**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN HÀ NỘI**

**KHOA TOÁN – CƠ – TIN HỌC**

**----\*\*\*----**



**BÀI BÁO CÁO**

***Đề tài: Tạo web phỏng đoán tin tức thật giả bằng Python***

***Giảng viên hướng dẫn: PGS.TS. Lê Trọng Vĩnh***

***Nhóm sinh viên thực hiện : Nhóm 5***

***Lớp học phần: Lập trình nâng cao***

***Hà Nội, tháng 11/2022.***

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN HÀ NỘI**

**KHOA TOÁN – CƠ – TIN HỌC**

**----\*\*\*----**



**BÀI BÁO CÁO**

***Đề tài: Tạo web dự đoán tin tức thật giả bằng Python***

**Thành viên nhóm 5:**

1. Đỗ Thị Hường – 20001929
2. Ngô Thị Thu Hằng – 20001913
3. Nguyễn Thị Thu Hiền – 20001914

# 1. Mô phỏng bài toán và quy trình làm việc:

## 1.1 Mô phỏng bài toán:

Web sẽ giúp người dùng phỏng đoán tin tức được nhập vào theo 2 cách là: Nhập url của trang web chứa tin tức hoặc nhập tin tức thủ công bằng Input Box có trên trang web. Bài toán sẽ được chia nhỏ thành 2 bài toán con: Xử lý dữ liệu để đưa ra phỏng đoán về thông tin được truyền vào và tạo lập trang web với PyScript.

**Phần 1:** Xử lý dữ liệu để đưa ra phỏng đoán về thông tin được truyền vào: Xử lý dữ liệu traning và lưu lại kết quả, từ những traning có được sẽ phỏng đoán dữ liệu chứ tin tức mới nhập được từ người dùng là thật hay giả.

Tập dữ liệu trainning của bài toán được lấy từ <https://www.kaggle.com/c/fake-news/data?select=train.csv>. **train.csv** là một tập dữ liệu training đầy đủ với các thuộc tính sau:

id: id duy nhất cho mỗi tin tức

title: tiêu đề của tin tức

author: tác giả của tin tức

text: văn bản của bài viết, có thể không đầy đủ

label: nhãn đánh dấu bài viết có khả năng không đáng tin cậy

+ 1: không đáng tin cậy

+ 0: đáng tin cậy

**Phần 2:** Xây dựng trang web với PyScript.

**<Viết thêm vào>**

**Phần 3:** Một số phương pháp khác để lấy dữ liệu cần dự đoán.

## 1.2 Quy trình làm việc:

- Chuẩn hóa dữ liệu đầu vào, để tối ưu hóa phân tích dữ liệu.

- Chuyển đổi dữ liệu văn bản thành các vector để máy tính có thể hiểu được.

- Ghi các kết quả bước vừa rồi vào model để phỏng đoán.

- Lưu trữ lại các kết quả training được.

- Tạo lập web để nhận các dữ liệu cần phỏng đoán.

- Chuẩn hóa dữ liệu và chuyển nó thành các vector.

- Truyền nó vào model đã tạo ở các bước trên để model đưa ra phỏng đoán.

- Xây dựng web cho phép nhập nhập dữ liệu và đưa ra phỏng đoán.

- Xử lý để lấy dữ liệu cần dự đoán khi dữ liệu được truyền theo dạng khác.

# 2. Chi tiết thực hiện:

## 2.1 Phần 1: Xử lý dữ liệu để đưa ra phỏng đoán về thông tin được truyền vào:

### 2.1.1: Giới thiệu các thư viện sử dụng:

[Pandas](https://pandas.pydata.org/) cung cấp các chức năng và cấu trúc dữ liệu cấp cao được thiết kế để làm việc với dữ liệu có cấu trúc hoặc dạng bảng nhanh chóng, dễ dàng và rõ ràng. Nó cung cấp chức năng lập chỉ mục để giúp dễ dàng định hình lại, cắt, tổng hợp và chọn tập hợp con dữ liệu.

[NLTK](https://www.nltk.org/) (Bộ công cụ ngôn ngữ tự nhiên - Natural Language Toolkit): là một nền tảng hàng đầu để xây dựng các chương trình Python hoạt động với dữ liệu ngôn ngữ tự nhiên. Nó cung cấp các giao diện dễ sử dụng cho hơn 50 kho ngữ liệu và tài nguyên từ vựng như WordNet, cùng với một bộ thư viện xử lý văn bản để phân loại, thông báo, tạo gốc, gắn thẻ, phân tích cú pháp và lý luận ngữ nghĩa...Các thư viện con sử dụng:

+ [NLTK corpus readers(nltk.corpus)](https://www.nltk.org/_modules/nltk/stem/porter.html): Các mô-đun trong gói này cung cấp các chức năng có thể được sử dụng để đọc các tệp văn bản ở nhiều định dạng khác nhau. Các chức năng này có thể được sử dụng để đọc cả tệp kho văn bản được phân phối trong gói kho văn bản NLTK và các tệp văn bản là một phần của kho văn bản bên ngoài.

+ [nltk.stem.porter](https://www.nltk.org/_modules/nltk/stem/porter.html): Loại bỏ các phụ tố hình thái từ các từ, chỉ để lại từ gốc.

[Scikit-learn (sklearn)](https://scikit-learn.org/stable/) là một mô-đun Python dành cho máy học được xây dựng dựa trên SciPy. Các thư viên con đã sử dụng:

+ [sklearn.feature\_extraction:](https://scikit-learn.org/stable/modules/feature_extraction.html) mô-đun này có thể được sử dụng để trích xuất các tính năng ở định dạng được thuật toán máy học hỗ trợ từ các bộ dữ liệu bao gồm các định dạng như văn bản và hình ảnh. Chương trình này sử dụng định dạng hình ảnh, nên mô-đun sử dụng sẽ là sklearn.feature\_extraction.text.

+ [sklearn.linear\_model](https://scikit-learn.org/stable/modules/linear_model.html): một tập hợp các phương thức dành cho hồi quy, trong đó giá trị mục tiêu dự kiến ​​là sự kết hợp tuyến tính của các feature.

+ [sklearn.metrics](https://scikit-learn.org/stable/modules/classes.html#module-sklearn.metrics): định lượng chất lượng dự đoán theo cặp.

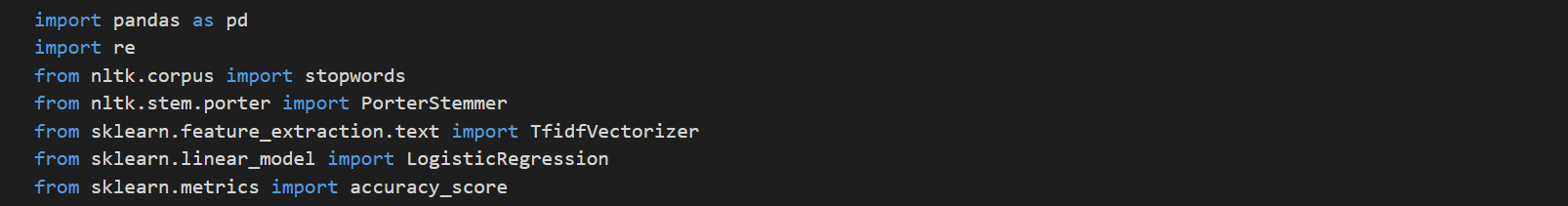
[Regex](https://docs.python.org/3/library/re.html) Biểu thức chính quy (Regular Expressions) trong Python có thể được định nghĩa là chuỗi các ký tự được sử dụng để tìm kiếm một mẫu trong chuỗi. Mô-đun re cung cấp hỗ trợ để sử dụng regex trong chương trình python.

[Pickle](https://docs.python.org/3/library/pickle.html) Mô pickle-đun thực hiện các giao thức nhị phân để tuần tự hóa và hủy tuần tự hóa cấu trúc đối tượng Python. “Pickling” là quá trình theo đó hệ thống phân cấp đối tượng Python được chuyển đổi thành luồng byte và “unpickling” là thao tác nghịch đảo, theo đó luồng byte (từ tệp nhị phân hoặc đối tượng giống byte) được chuyển đổi trở lại thành hệ thống phân cấp đối tượng.

### 2.2.2 Chi tiết thực hiện:

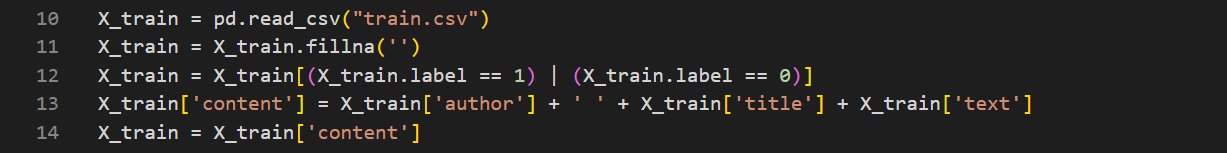
**Tạo model dự đoán và lưu trữ các model này, file đính kèm** [**training.py**](Part1/training.py)

\* Import các thư viện cần thiết:



Công dụng của các thư viện này đã được giải thích cơ bản ở trên, khi được xử dụng sẽ nói chi tiết về việc dùng thư viện đó trong bài toán như thế nào.

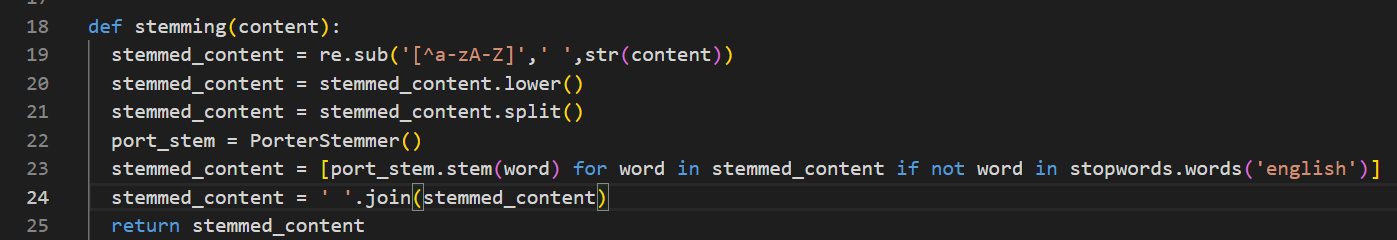
\* Đọc dữ liệu và bỏ đi các bản ghi không phù hợp với pandas:



* Phương thức pandas.read\_cvs() giúp đọc tệp giá trị được phân tách bằng dấu phẩy (csv) vào DataFrame, X\_train có type là DataFrame.
* DataFrame.fillna(‘’) giúp tìm tất cả các giá trị NaN và thay nó thành chuỗi rỗng.
* Loại bỏ các bản ghi với label khác 0 và 1, chúng không mang ý nghĩa gì với quá trình traing.
* Hợp nhất các trường mang lại giá trị đó là: author, title và text để sau chuyển đổi chúng trở thành các feature value của bài toán.

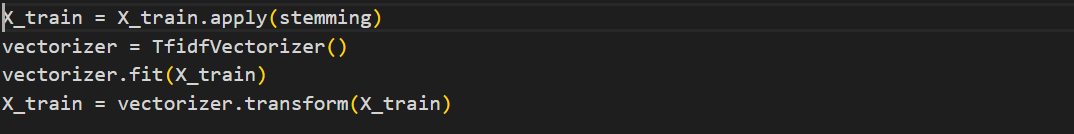
\* Lưu trữ trường label riêng vì chúng đã là dạng số mà máy tính có thể hiểu được, và đồng thời là predict value của các bản ghi training của bài toán.

\* Hàm stemming(content) sau sẽ giúp chuẩn hóa các feature: Đây là một bước quan trọng trong bất cứ phân tích nào nhất là trong machine learning, giữ nguyên dữ liệu gốc (raw documentation) thì khả năng những dữ liệu không cần thiết sẽ gây rối cho quá trình phân tích của máy tính, dẫn đến mất thời gian lâu hơn thậm chí dẫn đến những sai lệch không đáng có.



* Đầu tiên hàm stemming sẽ loại bỏ hết các kí tự đặc biệt và thay bằng chuỗi trống. Thư viện re được dùng ở đây. Các chữ hoa sẽ được chuyển thành chữ viết thường.
* Dữ liệu được chia nhỏ bởi các space.
* stopwords là một module thuộc nltk.corpus, chúng chứa các từ nối các từ không mang nhiều giá trị, ví dụ như: “and”, “or”, “but”... Ở đây chúng ta làm việc với tiếng Anh, chú ý stopwords cần được download trước khi xử dụng. Chúng ta sẽ loại bỏ tất cả các stopwords này.
* Sử dụng phương thức PorterStemmer.stem(word) để loại bỏ các tiền tố và hậu tố, đưa từ trở về từ gốc. Ví dụ training sẽ trở thành train, visited sẽ thành visit...
* Sau đó các từ được nối lại và trả về dữ liệu đã được chuẩn hóa.

\* Chuyển các feature từ dạng văn bản sang vector với TfidVectorizer:



* X\_train sẽ được chuẩn hóa bởi apply(stemming)
* Ta sẽ sử dụng class TfidVectorizer của sklearn.feature\_extraction. Hàm dựng của TfidVectorizer có rất nhiều thông số bổ sung, nhưng ở đây chúng ta chỉ cần một hàm dựng không đối số. Chi tiết về hàm dựng có thể xem ở [đây](https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html#sklearn.feature_extraction.text.TfidfVectorizer.fit).Trước tiên giải thích về TfidVectorizer:

+ TF-IDF (Term Frequency - Inverse Document Frequency: là một thuật toán thống kê số học nhằm phản ánh tầm quan trọng của một từ đối với một văn bản trong một tập hợp hay một ngữ liệu văn bản. Đây là thuật toán rất phổ biến để chuyển đổi văn bản thành một biểu diễn có ý nghĩa của các con số, được sử dụng trong máy học.

+ TF- term frequency – tần số xuất hiện của 1 từ trong 1 văn bản. Số lần xuất hiện của một từ trong một văn bản càng lớn đương nhiên nó sẽ càng quan trọng. Cách tính:

Thương của số lần xuất hiện 1 từ trong văn bản và số lần xuất hiện nhiều nhất của một từ bất kỳ trong văn bản đó. (giá trị sẽ thuộc khoảng [0, 1])

f(t,d) - số lần xuất hiện từ t trong văn bản d.

max{f(w,d):w∈d} - số lần xuất hiện nhiều nhất của một từ bất kỳ trong văn bản.

+ IDF – inverse document frequency. Tần số nghịch của 1 từ trong tập văn bản (corpus). Tính IDF để giảm giá trị của những từ phổ biến. Mỗi từ chỉ có 1 giá trị IDF duy nhất trong tập văn bản. Ở trên ta đã khẳng định tính quan trọng của những từ xuất hiện nhiều lần tuy nhiên, nếu từ đó xuất hiện nhiều lần ở tất cả các văn bản thì có khả năng đó là từ phổ biến. Ví dụ: nếu đề cập đến chủ đề World Cup, thì từ này sẽ được đề cặp đến rất nhiều trong nhiều văn bản nhưng lại không mang đến quá nhiều giá trị. Cách tính:

|D|: tổng số văn bản trong tập D

: số văn bản chứa từ nhất định, với điều kiện t xuất hiện trong văn bản d. Nếu từ đó không xuất hiện ở bất cứ 1 văn bản nào trong tập thì mẫu số sẽ bằng 0 => phép chia cho không không hợp lệ, vì thế người ta thường thay bằng mẫu thức 1+.

Cơ số logarit trong công thức này không thay đổi giá trị của 1 từ mà chỉ thu hẹp khoảng giá trị của từ đó. Vì thay đổi cơ số sẽ dẫn đến việc giá trị của các từ thay đổi bởi một số nhất định và tỷ lệ giữa các trọng lượng với nhau sẽ không thay đổi. (nói cách khác, thay đổi cơ số sẽ không ảnh hưởng đến tỷ lệ giữa các giá trị IDF). Tuy nhiên việc thay đổi khoảng giá trị sẽ giúp tỷ lệ giữa IDF và TF tương đồng để dùng cho công thức TF-IDF như bên dưới.

+ Giá trị TF-IDF:

Những từ có giá trị TF-IDF cao là những từ xuất hiện nhiều trong văn bản này, và xuất hiện ít trong các văn bản khác. Việc này giúp lọc ra những từ phổ biến và giữ lại những từ có giá trị cao (từ khoá của văn bản đó).

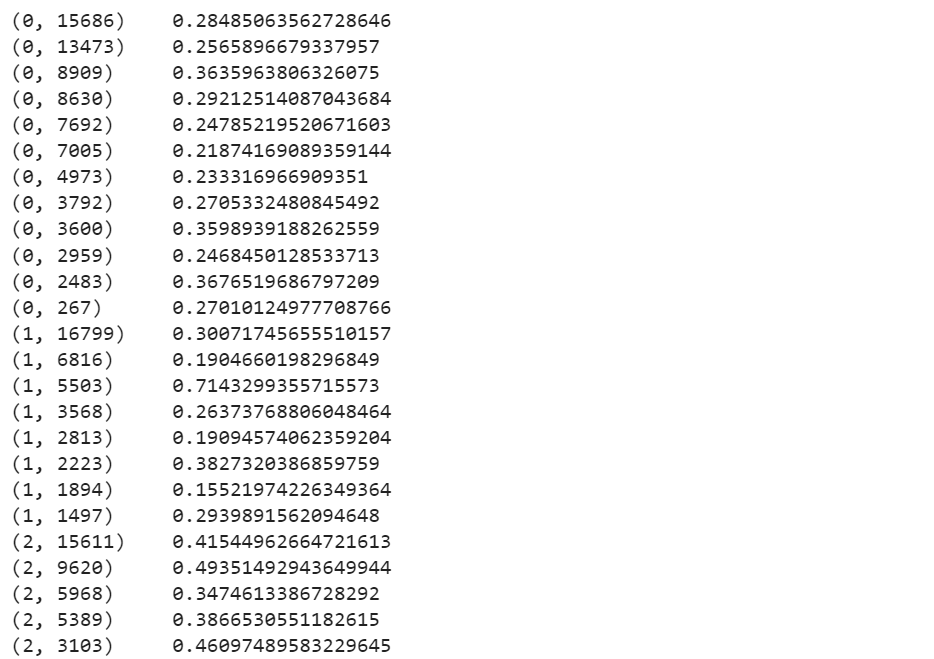
* Quay lại với TfidVectorizer được cung cấp sắn bởi Python, nhờ nó mà chúng ta có thể giải quyết những tính toán phức tạp ở trên một cách nhanh chóng:

+ TfidVectorizer.fit(raw\_documents): Học từ vựng và IDF từ bộ đào tạo, nó sẽ trả về TfidVectorizer phù hợp với tập training được đưa vào.

+ transform(raw\_documents): Chuyển đổi tài liệu thành ma trận tài liệu bởi những thứ mà nó học được từ hàm fit(raw\_documents) ở trên.

+ Ngoài ra có thể dùng hàm fit\_transform(raw\_documents) để thực hiện cả hai chức năng trên.

* Kết quả mà chúng ta nhận được sẽ có dạng như sau:



\* Xây dựng model dự đoán với LogisticRegression của sklearn.linear\_model:

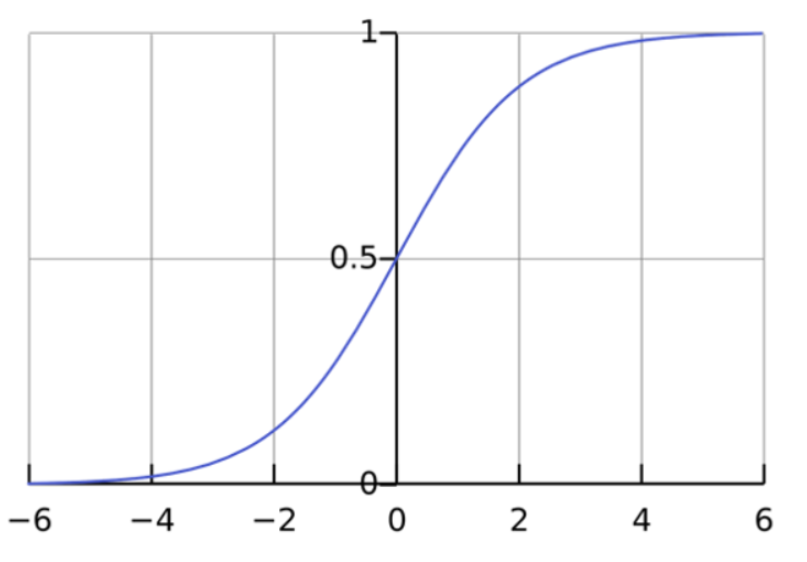
* Đầu tiên, ta sẽ tìm hiểu về LogisticRegression, đây là phương pháp để đưa ra dự đoán dạng nhị phân, trong bài toán của chúng ta là tin thật/ giả:

+ Hàm Sigmoid là hàm thường được sử dụng trong LogisticRegression của Python. Công thức của nó là:

Trong đó:

* y là xác suất dự đoán, nó luôn nằm trong khoảng từ [0,1] theo tính chất của hàm Sigmoid
* s = WX + b, với w là trọng số và X là input features, b là hệ số tự do, với bài toán có nhiều feature như bài toán của chúng ta WX = w1x1 + w2x2+… wnxn.

+ Nếu vẽ phương trình hồi quy logistic này, bạn sẽ có một đường cong hình chữ S như hình dưới đây:



+ Khi đó dự đoán nhị phân được đưa ra dựa trên Threshold Value = 0.5, khi đó y <0.5 thì sẽ được cho là có giá trị tương đương với 0 và y >= 0.5 được cho là có giá trị tương đương với 1.

* Quay trở lại với LogisticRegression đã được dựng sẵn bởi Python. Hàm dựng tổng quát của LogisticRegression có thể xem ở [đây](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html). Chúng ta sẽ sử dụng hàm dựng không có đối số và xử lý dữ liệu như sau:



Hàm fit(X, y, sample\_weight=None)

Tham số:

X: {array-like, sparse matrix} of shape (n\_samples, n\_features): Vectơ training, trong đó n\_samples là số lượng mẫu và n\_features là số lượng feature.

y: array-like of shape (n\_samples,): Vectơ mục tiêu liên quan đến X.

sample\_weightarray-like of shape (n\_samples,) default=None: Mảng trọng số được gán cho các mẫu riêng lẻ. Nếu không được cung cấp, thì mỗi mẫu được cung cấp trọng lượng đơn vị.

Returns: self, công cụ ước tính được trang bị.

\* Model đã được xây dựng nhưng công việc của chúng ta vẫn còn: kiểm tra độ chính xác của model. Việc kiểm tra này được dựa trên sự ăn khớp giữa predict chúng ta tìm được khi áp dụng model với tập X\_train và giá trị Y\_train thực tế đã có:

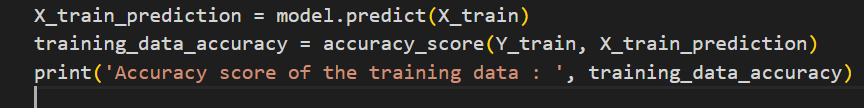
* Chúng ta sẽ dựa vào hàm sklearn.metrics.accuracy\_score(y\_true, y\_pred, \*, normalize=True, sample\_weight=None): Điểm phân loại độ chính xác. Trong phân loại nhiều nhãn, hàm này tính toán độ chính xác của tập hợp con: tập hợp nhãn được dự đoán cho một mẫu phải khớp chính xác với tập hợp nhãn tương ứng trong y\_true:

+ Tham số :

* y\_true: 1d array-like hoặc mảng nhãn/ma trận thưa, đó là nhãn chân lý (chính xác).
* y\_pred: 1d array-like hoặc mảng nhãn/ma trận thưa, nhãn dự đoán
* normalize: bool, mặc định=True. Nếu False, trả lại số lượng mẫu được phân loại chính xác. Nếu không, hãy trả lại số phần mẫu được phân loại chính xác.
* sample\_weight array-like of shape (n\_samples,), default=None, trọng lượng mẫu.

+ Return: float, dựa vào tham số normalize.

* Chúng ta sử dụng hàm này như thế nào:

Hàm predict( X )

Dự đoán nhãn lớp cho các mẫu trong X:

+ Tham số :X {dạng mảng, ma trận thưa thớt} (n\_samples, n\_features), Ma trận dữ liệu muốn nhận dự đoán.

+ return: y\_pred nd-array dạng (n\_samples,), Vector chứa các nhãn lớp cho mỗi mẫu.

* Như vậy chỉ cần tính accuracy\_score của Y\_train, X\_train\_prediction, ta sẽ thấy được phần nào sự chính xác của model. Giá trị được tính ra là 0.9795192307692308, một độ chính xác khá cao.

\* Để tiết kiệm thời gian sử dụng model cho những phân tích sau này nhất là khi chúng ta sử dụng PyScript để chạy code Python sẽ gặp bất lợi về thời gian, chúng ta sẽ lưu lại giá trị của vectorizer để chuyển dữ liệu input thành các vector và model để dự đoán kết quả với việc sử dụng thư viện pickle.

