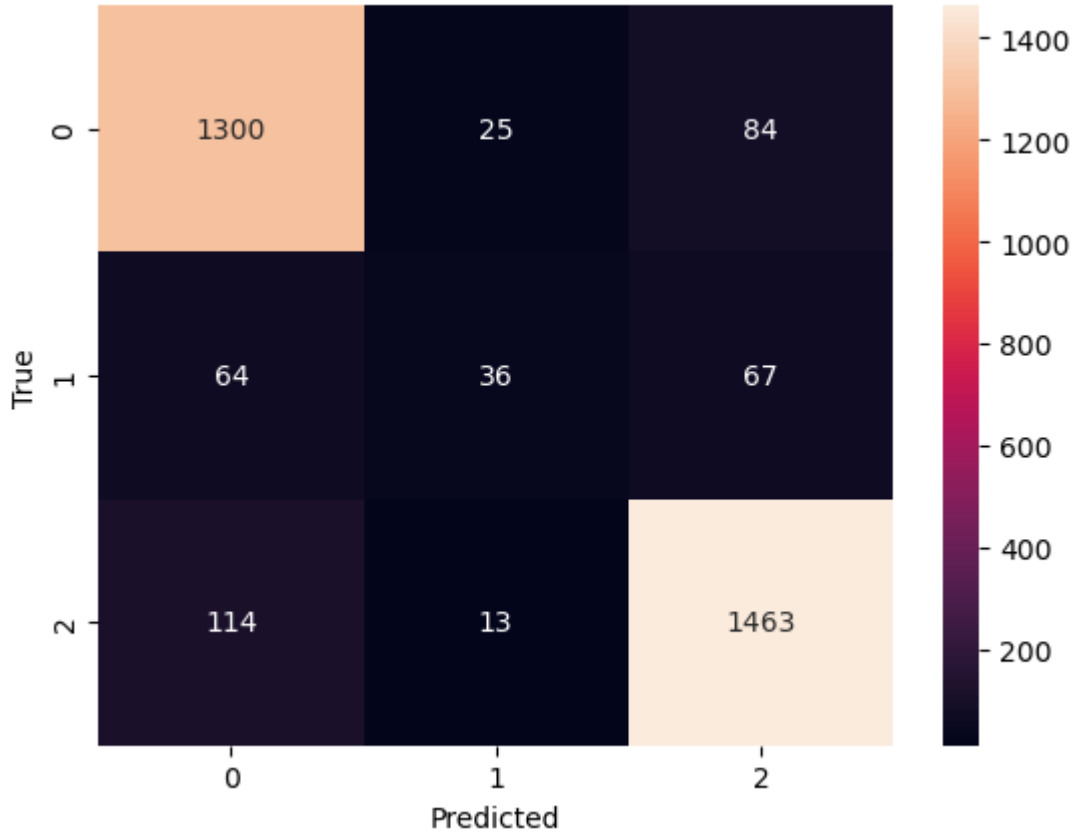
Sau khi được tách từ và sử dụng TfidfVectorizer:

```
svm_underthesea accuracy_score:  0.8948199620972836
svm_underthesea precision_score:  0.8604426872047132
svm_underthesea recall_score:  0.6577187158773855
svm_underthesea f1_score:  0.6650767954566609
```

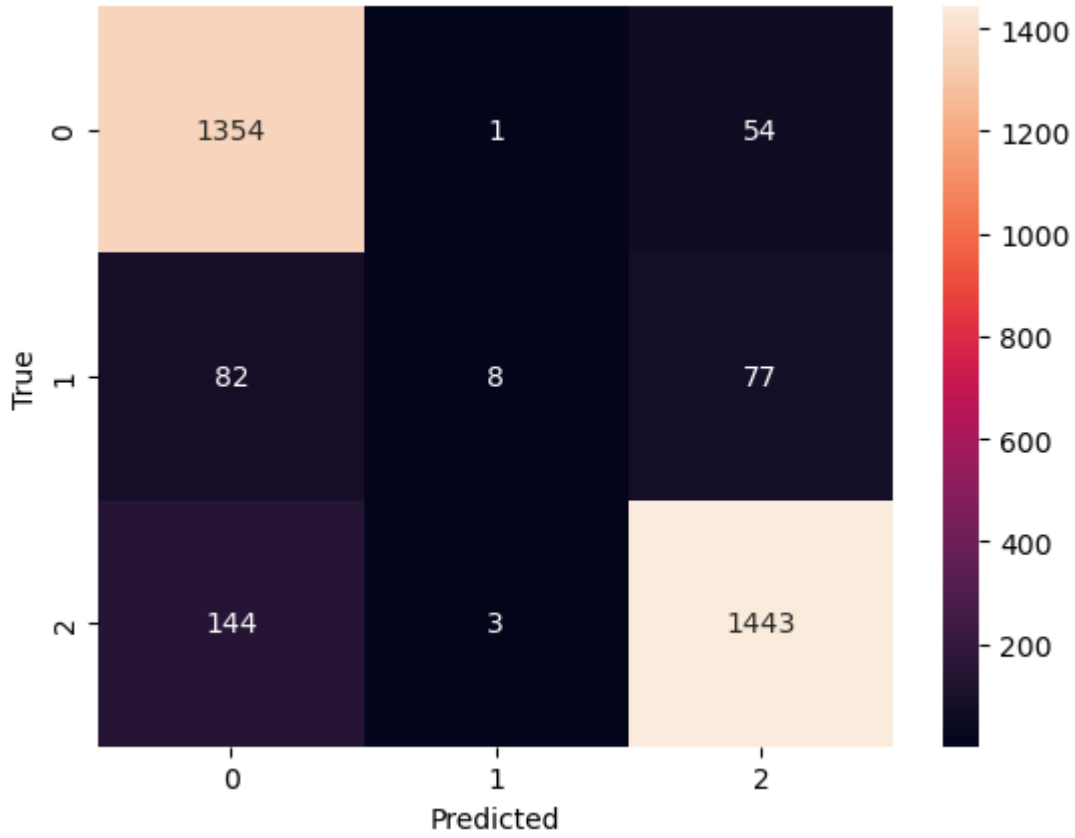
Nhận xét: đối với SVM, việc tối ưu có vẻ mang lại một chút hiệu suất với tất cả các điểm số đều tang, tuy hiên không nhiều. Dù vậy khi chạy SVM máy phải hoạt động khá vất vả, không nhẹ nhàng như Logistic Regression. Đối với một hệ thống dữ liệu nhỏ điều này có thể không thành vấn đề nhưng với những bộ dữ liệu lớn lại là một trở ngại khá lớn so với Logistic Regression, do hiệu suất sau khi tối ưu không cao hơn nhiều nhưng lại tiêu tốn tài nguyên gấp vài lần so với Logistic Regression.

c) Khả năng dự đoán

Logistic Regression



SVM



Chúng ta có thể thấy ở label 1 bị dự đoán nhầm nhiều nhất, ngược lại với label 2 thì dự đoán có phần chính xác hơn. Xét thấy mô hình SVM phân loại label 1 không tốt, nên ta nhận định

SVM có khả năng phân loại nhị phân tốt, và nếu so với LR thì SVM có phần tốt hơn. Về LR thì mô hình này có khả năng phân loại 3 lớp tốt hơn so với SVM một chút tuy nhiên không nhiều.

5. Kết luận và hướng phát triển.

Kết luận: Sử dụng model Logistic Regression sẽ tối ưu hơn về mặt thời gian cũng như tài nguyên, tuy nhiên hiệu suất lại không quá ấn tượng và khó có thể tối ưu. Ngoài ra cần chỉnh sửa lại bộ dữ liệu về một số câu không mang ý nghĩa, khiến máy học không chuẩn dẫn đến việc dự đoán sai.

Về hướng phát triển: em nghĩ có thể tính toán lại và sử dụng nhiều giá trị của siêu tham số để thử nghiệm từ đó đưa ra kết quả tốt hơn. Ngoài ra quan trọng nhất là bộ dữ liệu phải được lựa chọn sạch sẽ, mang đầy đủ tính chất của câu và mang ý nghĩa hình thái. Trong tương lai em sẽ sử dụng nhiều mô hình khác liên quan đến không chỉ Machine Learning mà còn liên quan đến Deep Learning hoặc AI.