I. Giới thiệu bài toán

1.1. Mô tả bài toán

Bài toán phân loại các sản phẩm vào các nhãn dựa trên title của chúng.

- Đầu vào: tập dữ liệu chứa title sản phẩm đã được gán nhãn
- Đầu ra: một mô hình phân loại có khả năng phân loại đủ tốt giữa các nhãn.

1.2. Dữ liệu

Dữ liệu dùng cho bài toán gồm 2 file txt:

- Train.txt: gồm 91868 sản phẩm với title đã được gán nhãn. Tập nhãn gồm 7 labels: __label__1.0, __label__12.0, __label__14.0, __label__16.0, __label__24.0, __label__9.0, __label__11.0.
- Test.txt: gồm 19928 sản phẩm kèm nhãn tương ứng. Tập nhãn cũng gồm 7 nhãn như tập train.txt.

II. Lý thuyết

2.1. Đưa text về dạng số

2.1.1. CountVectorizer

Biến đổi văn bản từ dạng thô sang một vector thưa dựa trên số lần xuất hiện của các từ trong từ vựng.

Số chiều của vector bằng kích thước của từ vựng.

2.1.2. TfidfVectorizer

Khác với CountVectorizer ở chỗ, TfidfVectorizer tính toán lại trọng số cho từng feature là các từ trong từ vựng, các từ xuất hiện quá nhiều lần trong nhiều văn bản thường mang ít ý nghĩa và vì vậy nó được đánh trọng số nhỏ hơn. Trọng số này được tính dựa trên tf (term frequency) và idf (inverse document frequency):

- $tf(t,d)$ = số lần xuất hiện của từ t trong document d.
- $idf(t) = log\frac{n}{df(t)} + 1$, với $df(t)$ là số document chứa t trong toàn bộ tập documents.
- $if\_idf(t) = if(t,d).idf(t)$

Sau khi thu được vector tfidf cho document đó, chuẩn hóa nó bằng norm 2.

2.2. Một số mô hình phân loại

2.2.1. Random forest classifier

Là một thuật toán thuộc loại ensemble learning được tối ưu cho thuật toán cây quyết định. Thuật toán xây dựng một tập các cây quyết định có thể dựa trên phương pháp bagging, random patches hoặc random subspaces để tăng tính đa dạng cho thuật toán, giúp cho thuật toán có thể hoạt động tốt hơn, cải thiện kết quả.

Với số lượng cây đủ lớn, rừng ngẫu nhiên có khả năng tránh được overfit. Số lượng cây càng lớn, thuật toán hoạt động càng tốt, nhưng sẽ chạy chậm hơn.

2.2.2. Logistic regression

Là một thuật toán hồi quy thường được dùng cho bài toán phân loại. Hồi quy logistic tính toán khả năng một mẫu thuộc một nhãn cụ thể.

❖ Đối với phân lớp nhị phân:

Sử dụng hàm sigmoid làm hàm kích hoạt để tính xác xuất để thể hiện $\boldsymbol{x}$ rơi vào class 1.

- Hàm dự đoán:

$$\hat{p} = h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sigma(\boldsymbol{x}^T\boldsymbol{\theta})$$

$$\sigma(t) = \frac{1}{1+e^{-t}}$$

Khi đó:

$$\hat{y} = \begin{cases} 0 & nếu\ \hat{p} < 0.5 \\ 1 & nếu\ \hat{p} \geq 0.5 \end{cases}$$

- Hàm mất mát:

$$\boldsymbol{J}(\boldsymbol{\theta}) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}\log\left(\hat{p}^{(i)}\right) + (1-y^{(i)})\log(1-\hat{p}^{(i)})]$$

Mục tiêu là tìm $\boldsymbol{\theta}$ để cực tiểu hóa hàm mất mát. Ở đây, $\boldsymbol{J}(\boldsymbol{\theta})$ (gọi là hàm log loss) là một hàm lồi nên có thể sử dụng gradient decent để tìm cực trị.

❖ Đối với phân loại nhiều lớp:

Sử dụng hàm softmax thay cho hàm sigmoid.

- Hàm dự đoán:

$$\hat{p}_k = \sigma(\boldsymbol{x}^T\boldsymbol{\Theta})_k = \frac{e^{\boldsymbol{x}^T\boldsymbol{\theta}^{(k)}}}{\sum_{j=1}^{K} e^{\boldsymbol{x}^T\boldsymbol{\theta}^{(j)}}}$$

Trong đó, $\boldsymbol{\theta}^{(k)}$ tương ứng được sử dụng để tính xác suất thể hiện $\boldsymbol{x}$ thuộc vào lớp k.

K là số lớp cần phân loại. $\boldsymbol{\Theta}$ là ma trân tham số, với mỗi hàng lưu giá trị tương ứng $\boldsymbol{\theta}$.

$$\hat{y} = arg\max_{k} \hat{p}_k$$

- Hàm mất mát:

$$\boldsymbol{J}(\boldsymbol{\Theta}) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)}\log\left(\hat{p}_k^{(i)}\right)$$

$y_k^{(i)}$ ($\in \{0,1\}$) có ý nghĩa: thể hiện $\boldsymbol{x}^{(i)}$ có thuộc vào lớp k không**.**

Tìm $\boldsymbol{\Theta}$ để cực tiểu hóa $\boldsymbol{J}(\boldsymbol{\Theta})$ bằng Gradient decent.

2.2.3. Naive bayes

Thuật toán được xây dựng dựa trên định lý Bayes với sự giả định mạnh mẽ về sự độc lập giữa mọi cặp đặc trưng của mẫu.

Cho thể hiện $\boldsymbol{x}$ và phân lớp y:

$$P(y|\boldsymbol{x}) = P(y|x_1, x_2, \dots x_n) = \frac{P(\boldsymbol{x}|y).P(y)}{P(\boldsymbol{x})}$$
$$= \frac{P(x_1, x_2, \dots x_n|y).P(y)}{P(\boldsymbol{x})}$$
$$= \frac{P(y).\prod_{i=1}^{n} P(x_i|y)}{P(\boldsymbol{x})}$$

(theo giả định về tính độc lập giữa các feature của $\boldsymbol{x}$)

$P(\boldsymbol{x})$ là hằng số tính được từ dữ liệu đầu vào

$$\Longrightarrow P(y|\boldsymbol{x}) \propto P(y).\prod_{i=1}^{n} P(x_i|y)$$

$$\Longrightarrow \hat{y} = arg\max_{y} P(y).\prod_{i=1}^{n} P(x_i|y)$$

2.3. Độ đo đánh giá

- Accuracy: là độ đo đơn giản được sử dụng trong các mô hình phân loại, được tính bằng số dự đoán đúng trên toàn bộ tập kiểm thử. Nhưng đối với bộ dữ liệu mất cân bằng thì độ chính xác không mang ý nghĩa và không được khuyên dùng.
- Confusion matrix: sử dụng tập nhãn dự đoán so sánh với tập nhãn đúng đưa ra các thống kê về số lượng mẫu dự đoán đúng và sai trên từng nhãn:

		Predict	
		Positive	Negative
Actual	Positive	TP	FN
	Negative	FP	TN

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

Precision trên 1 lớp cao thể hiện độ chính xác trên các dự đoán mẫu thuộc lớp này cao; recall trên 1 lớp cao thể hiện trong tất cả các mẫu của lớp này thì tỉ lệ dự đoán đúng là cao. Như vậy, một mô hình tốt khi cả precision và recall đều cao. Kết hợp precision và recall tạo ra một độ đo duy nhất để đánh giá gọi là F1_score:

$$F1\_score = \frac{2 * Precision * Recall}{Precision + Recall}$$

F1_score cao khi cả Precision và Recall cao.

Tuy nhiên F1_score cũng không phải là tất cả những gì mà ta hướng tới, tùy vào từng mục đích của bài toán, phải có chiến lược hợp lý trong Precision/Recall traddeoff.

III. Thực nghiệm

3.1. EDA và tiền xử lý dữ liệu

- Tập dữ liệu bao gồm 2 cột, một cột chứa thông tin về nhãn, một cột chứa title của sản phẩm.

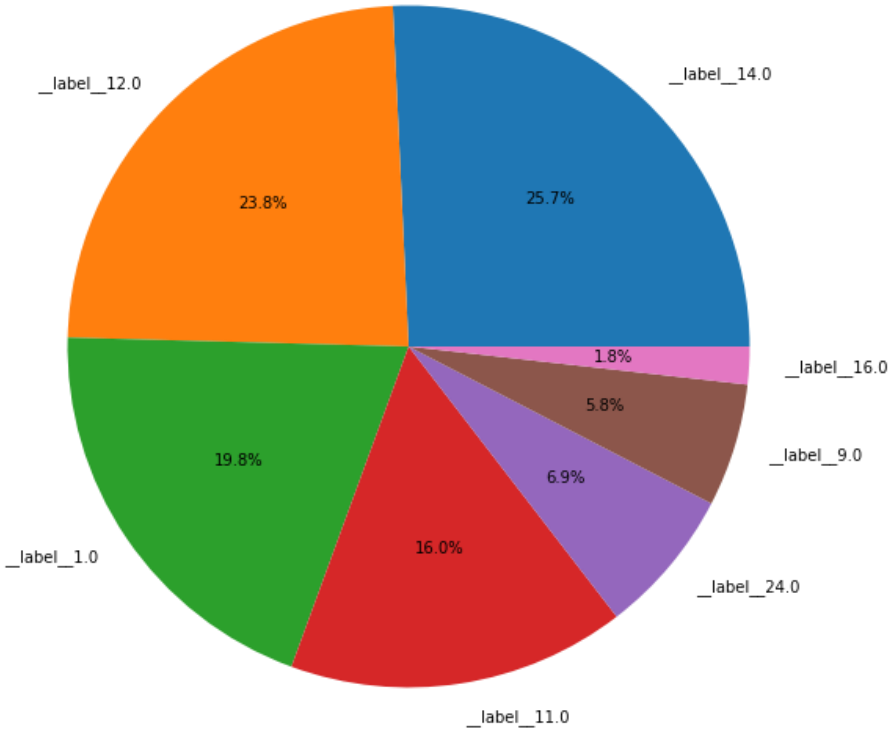
```
__label__1.0 cáp chuyển pd
__label__1.0 tai nghe h
__label__12.0 áo sơ mi namu
__label__1.0 đồng hồ fasfit
__label__24.0 mô hình clearence anna lego disney
__label__11.0 mascara australis
__label__24.0 mô hình lego nexo knights ba anh em clearence
__label__24.0 đồ chơi cầu trượt vivitoys
__label__11.0 thực phẩm bảo vệ sức khỏe nature made
__label__14.0 cây đá serpentine
__label__1.0 bộ tua pin
__label__11.0 mascara lót mascara
```

- Các bước làm sạch dữ liệu bao gồm xóa bỏ các kí tự số và kí tự đặc biệt, tách từ sau đó loại bỏ các từ không có nghĩa (một số từ như tm, dk, mmnf,… các từ chỉ chứa 1 chữ cái là phụ âm)
- Kiểm tra trùng lặp dữ liệu và loại bỏ trùng lặp, dữ liệu sau loại bỏ còn 87502/91868 mẫu.
- Kiểm tra tính đúng đắn của dữ liệu: một số title được gán nhiều nhãn:

	label	processed
7534	__label__1.0	đèn pin
25489	__label__14.0	đèn pin
86483	__label__16.0	đèn pin

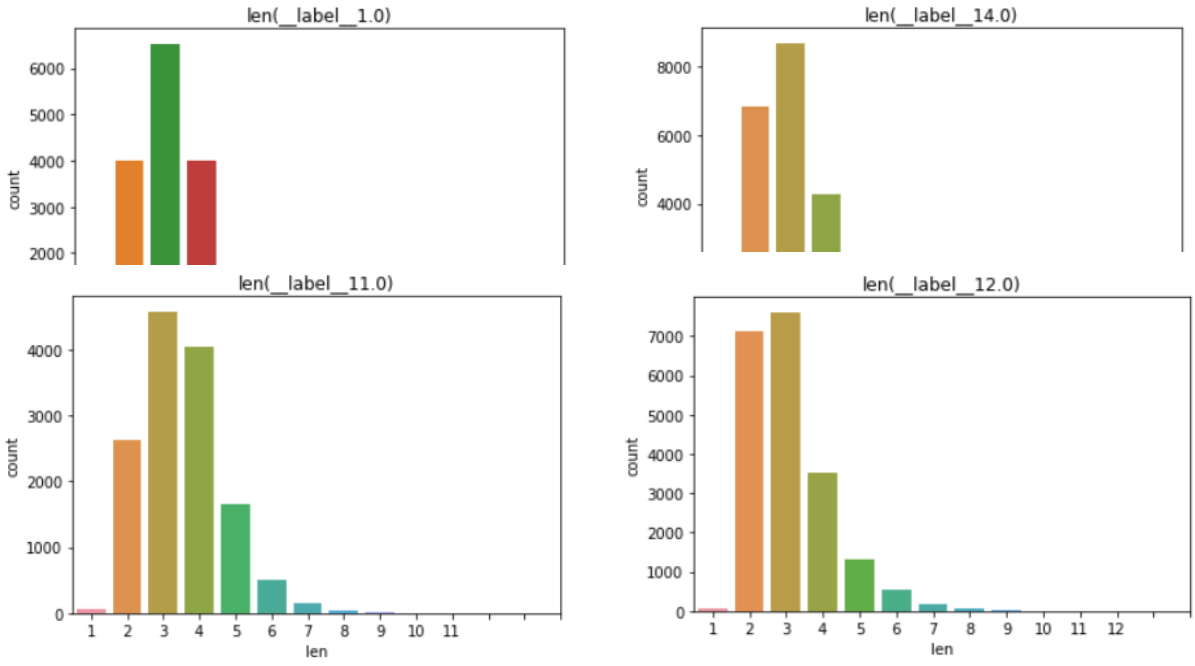
Các mẫu này cần được loại bỏ. sau khi loại bỏ, dữ liệu còn lại: 85590 mẫu.

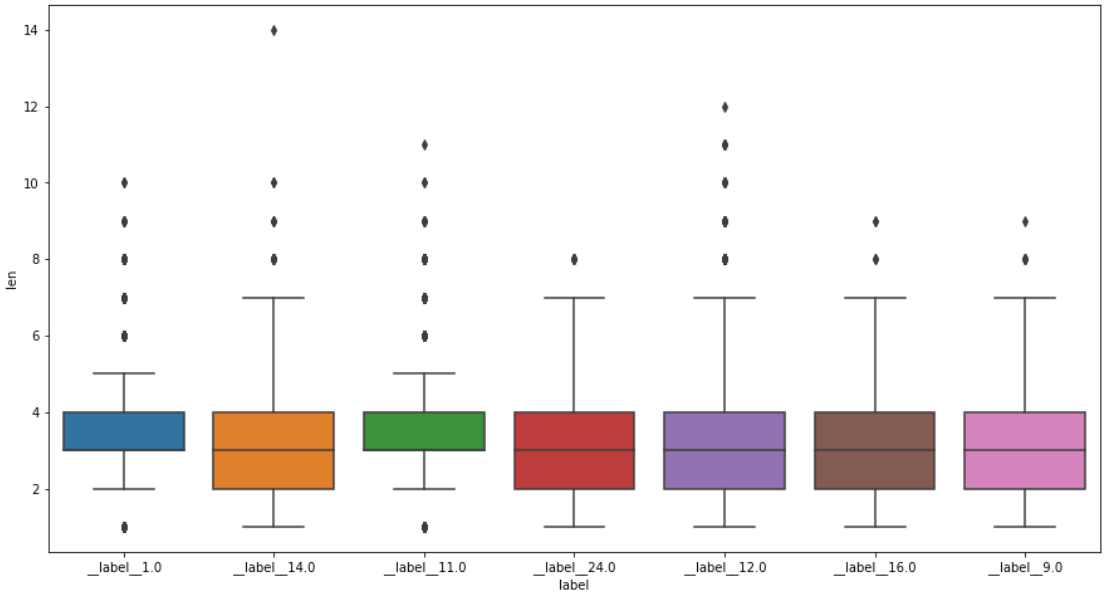
- Số lượng nhãn gồm 7 nhãn và phân phối nhãn như sau:



⇨ Bộ dữ liệu lệch nhiều về các nhãn 12, 14, 1; trong khi đó, các nhãn 16, 19, 24 thì có rất ít mẫu.

- Các title ngắn, rơi nhiều nhất vào 2-5 từ. Phân phối độ dài title của các nhãn có sự tương đồng nhau. Một số phân phối trên các nhãn:





⇨ Loại bỏ các title quá ngắn hoặc quá dài ngưỡng lấy là [2, 8), số lượng mẫu còn lại: 84854 mẫu.

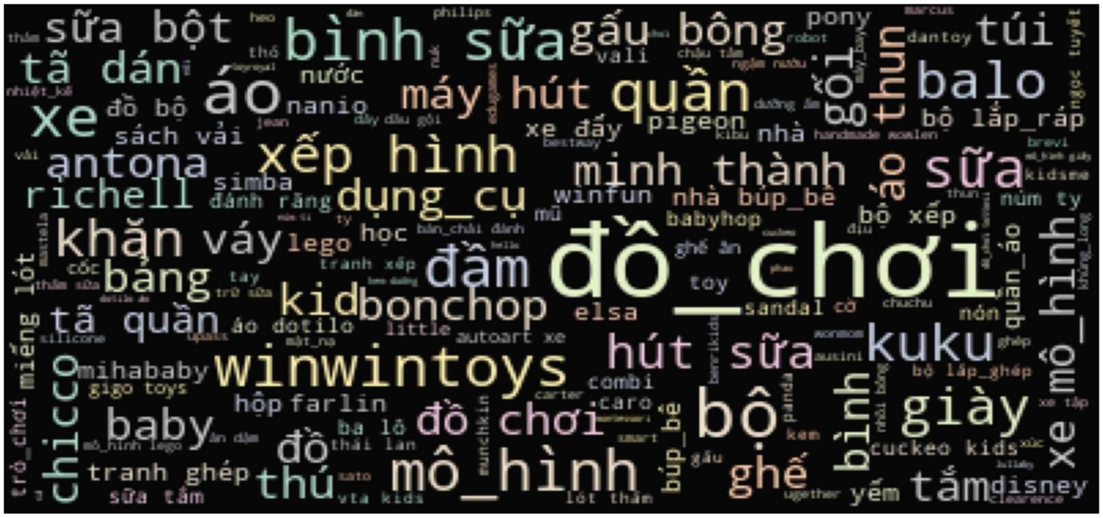
- Trực quan hóa dữ liệu ứng với từng nhãn để quan sát các từ khóa được sử dụng nhiều:

__label__1.0 train

__label__14.0 train

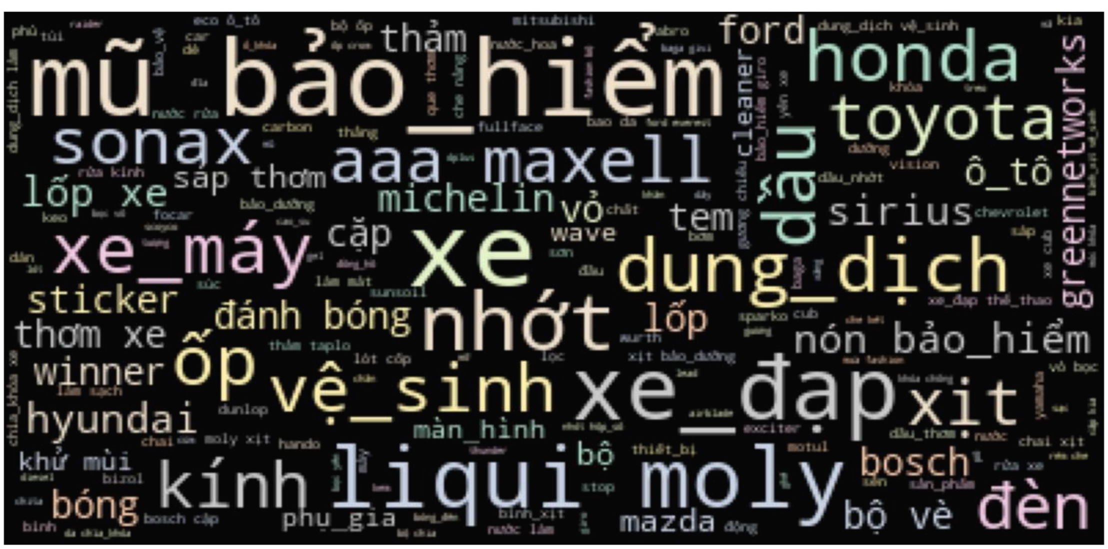
__label__11.0 train

__label__24.0   train



__label__12.0   train

__label__16.0   train



__label__9.0 train

Có thể thấy nhãn 24 và nhãn 12 có sự trùng lặp khá nhiều về các từ khóa. Thống kê trên 200 từ khóa xuất hiện nhiều nhất ở 2 nhãn này thì số từ khóa trùng lặp là 25 từ:

```
{'balo', 'disney', 'váy', 'khăn', 'giày thểthao', 'khóa', 'quần',
'áo dotilo', 'mũ', 'dây', 'dép', 'áo', 'dotilo áo', 'đầm', 'ba lô'
, 'caro', 'túi', 'áo thun', 'sandal', 'dài', 'jean', 'nón', 'giày'
, 'bộ', 'vali'}
```

- Đưa dữ liệu từ text sang số sử dụng TfidfVectorizer với ngram_range=(1, 3), khi đó, mỗi title sẽ được biểu diễn dưới dạng 1 vector 21685 chiều.
- Dữ liệu nhãn được đưa về dạng số sử dụng LabelEncoder, mỗi nhãn sẽ tương ứng với 1 số từ 0-6

  ⇨ Dữ liệu đã sẵn sàng để đưa vào mô hình.

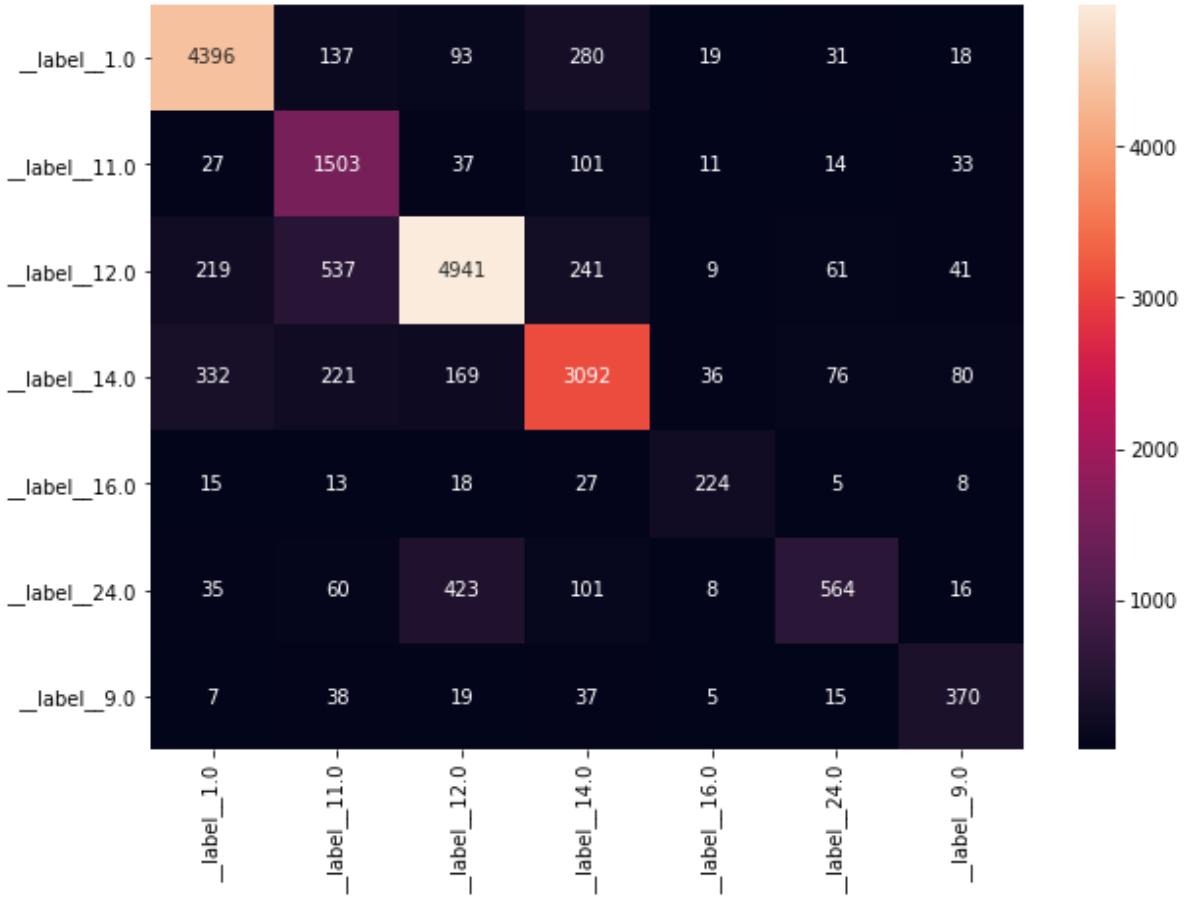
3.2. Modeling and fine tune

- Với thuật toán RandomForest, kết quả thu được trên tập test:

	precision	recall	f1-score	support
__label__1.0	0.87	0.88	0.88	4974
__label__11.0	0.60	0.87	0.71	1726
__label__12.0	0.87	0.82	0.84	6049
__label__14.0	0.80	0.77	0.78	4006
__label__16.0	0.72	0.72	0.72	310
__label__24.0	0.74	0.47	0.57	1207
__label__9.0	0.65	0.75	0.70	491
accuracy			0.80	18763
macro avg	0.75	0.76	0.74	18763
weighted avg	0.81	0.80	0.80	18763

F1_score=0.74

Xem xét confusion matrix:

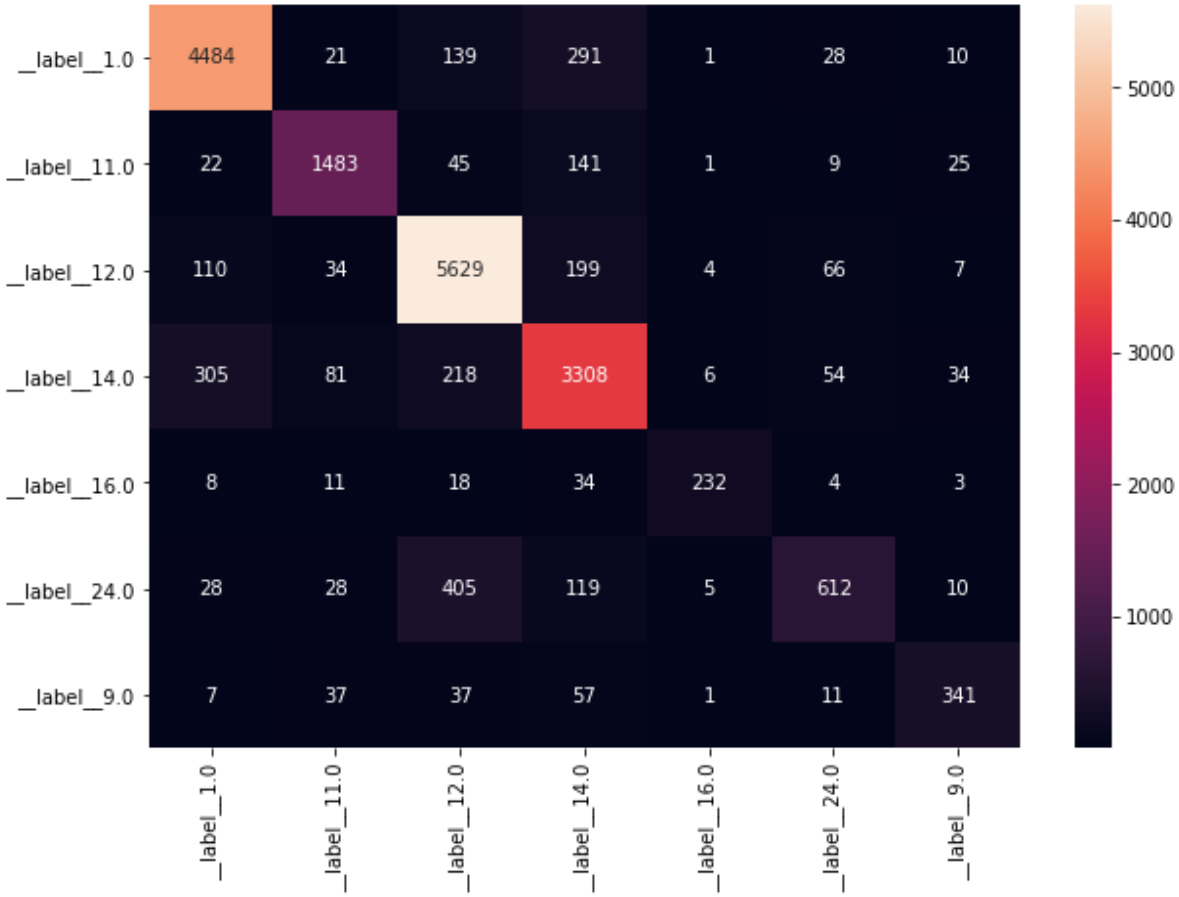


Nhầm lẫn xảy ra nhiều ở các nhãn có số lượng mẫu ít. Trong đó, dự đoán nhầm nhãn 24 thành nhãn 12 chiếm gần nửa số mẫu mang nhãn 24.

- Naïve bayes cho kết quả khá tốt, nhỉnh hơn so với thuật toán random forest với F1_score=0.81:

	precision	recall	f1-score	support
__label__1.0	0.90	0.90	0.90	4974
__label__11.0	0.87	0.86	0.87	1726
__label__12.0	0.87	0.93	0.90	6049
__label__14.0	0.80	0.83	0.81	4006
__label__16.0	0.93	0.75	0.83	310
__label__24.0	0.78	0.51	0.61	1207
__label__9.0	0.79	0.69	0.74	491
accuracy			0.86	18763
macro avg	0.85	0.78	0.81	18763
weighted avg	0.86	0.86	0.85	18763

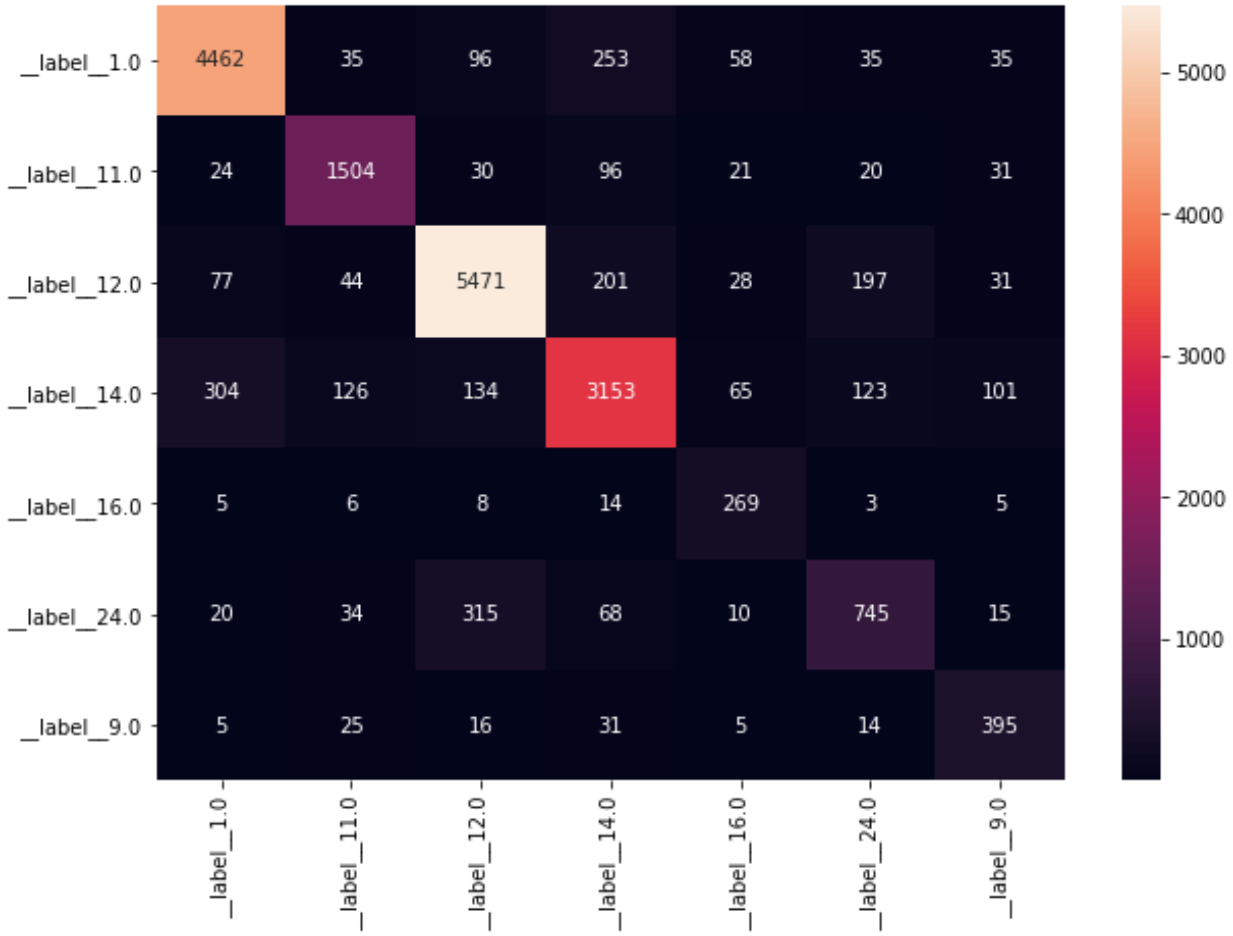
nhầm lẫn giữa 2 nhãn 24 và 12 có giảm một chút, không đáng kể:



- Một mô hình khác được sử dụng là logistic regression: thu được F1_score=0.79

	precision	recall	f1-score	support
__label__1.0	0.91	0.90	0.90	4974
__label__11.0	0.85	0.87	0.86	1726
__label__12.0	0.90	0.90	0.90	6049
__label__14.0	0.83	0.79	0.81	4006
__label__16.0	0.59	0.87	0.70	310
__label__24.0	0.66	0.62	0.64	1207
__label__9.0	0.64	0.80	0.72	491
accuracy			0.85	18763
macro avg	0.77	0.82	0.79	18763
weighted avg	0.86	0.85	0.85	18763

Confusion matrix:

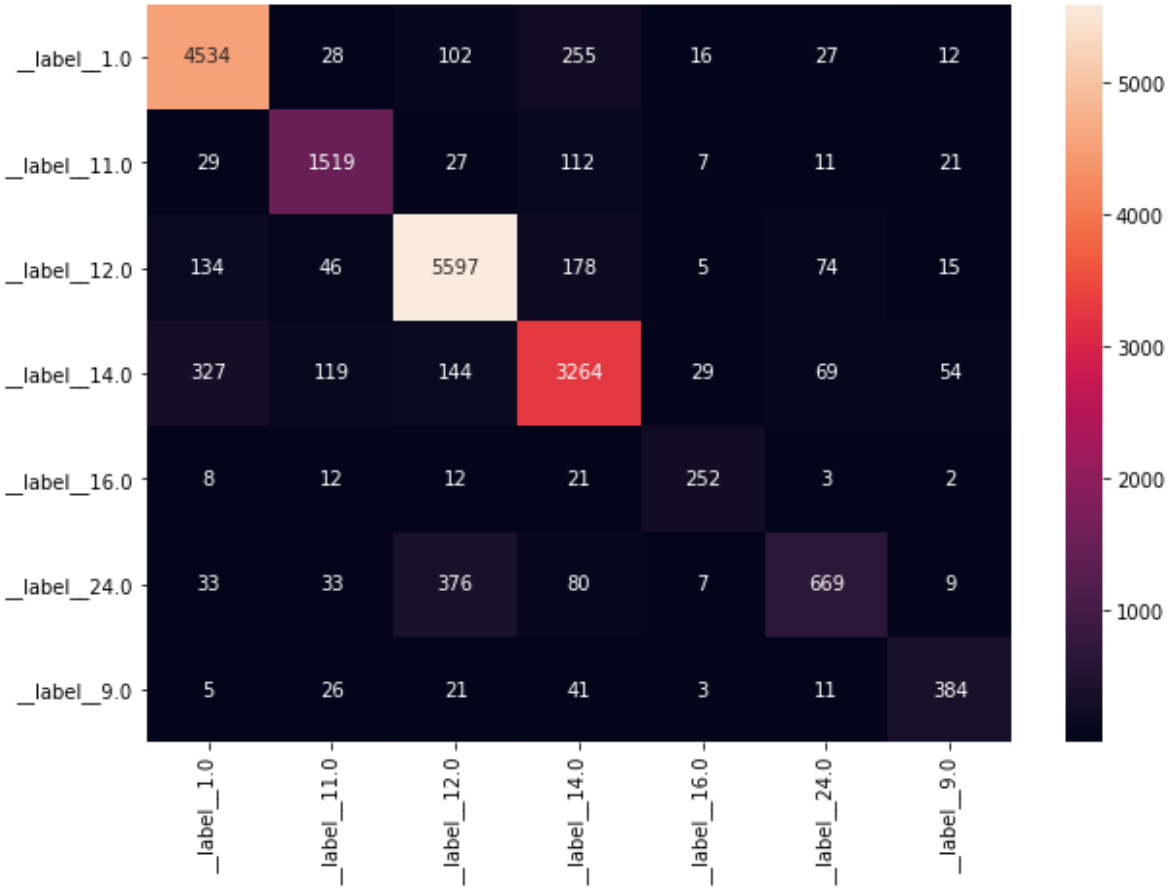


⇨ Mô hình tốt nhất cho bài toán là Naïve bayes với F1_score=0.81.

- Thử sử dụng Voting classifiers với 2 thuật toán trên để xem liệu kết quả có cải thiện không. Kết quả thu được F1_score=0.82 hơn naïve bayes 0.01.

	precision	recall	f1-score	support
__label__1.0	0.89	0.91	0.90	4974
__label__11.0	0.85	0.88	0.87	1726
__label__12.0	0.89	0.93	0.91	6049
__label__14.0	0.83	0.81	0.82	4006
__label__16.0	0.79	0.81	0.80	310
__label__24.0	0.77	0.55	0.65	1207
__label__9.0	0.77	0.78	0.78	491
accuracy			0.86	18763
macro avg	0.83	0.81	0.82	18763
weighted avg	0.86	0.86	0.86	18763

Confusion matrix:



⇨ Mô hình thu được kết quả tốt nhất khi dùng Voting Classifiers với F1_score=0.82.
Kết quả phân loại giữa nhãn 24 và 12 còn nhiều nhầm lẫn. Một số nhãn bị predict nhầm:

	label	processed
8192	__label__24.0	túi vải moden
734	__label__24.0	ba lô canvas sesame
3972	__label__24.0	áo thun winfa
7824	__label__24.0	áo thun kidgirl
6804	__label__24.0	giày love papamama

Các từ khóa được dùng nhiều trong các title này thuộc vào tập giao nhau của 2 nhãn và số lượng mẫu của nhãn 12 được học gấp 4 lần số lượng mẫu của nhãn 24 nên khả năng mô hình học và dự đoán nhãn 12 sẽ cao hơn.

Nhầm nhãn 24 thành nhãn 12