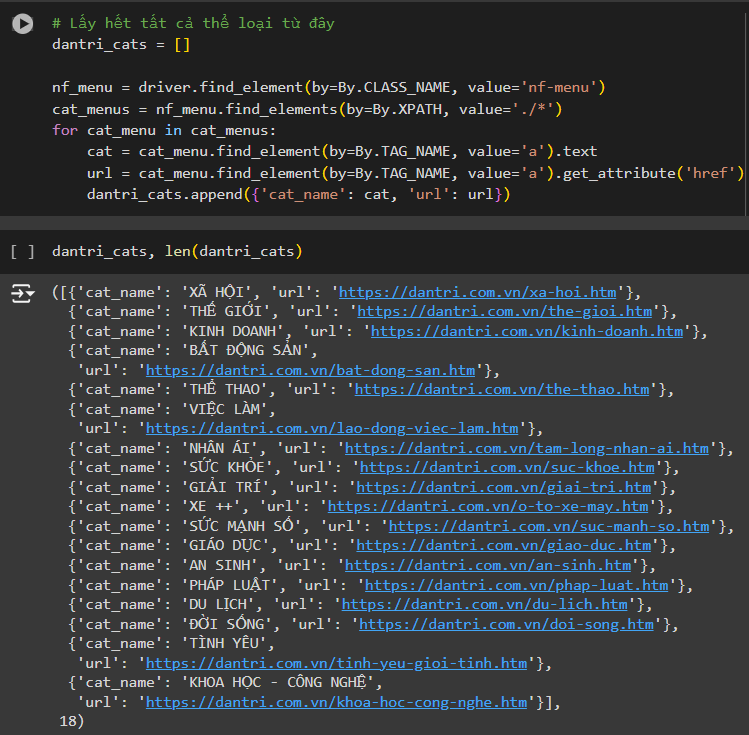
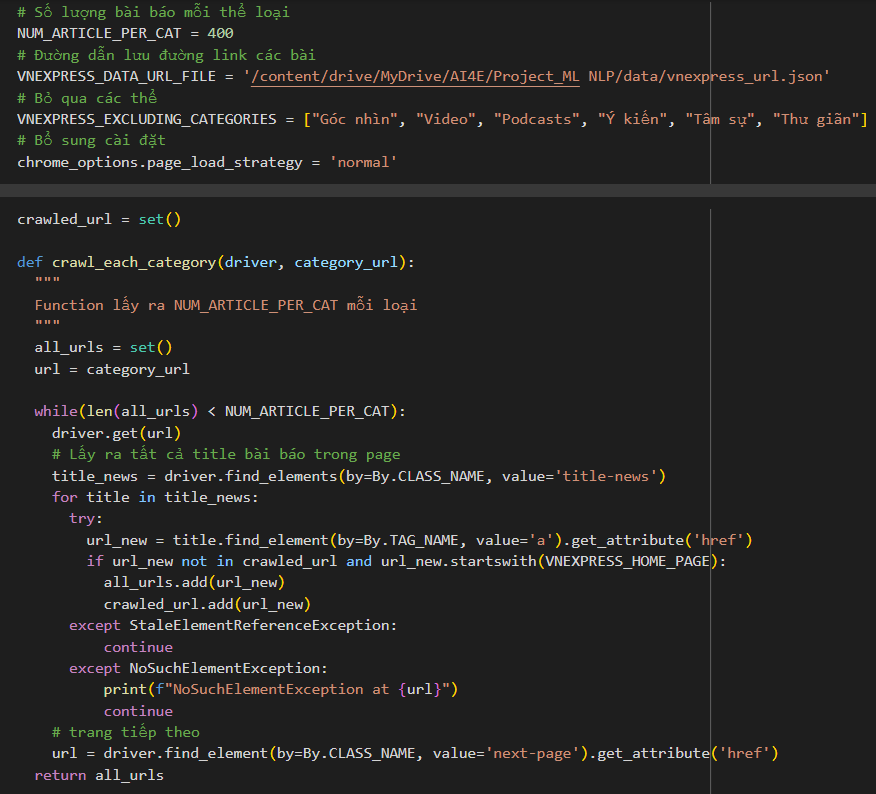
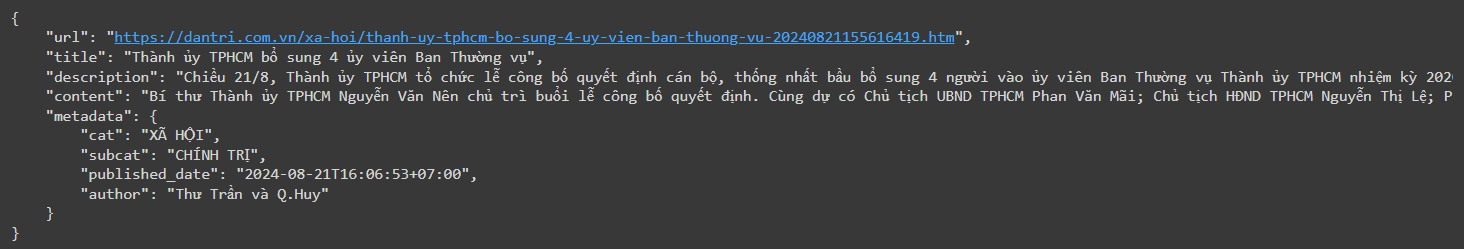
**BÁO CÁO PROJECT ML/NLP**

1. Thành viên nhóm
2. Ngô Minh Quân
3. Nguyễn Quang Minh
4. Lê Minh Hoàng
5. Mai Tiến Dũng
6. Chi tiết thực hiện Project
7. Thu thập news từ 3 nguồn báo: VNExpress, Dân trí, VietNamNet

* Sử dụng thư viện selenium để crawl
* Quy trình thực hiện:
  + Thu thập url và tên các thể loại báo từ menu theo yêu cầu
  + Với từng thể loại thu thập ~ 400 đường dẫn bài báo
  + Demo với báo Dân Trí
* Tổng cộng thu được 20522 bài báo, trong đó đặc biệt có thể loại Văn hóa thu thập ~1000 bài do chỉ có ở trong báo VietNamNet, đảm bảo chất lượng phân lọai, phân cụm.

1. Bóc tách các trường:

* Cài đặt thêm strategy ‘eager’ để thu thập bỏ qua các hình ảnh
* Các trường được bóc tách: Tiêu đề, url, description, thể loại, ngày phát hành, tác giả, content
* Báo VNExpress: truy cập vào đường link từng bài báo
  + Tiêu đề nằm trong thẻ ‘title-detail’
  + Description nằm trong thẻ ‘description’
  + Thể loại nằm trong các thẻ thẻ ‘breadcrumb.li’, thể loại chính nằm ở vị trí thứ nhất, thể loại con nằm ở vị trí thứ 2 nếu có
  + Ngày phát hành nằm ở thuộc tính ‘content’ trong thẻ có giá trị ‘[itemprop="datePublished”’
  + Content ở trong các thẻ con trực tiếp của thẻ ‘article.fck\_detail’, bỏ qua các ảnh, caption ảnh và bảng
* Báo VietNamNet:
  + Tiêu đề nằm trong thẻ ‘content-detail-title’
  + Description nằm trong thẻ ‘content-detail-sapo’
  + Thể loại nằm trong các thẻ thẻ ‘bread-crumb-detail.li’, thể loại chính nằm ở vị trí thứ nhất, thể loại con nằm ở vị trí thứ 2 nếu có
  + Ngày phát hành: lấy thẻ thứ 2 có thuộc tính type="application/ld+json", ngày phát hành nằm trong 'datePublished'
  + Content ở trong các thẻ con trực tiếp của thẻ ‘maincontent’, bỏ qua các ảnh, caption ảnh và bảng
  + Author nằm trong thẻ ‘article-detail-author\_\_info’ hoặc phần tử cuối cùng của content
* Báo Dân trí: Bỏ qua các emagazine
  + Tiêu đề nằm trong thẻ ‘title-page’
  + Description nằm trong thẻ ‘singular-sapo’, bỏ đi “Dân tri -” ở đầu description
  + Thể loại nằm trong các con của thẻ ‘dt-font-Inter’, thể loại chính nằm ở thẻ có thuộc tinh ‘'data-content-piece' có giá trị bằng ‘article-breadcrumb-position\_1’, thể loại con nằm ở vị trí có giá trị ‘article-breadcrumb-position\_1’
  + Ngày phát hành: lấy thẻ thứ 2 có thuộc tính type="application/ld+json", ngày phát hành nằm trong 'datePublished'
  + Content ở trong các thẻ con trực tiếp của thẻ ‘singular-content’, bỏ qua các ảnh, caption ảnh và bảng
  + Author nằm trong thẻ ‘author-name’.
* Kết quả mỗi bài báo sẽ có dạng:

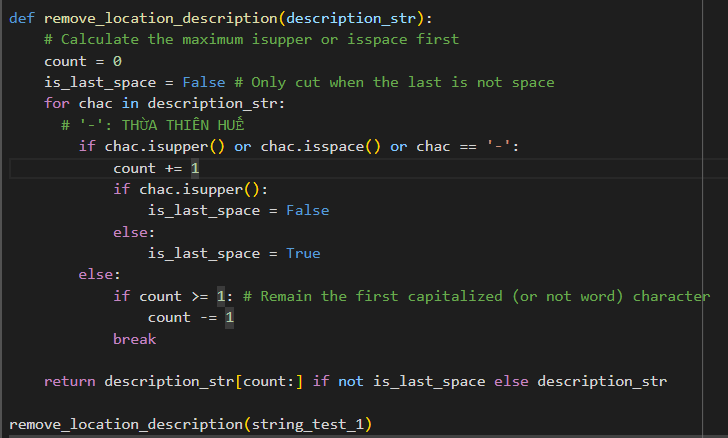
1. Xử lí dữ liệu sau khi thu thập
2. Xử lí chung cho các thể loại:

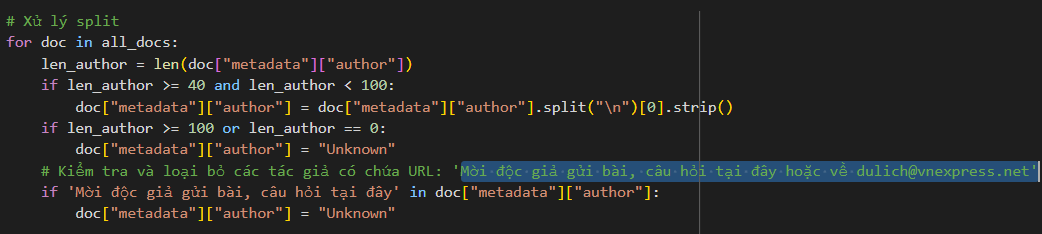
* Kiểm tra và loại bỏ các description có độ dài < 50
* Kiểm tra tác giả, với các tác giả có độ dài tên > 100 kí tự hoặc xâu rỗng thì đổi về “Unkown’
* Loại bỏ các bài báo có độ dài content < 300 kí tự, đó là các video, multiple-choice questions, quảng cáo, thông báo, giới thiệu thông tin các nhân… Ví dụ



* Định dạng ngày tháng về timestamp

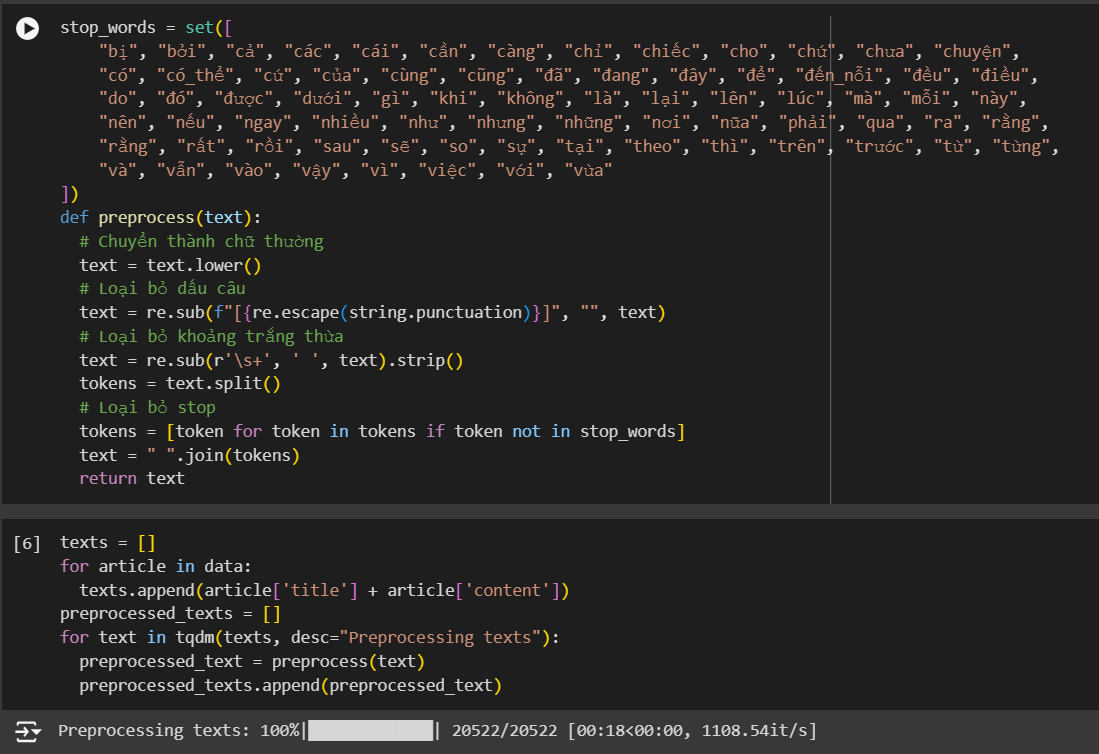
1. Các trường hợp xử lí riêng:

* Xử lí lỗi dính chữ là các tỉnh ở trước mỗi description:
* Xử lí thêm về tác giả: tác giả bị dính theo sau 1 đoạn văn bản ngăn cách bằng ‘\n’ hay các độ dài < 100 nhưng có dạng: “Mời độc giả gửi bài, câu hỏi tại đây hoặc về dulich@vnexpress.net”

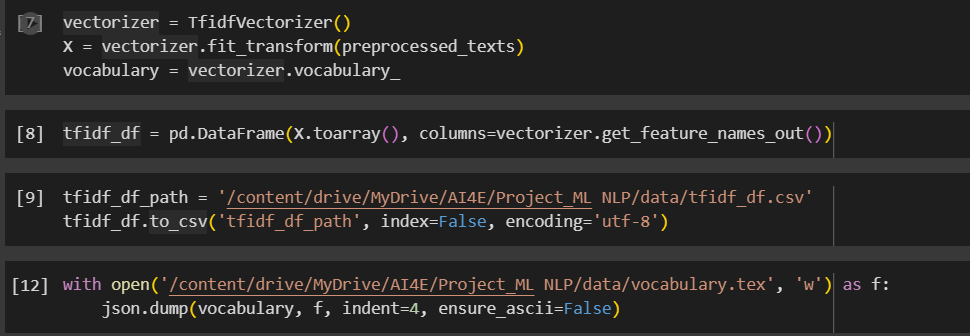


4. Tiền xử lí và lưu biểu diễn, từ điển:

* Tiền xử lí văn bản:

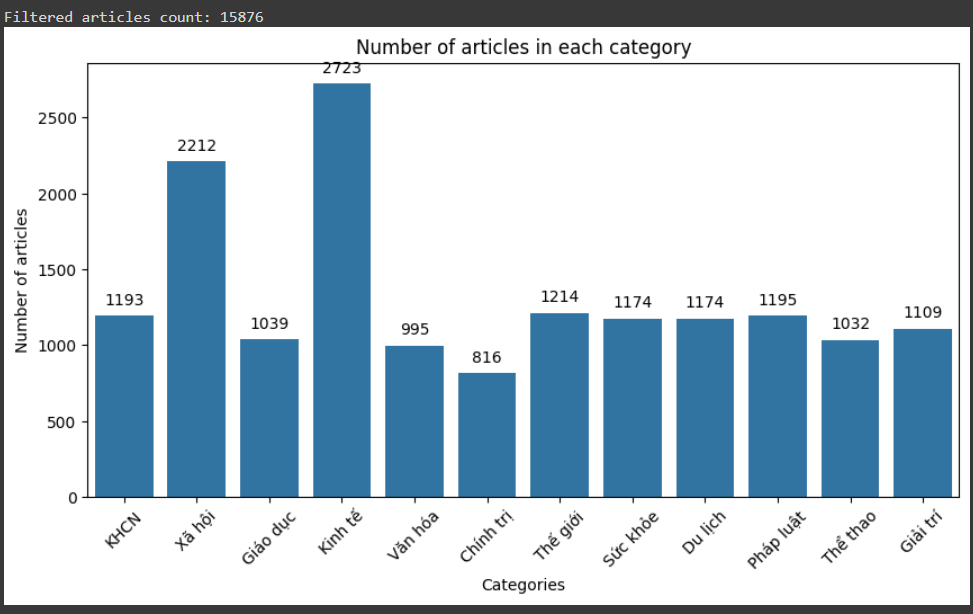


* Lưu biểu diễn TF-IDF và từ điển:

5. Phân loại các thể loại:

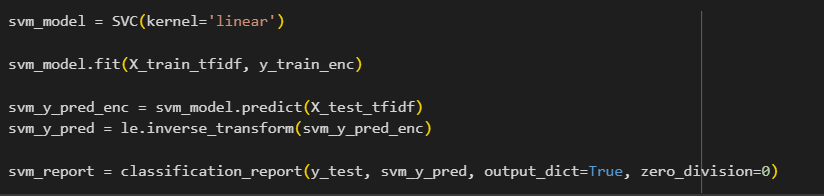
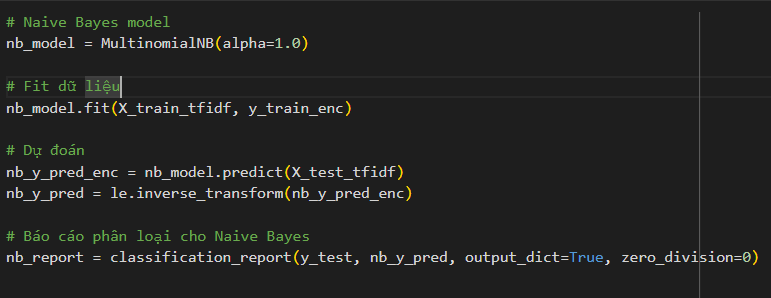
1. Tiền xử lí dữ liệu:

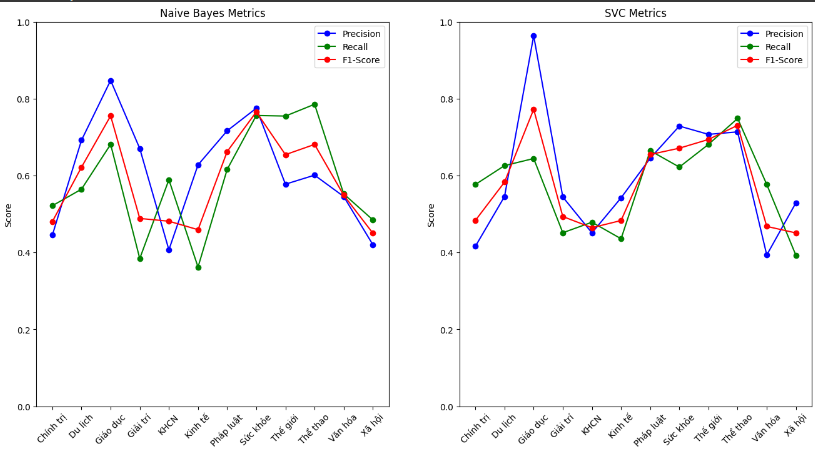
* Chuyển đổi các thể loại có liên quan thành 12 thể loại theo yêu cầu, tổng cộng có 15876 bài báo sau khi lọc

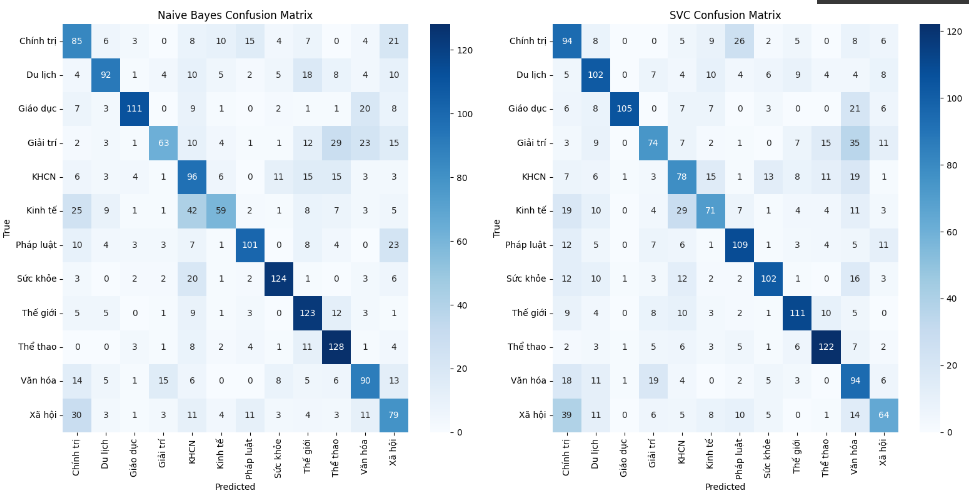


* Dữ liệu chưa cân bằng giữa các thể loại, lấy mẫu 816 bài/thể loại, tổng cộng dữ liệu để huấn luyện có 9792 bài
* Lấy ra text gồm content và title của các bài báo và nhãn là các thể loại tương ứng
* Làm sạch dữ liệu: chuyển về chữ thường, loại bỏ dấu câu, khoảng trắng thừa và các stop words
* Mã hóa nhãn và chia dữ liệu thành tập train và test với tỉ lệ 8:2

1. Huấn luyện các mô hình truyền thống

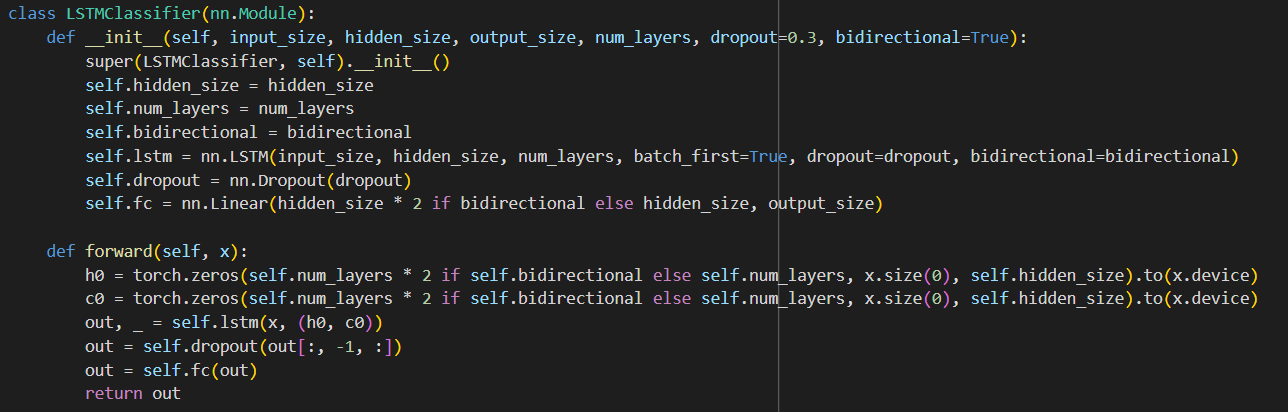
* Sử dụng TF-IDF để biểu diễn texts thành các vector
* Huấn luyện mô hình Naive Bayes:
* Huấn luyện mô hình SVM:
* Kết quả: Accurancy Naive Bayes: 0.59, SVM: 0.57



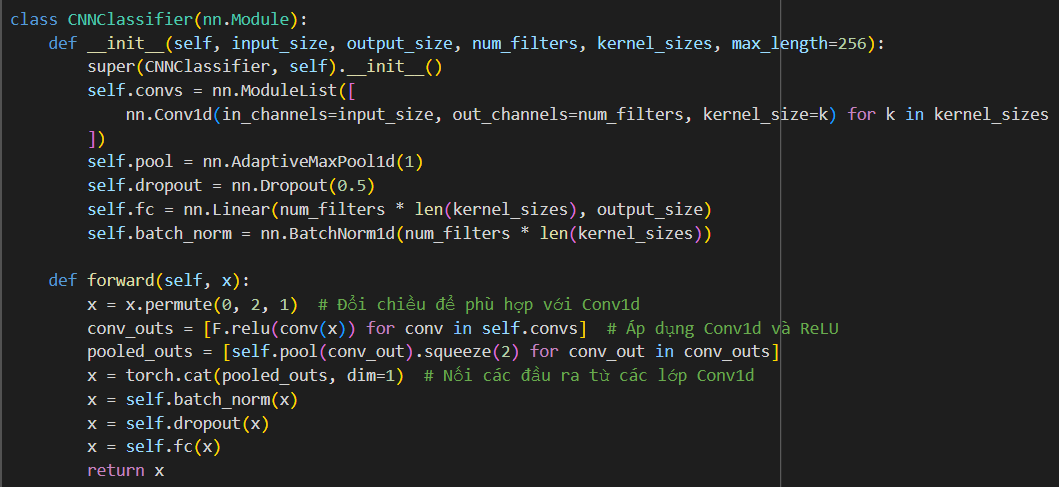


1. Huấn luyện các mô hình học sâu:

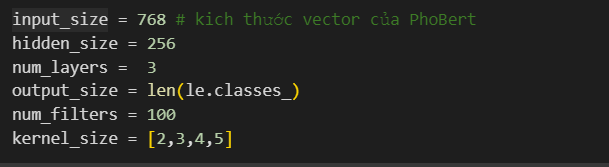
* Sử dụng phobert-base để mã hóa văn bản
* Tạo DataLoader cho mô hình huấn luyện
* Kiến trúc các mô hình học sâu LSTM:
  + Khởi tạo mô hình bao gồm input\_size, hidden\_size, output\_size, nums\_layer, bidirectional học ngữ cảnh 2 chiều
  + Bao gồm các lớp:
    - LSTM layer: nhận đầu vào là chuỗi có kích thước input\_size, trả về trạng thái ẩn có kích thước hidden\_size
    - Dropout Layer
    - Lớp Fully conected nhận đầu vào là trạng thái ẩn cuối cùng trả về đầu ra có kích thước output\_size

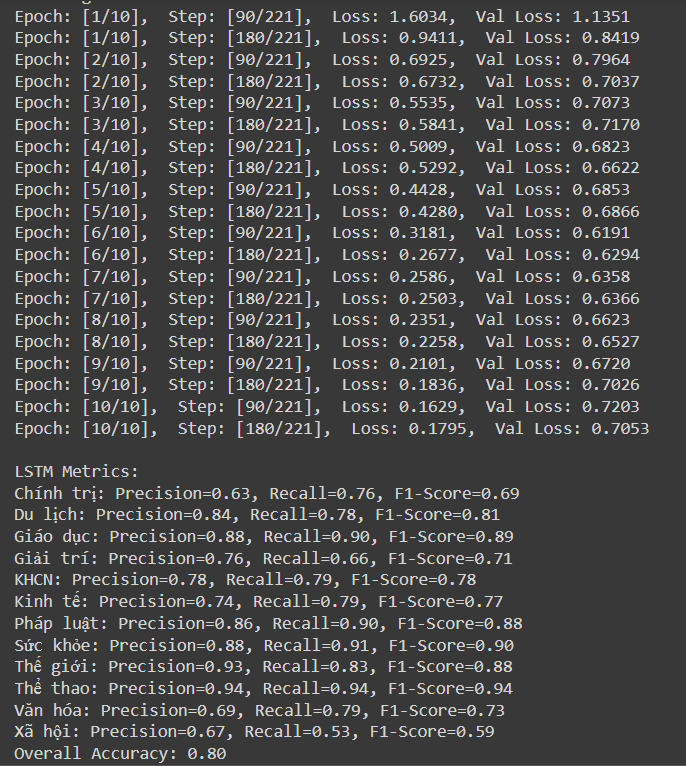
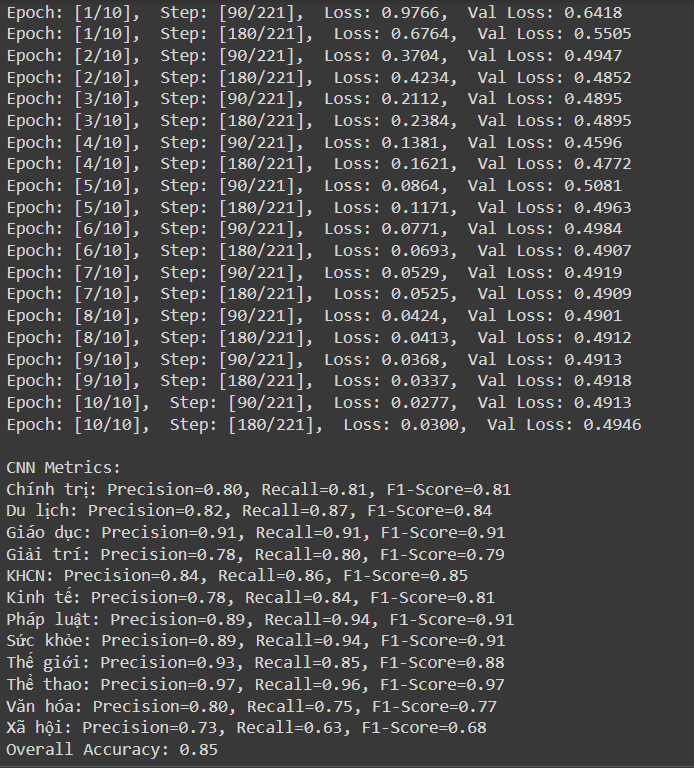
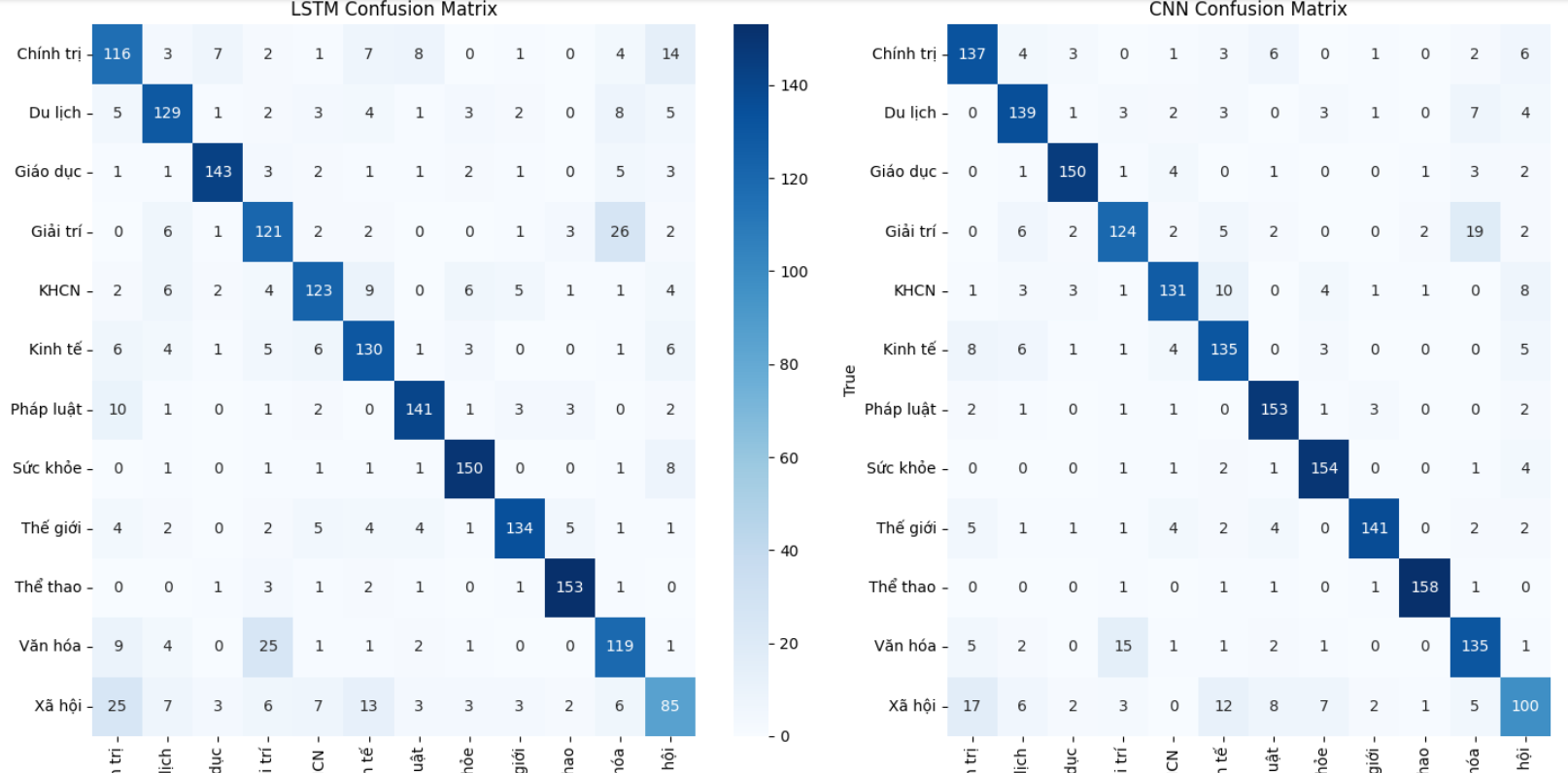
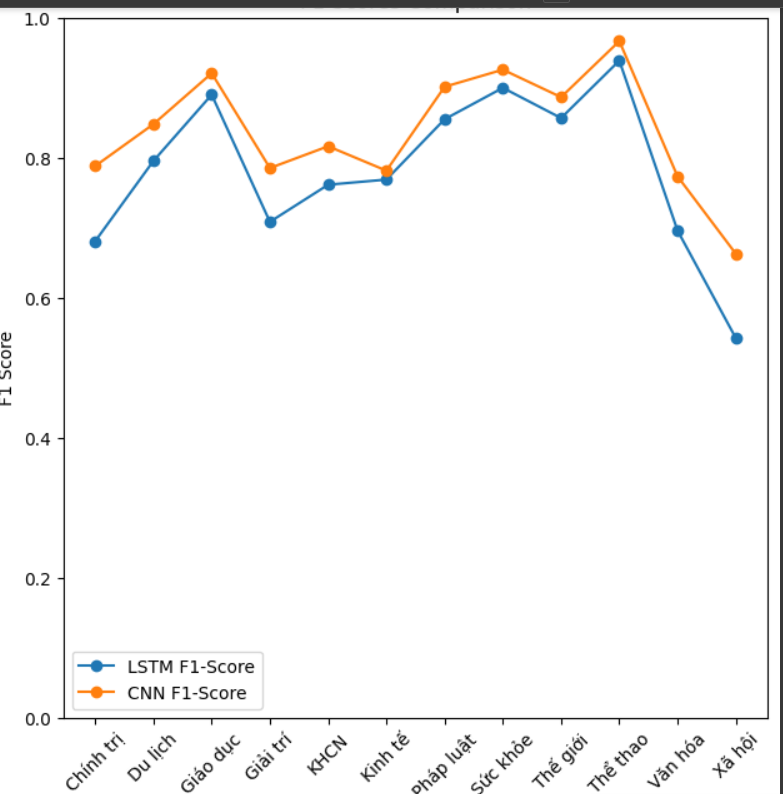
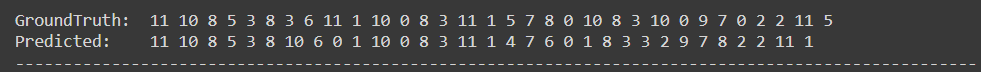


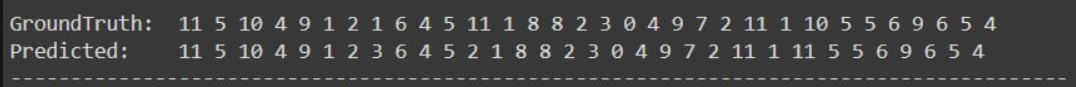
* Kiến trúc mô hình CNN: bao gồm các lớp
  + 4 lớp convolution layers với kích thước kernel 2,3,4,5
  + Lớp Maxpooling thực hiện giảm chiều dữ liệu
  + Dropout giảm thiểu overfitting
  + Fully connected nhận đầu vào là kết quả từ lớp tích chập và pooling đầu ra có kích thước output\_size
  + Batch Normalization: chuẩn hóa dữ liệu

- 

* Hyper parameters của 3 mô hình sau khi thử nghiệm:



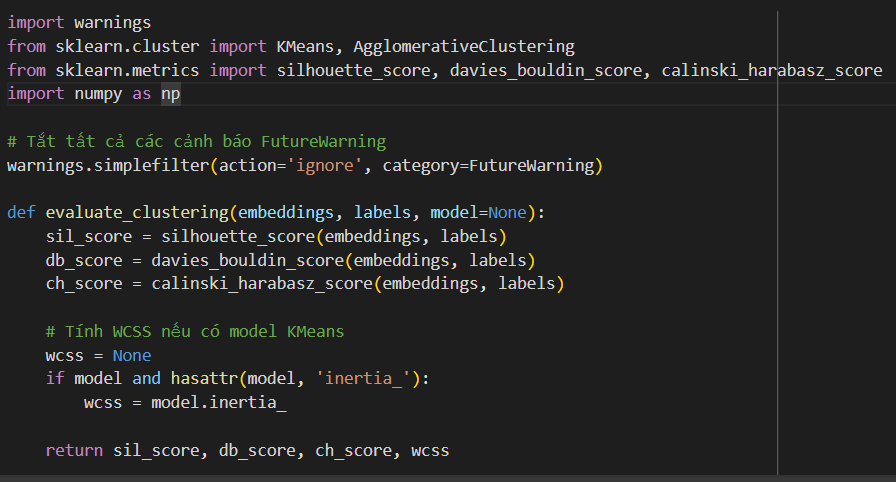
* Kết quả:
  + LSTM model:
  + CNN model: 
* So sánh F1-score:
* Ma trận nhầm lẫn:
* Demo 1 số nhãn:
  + LSTM:
  + CNN:



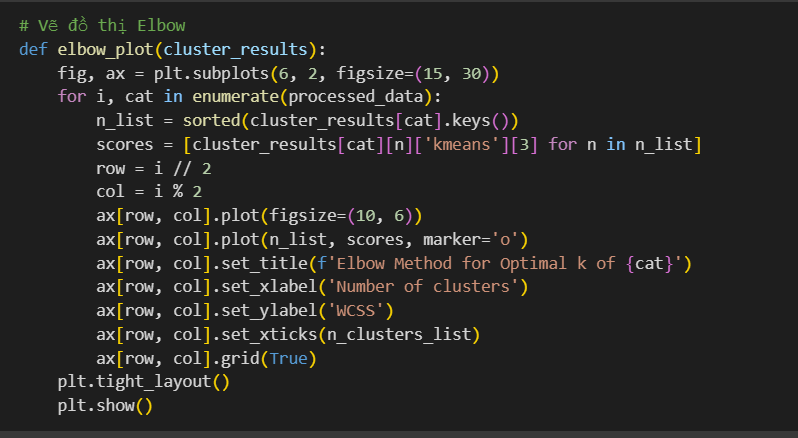
5. Phân cụm các thể loại:

1. Tiền xử lí văn bản: Tương tự với bài toán phân loại
2. Hàm và các biểu đồ đánh giá:

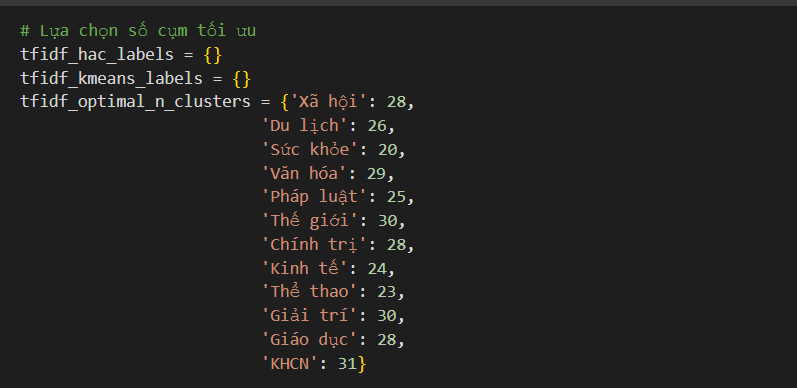
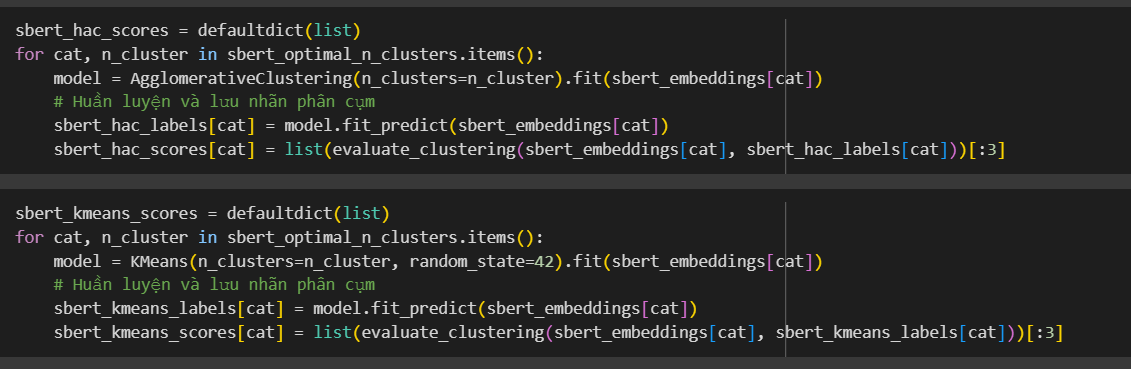
* Sử dụng các chỉ số silhouette\_score, davies\_bouldin\_score, calinski\_harabasz\_score để đánh giá chất lượng phân cụm

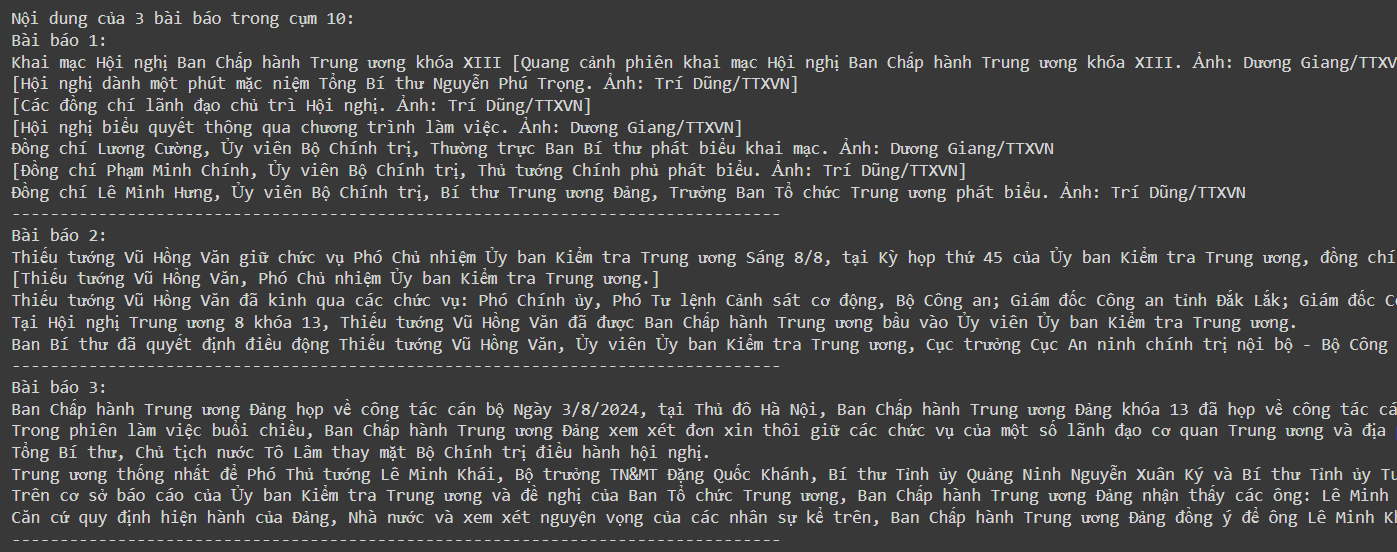


* Vẽ đồ thị elbow tìm điểm khuỷu tay là số lượng cụm, với mỗi thể loại từ 10-40 cụm

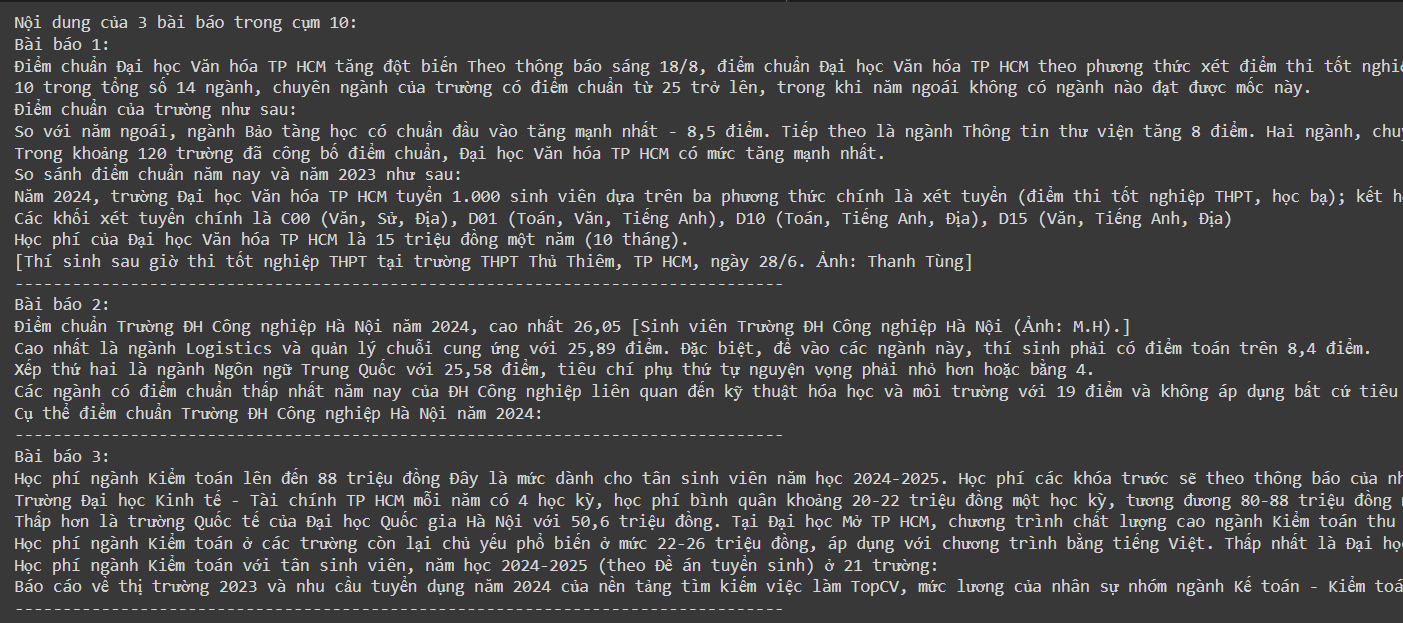


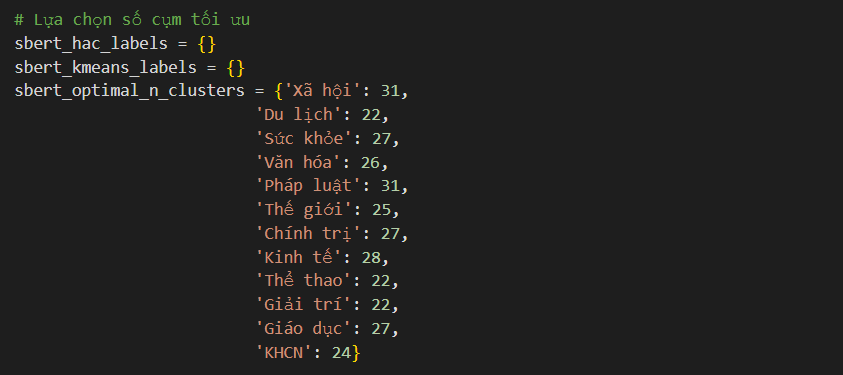
1. TFIDF:

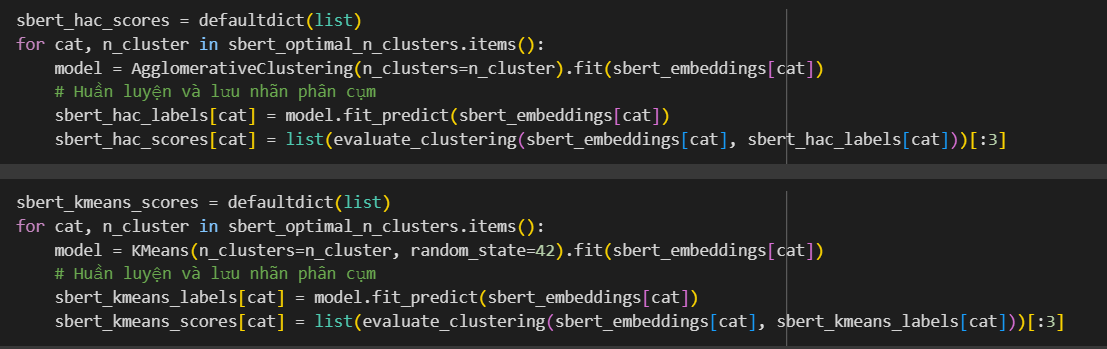
* Sử dụng TF-IDF biểu diễn văn bản thành các vector
  + Từ biểu đồ elbow lựa chọn được số cụm của từng loại:
  + Huấn luyện mô hình với 2 thuận toán phân cụm phân cấp HAC và phâm cụm KMeans và lưu các đánh giá của từng model
  + Demo 1 vài kết quả: 
    - Lấy mẫu 3 bài trong cụm 10 của thể loại Chính trị (TFIDF + HAC) đều nói về Trung ương Đảng

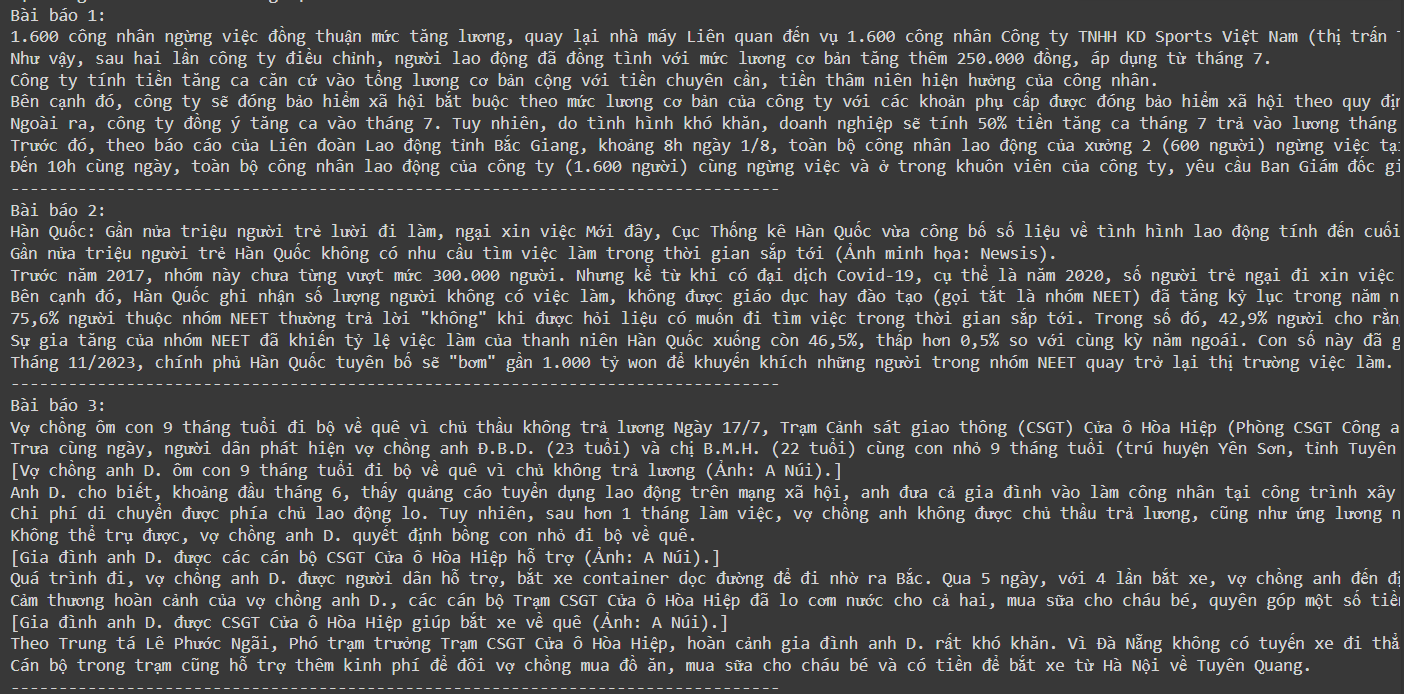


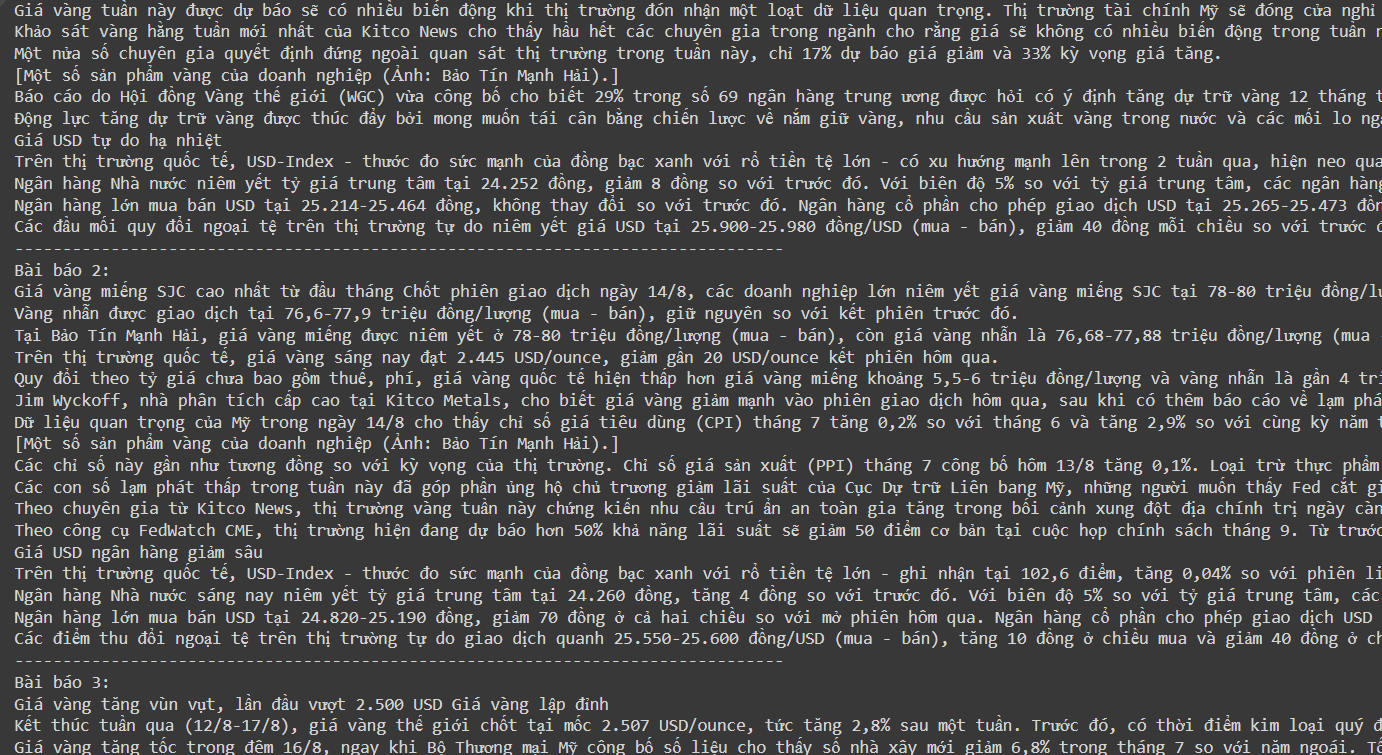
* Lấy mẫu 3 bài trong cụm 10 của thể loại Giáo dục(TFIDF + Kmeans) đều nói về Đại học và tân sinh viên

1. SBert:

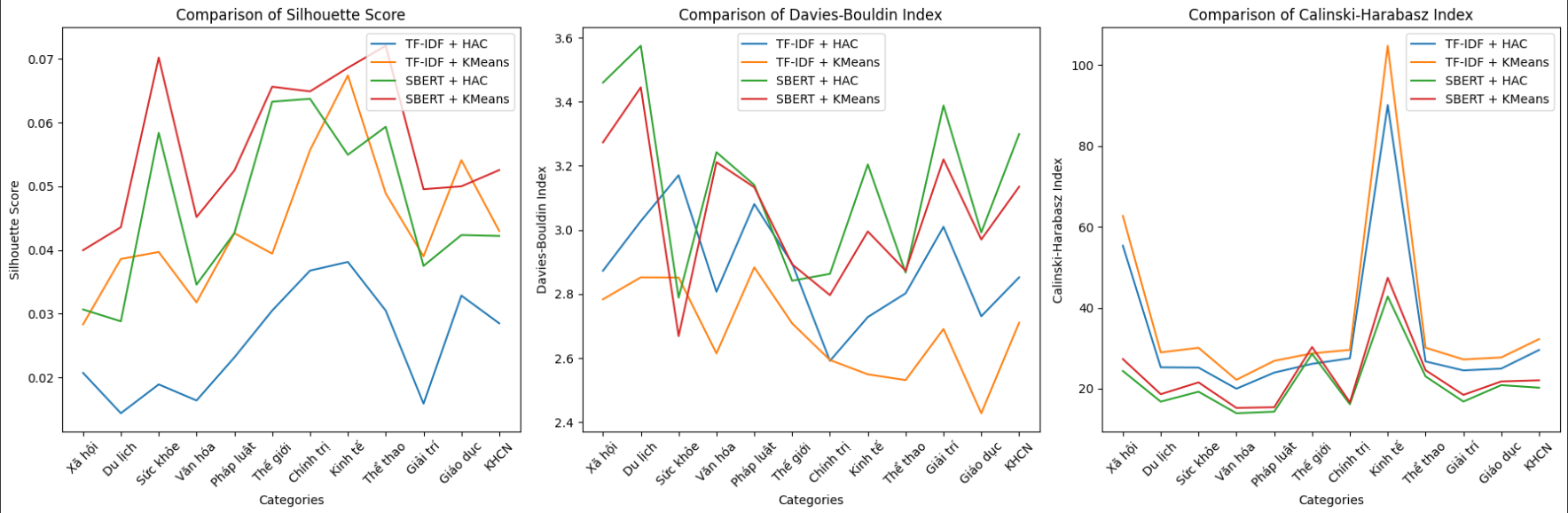
* Cài đặt model Sbert của thư viện sentence\_transformer để biểu diễn văn bản thành vector có kích thước 768
* Từ biểu đồ elbow lựa chọn được số cụm của từng loại:
* Huấn luyện mô hình với 2 thuận toán phân cụm phân cấp HAC và phân cụm KMeans và lưu các đánh giá của từng model:



* Demo 1 vài kết quả:
  + Lấy mẫu 3 bài trong cụm 15 của thể loại Kinh tế(SBERT + Kmeans) đều nói về Giá vàng
  + Lấy mẫu 3 bài trong cụm 10 của thể loại Xã hội(SBERT + HAC) đều nói về Lương

1. Đánh giá so sánh các model phân cụm:

* Biểu đồ so sánh 3 chỉ số silhouette\_score, davies\_bouldin\_score, calinski\_harabasz\_score

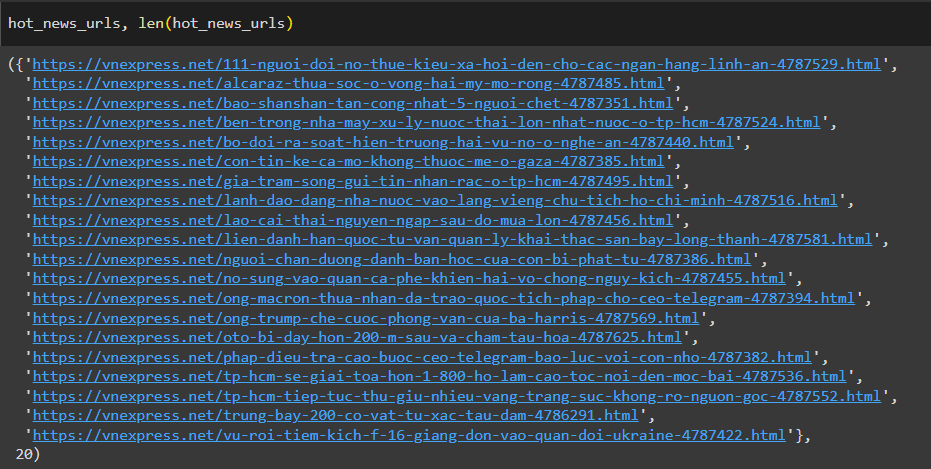


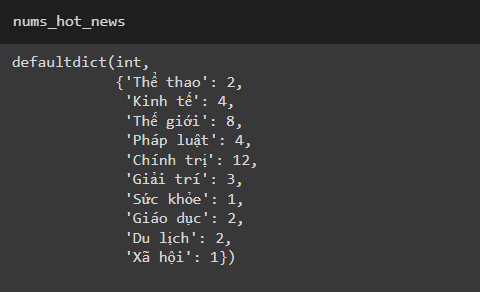
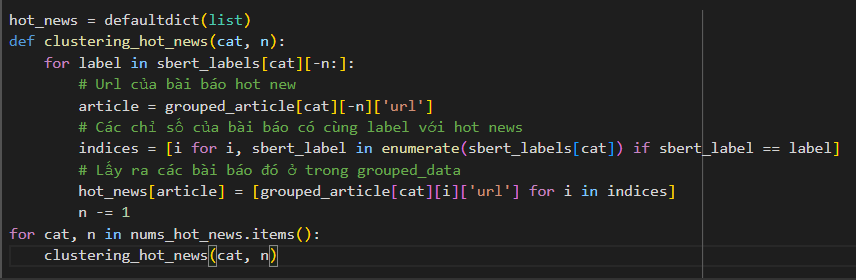
* Từ biểu đồ ta có nhận xét:
  + Kmeans có chỉ số đánh giá tốt hơn HAC trên cả 3 chỉ số trong tập dữ liệu này, cụ thể:
    - Chỉ số silhouette\_score Kmeans + Sbert có kết quả tốt nhất, thể hiện mức độ tương đồng của các điểm dữ liệu so với cụm của nó và các cụm khác khá tốt
    - Chỉ số davies\_bouldin\_score Kmeans + TF-IDF có giá tri thấp nhất, thể hiện mức độ tương đồng của các cụm thấp
    - Chỉ số calinski\_harabasz\_score Kmeans + TF-IDF có giá trị cao nhất thể hiện mức độ phân tách giữa các cụm tốt nhất
    - Nguyên nhân: Kmeans có khả năng xử lí tốt các tập dữ liệu lớn và có số chiều cao. Ngoài ra các thể loại báo được phân loại không có sự phân cấp rõ ràng thành các cụm nên HAC có kết quả không tốt bằng Kmeans
  + SBert có chỉ số silhouette\_score tốt hơn nhưng chỉ số davies\_bouldin\_score, calinski\_harabasz\_score tệ hơn TF-IDF
    - Sbert biểu diễn vector với số chiều cao hơn(768), giúp hiểu được ngữ cảnh của văn bản, tạo ra vector có sự gắn kết tốt trong cụm nên silhouette\_score cao hơn, nhưng sẽ có ý nghĩa chồng chéo giữa các cụm nên phân tách không được tốt nên davies\_bouldin\_score, calinski\_harabasz\_score tệ hơn
    - TF-IDF mã hóa dựa trên tần suất từ xuất hiện, không nắm bắt được ngữ nghĩa nên độ tương đồng giữa điểm dữ liệu và cụm không cao, nhưng phân tách các cụm tốt hơn, đặc biệt khi các cụm có các từ khóa đặc biệt

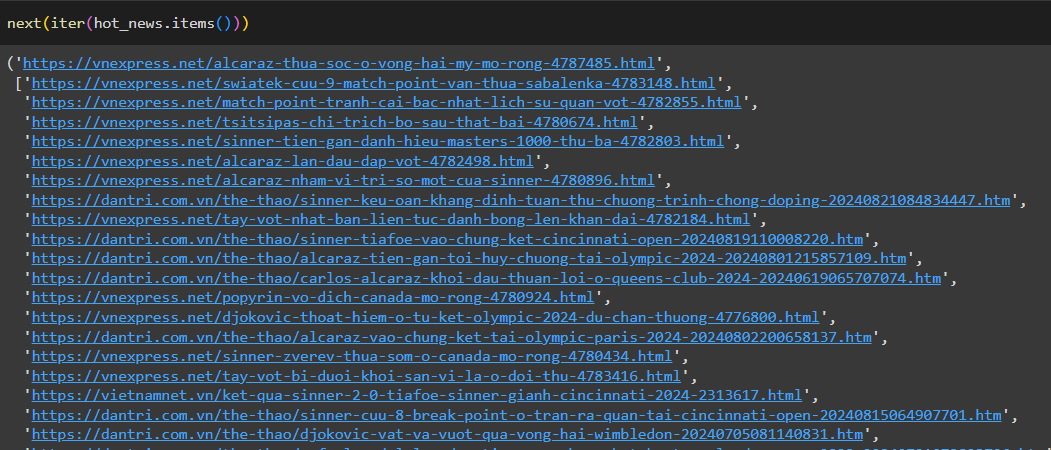
6. Phần yêu cầu thêm

1. Tổng hợp và phân cụm Hot news

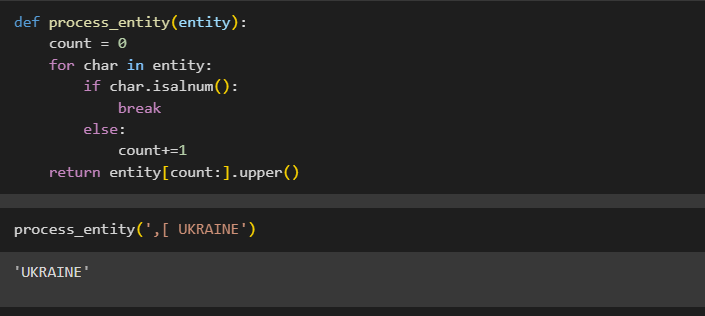
* Thu thập Hot news:
  + Truy cập vào page hot news của báo VnExpress và VietNamNet
  + Tương tự với thu thập các bài báo ở trên, thu thập đường dẫn tới các hot news và bóc tách các trường



* Xử lí sau khi thu thập:
  + Xử lí chung với các bài báo ở bước 3, với các hot news sẽ được thêm trường ‘hot\_news’ vào
  + Sau khi xử lí xong, tách hot\_news ra để sau khi xử lí xong cả 3 nguồn báo, shuffle xong thêm hot\_news vào cuối để thuận tiện cho việc phân cụm hot\_news sau này
* Phân cụm:
  + Thống kê các hot\_news trong từng thể loại, tương ứng là các bài báo cuối cùng mỗi thể loại trong grouped\_articles:
* Với kết quả phân cụm ở bước trên, lấy nhãn của các hot news này và tìm chỉ số của các bài báo có nhãn tương ứng
* Demo kết quả của cụm đầu tiên: với hot new là 1 tin tức thể thao về tennis, kết quả phân cụm gồm các bài báo về tennis



1. Phát hiện các thực thể được đặt tên NER trong các bài báo

* Sử dụng thư viện underthesea
* Tiền xử lí các văn bản, loại bỏ các khoảng trắng thừa và các stop word. Không đưa văn bản về chữ thường trước rồi mới bỏ stop words do các thực thể được đặt tên có viết hoa
* Hàm xử lí tên các thực thể: khi phát hiện tên các thực thể có thể bị dính các kí tự của phần văn bản trước đó
* Sử dụng thuật toán NER của thư viện underthesea để phát hiện các token là các thực thể được đặt tên
  + Các nhãn B- (bắt đầu 1 thực thể), I - (bên trong 1 thực thể), O (không phải thực thể)
  + Chỉ lấy ra các token có nhãn B- và I-, token từ B- ghép liên tục cho tới token có nhãn I- xa nhất ta được tên của 1 thực thể
  + Lấy ra url của văn bản đó và lưu vào cơ sở dữ liệu tương ứng của thực thể
  + Demo kết quả: