ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN CUỐI KỲ

Môn học: Máy học

Học kỳ II (2019-2020)

PHÂN TÍCH CẢM XÚC CỦA MỘT ĐOẠN BÌNH LUẬN VỀ ĐỒ ĂN

Sinh viên thực hiện			
STT	Họ tên	MSSV	Lớp
1	Nguyễn Phúc Đạt	18520573	CS114.K21.KHTN

Thành phố Hồ Chí Minh, tháng 8 năm 2020

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN CUỐI KỲ

Môn học: Máy học

Học kỳ II (2019-2020)

PHÂN TÍCH CẢM XÚC CỦA MỘT ĐOẠN BÌNH LUẬN VỀ ĐỒ ĂN

Sinh viên thực hiện			
STT	Họ tên	MSSV	Lớp
1	Nguyễn Phúc Đạt	18520573	CS114.K21.KHTN

Thành phố Hồ Chí Minh, tháng 8 năm 2020

MỤC LỤC

CHƯƠNG I: GIỚI THIỆU

1. Đặt vấn đề.

Với sự phát triển của các trang buôn bán điện tử ngày nay thì việc thu thập đánh giá từ các bình luận của người dùng là rất cần thiết. Các bình luận của người dùng giúp cho các trang web dễ dàng lọc, đề xuất các mặt hàng, địa điểm phù hợp với từng người dùng và đánh giá chất lượng dịch vụ của các đối tác. Với hàng chục nghìn cho tới hàng trăm nghìn bình luận mỗi ngày hiện nay thì việc phân loại (tốt, xấu) cho các bình luận từ người dùng không phải là điều dễ dàng cần đòi hỏi rất nhiều nhân lực. Cùng sự phát triển của AI, hiện nay chúng ta có thể giải quyết bài toán này bằng các thuật toán và mô hình học sâu với độ chính xác cao tương đương với con người.

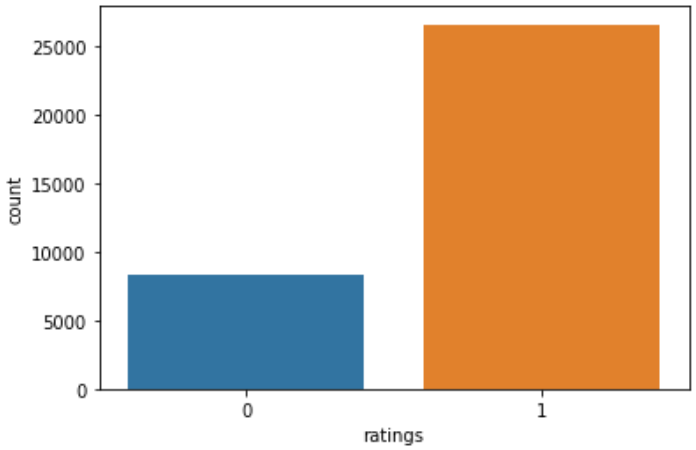
2. Bài toán cụ thể.

Trong assignment này, bài toán được tiếp cận là phân tích cảm xúc (Sentiment Analysis) thông qua tập dữ liệu foody chứa các bình luận được crawl trực tiếp từ https://www.foody.vn/. Nếu nhìn theo kiểu black box, đầu vào (input) của bài toán là một câu hoặc đoạn văn bản và đầu ra (output) là trạng thái tích cực, tiêu cực hay trung hoà (positive - negative - neutral). Tuy nhiên trong phạm vi của assignment này, ta chỉ có hai trạng thái cảm xúc được quan tâm đến là positive và negative.

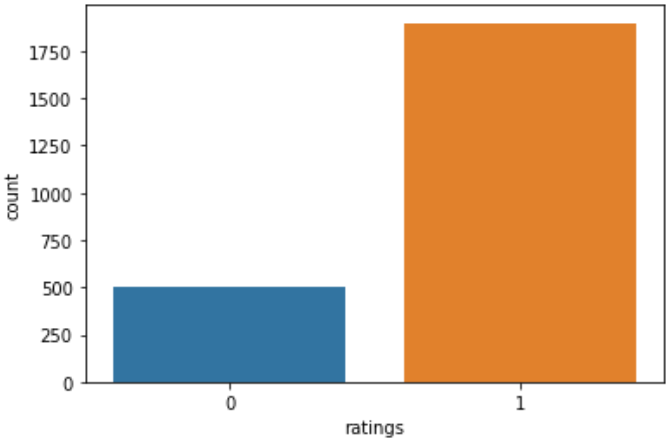
CHƯƠNG II: Dữ liệu

1. Chuẩn bị dữ liệu

Dataset của bài toán này chứa 61.870 bình luận và điểm đánh giá của mỗi bình luận đó trên thang điểm từ 0.0 đến 10.0, dữ liệu được crawl từ hai thành phố lớn là Hồ Chí Minh và Hà Nội . Tuy nhiên, để bộ dữ liệu phù hợp với bài toán là đánh giá cảm xúc tiêu cực hay tích cực, nên trong bước tiền xử lý dữ liệu, các bình luận được đánh giá >= 8.0 thì sẽ được gán nhãn tích cực (1), các bình luận được đánh giá <= 5.0 thì được gán nhãn tiêu cực (0). Do đó số lượng bình luận trong bộ dữ liệu sẽ bị giảm xuống còn 34.985 bình luận bao gồm 8366 bình luận tiêu cực và 26619 bình luận tích cực.



Dữ liệu dùng để đánh giá overfit được crawl từ Quảng Nam chứa 2399 bình luận và điểm đánh giá của mỗi bình luận, bao gồm 500 bình luận tiêu cực và 1899 bình luận tích cực.



Để thu thập được dữ liệu từ foody, Web Scraper – công cụ của Chrome Extension được sử dụng để crawl dữ liệu mà không cần dùng đến code. Web Scraper hỗ trợ thu thập dữ liệu bằng cách kéo thả các component của trang web.

2. Thống kê dữ liệu sử dụng

Nhìn vào dataset của bài toán ta có thể thấy sự mất cân bằng trầm trọng xảy ra giữa hai class. Ta sử dụng kỹ thuật undersampling để khử imbalance, tức chỉ chọn ra vài phần tử của class trội hơn và kết hợp với class còn lại để làm dữ liệu training.

CHƯƠNG III: Tiền xử lý dữ liệu

1. **Loại bỏ các ký tự lặp lại**

   Để thể hiện cảm xúc của mình qua một từ, người dùng có thể kéo dài ký tự trong từ đó. Ta phải bỏ các ký tự lặp lại để các câu, các từ được thống nhất với nhau.

   Ví dụ: “Món này ngon quáaaaa !!” □ “Món này ngon quá !!”

2. **Chuyển về in thường (lowercase)**

   Việc đưa dữ liệu về chữ viết thường là rất cần thiết. Bởi vì đặc trưng này không có tác dụng ở bài toán phân loại văn bản. Đưa về chữ

viết thường giúp giảm số lượng đặc trưng (vì máy tính hiểu hoa thường là 2 từ khác nhau) và tăng độ chính xác hơn cho mô hình.

3. **Chuẩn hóa dấu câu và các từ cảm xúc (sentiment word) trong Tiếng Việt.**

Chuẩn hóa dấu câu: Kiểu gõ dấu khác nhau thì bạn nhìn mắt thường cũng sẽ thấy được sự khác nhau: `òa` và `oà` lần lượt là kiểu gõ cũ (phổ biến hơn) và kiểu gõ mới.

Chuẩn hóa các sentiment words trong Tiếng Việt: "okie, okay, oke" □ "ok", "ko, k, kh, kô" □ "không".

4. **Chuẩn hóa các emojis về hai loại: tích cực và tiêu cực.**

Các emoji được chuẩn hóa sẽ được chọn dựa trên các emojis được sử dụng nhiều như emojis mang ý nghĩa tích cực (positive): "☺", "❤", "♡", "♥", "😂", … và emojis mang nghĩa tiêu cực (negative): "☹", "😭", "😒", "😡", …

Các emojis được lựa chọn dựa trên sentiment score từ trang sentiment raking với sentiment score >= 0.2 là emoji tích cực, sentiment score < 0 là emoji tiêu cực.

5. **Loại bỏ các ký tự trong HTML.**

Dữ liệu được thu thập từ các website đôi khi vẫn còn sót lại các đoạn mã HTML. Các mã HTML code này là rác, chẳng những không có tác dụng cho việc phân loại mà còn làm kết quả phân loại văn bản bị kém đi.

6. **Loại bỏ các punctuations.**

Như đã đề cập ở trên, tiền xử lý bao gồm việc loại bỏ các dữ liệu không có tác dụng cho việc phân loại văn bản. Việc này giúp.

- Giảm số chiều đặc trưng, tăng tốc độ học và xử lý
- Tránh làm ảnh hưởng xấu tới kết quả của mô hình

Các dấu chấm, chấm phẩy, mở ngoặc đơn, đóng ngoặc đơn và các ký tự đặc biệt khác không giúp bạn phân loại một văn bản thuộc cảm xúc nào. Do đó, chúng ta nên loại bỏ nó đi.

7. **Tách từ**

Đơn vị từ trong tiếng Việt bao gồm từ đơn (yêu) và từ ghép (học sinh). Nên chúng ta cần phải nói cho mô hình học máy biết đâu là từ đơn, đâu là từ ghép. Nếu không thì từ nào cũng sẽ là từ đơn hết.

Bởi vì mô hình của chúng ta sẽ coi các từ là đặc trưng, tách nhau theo dấu cách. Do đó, chúng ta phải nối các từ ghép lại thành một từ để không bị tách sai.

Bài toán này là một bài toán cơ sở trong NLP – bài toán tách từ (word segmentation). Thật may là hiện nay có khá nhiều thư viện mã nguồn mở của bài toán này. Do đó, chúng ta chỉ việc cài đặt và sử dụng.

Ví dụ: "Học sinh học sinh học" □ "Học_sinh học sinh_học"

Underthesea là thư viện xử lý ngôn ngữ tự nhiên cho Tiếng Việt được sử dụng cho việc tách từ trong bài toán này.

8. **Tăng cường dữ liệu (data augmentation)**

   Augmentation data bằng cách thêm vào các sample của chính tập train nhưng không dấu. (Bình luận không dấu khá phổ biến).

   Ví dụ : "Món này ngon quá" □ "Mon nay ngon qua"

9. **Những khó khăn**

   Khi xử lý với dữ liệu được crawl trực tiếp về, trong bộ dữ liệu có lẫn những bình luận tiếng Anh, tiếng Hàn, tiếng Nhật. Cùng với đó là những bình luận bị dán sai nhãn, ví dụ như: " Chỗ này đẹp quá!!" nhưng lại gán cho điểm số chỉ có 5.0.

CHƯƠNG IV: Trích chọn đặc trưng

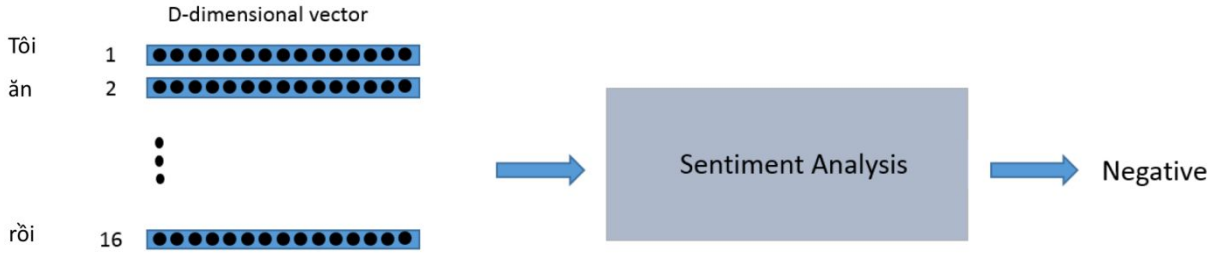
Trong project này, bài toán sẽ được tiếp cận bằng hai cách: TF-IDF và Word Embeddings. Với TF-IDF, bài toán sẽ được huấn luyện bằng các mô hình thuật toán đơn giản của thư viện sklearn như: MultinomialNB, Logistic Regression, LinearSVC. Còn đối với Word Embeddings, bài toán sẽ được huấn luyện dựa trên một neural network tự xây dựng đơn giản sử dụng thư viện Keras của tensorflow.

1. **TF-IDF(Term Frequency – Inverse Document Frequency)**

   TF-IDF là 1 kĩ thuật sử dụng trong khai phá dữ liệu văn bản. Trọng số này được sử dụng để đánh giá tầm quan trọng của một từ trong một văn bản. Giá trị cao thể hiện độ quan trọng cao và nó phụ thuộc vào số lần từ xuất hiện trong văn bản nhưng bù lại bởi tần suất của từ đó trong tập dữ liệu. Một vài biến thể của tf-idf thường được sử dụng trong các hệ thống tìm kiếm như một công cụ chính để đánh giá và sắp xếp văn bản dựa vào truy vấn của người dùng. Tf-idf cũng được sử dụng để lọc những từ stopwords trong các bài toán như tóm tắt văn bản và phân loại văn bản.

2. **Word Embeddings**

   Nếu như chúng ta giữ nguyên định dạng đầu vào là chuỗi ký tự thì rất khó để thực hiện các thao tác biến đổi như tích vô hướng (dot product) hoặc các thuật toán trên mạng neural network như backpropagation. Thay vì dữ liệu đầu vào là một chuỗi, chúng ta cần chuyển đổi các từ trong tập từ điển sang dạng vector số học trong đó có thể thực hiện được các phép toán nêu trên.

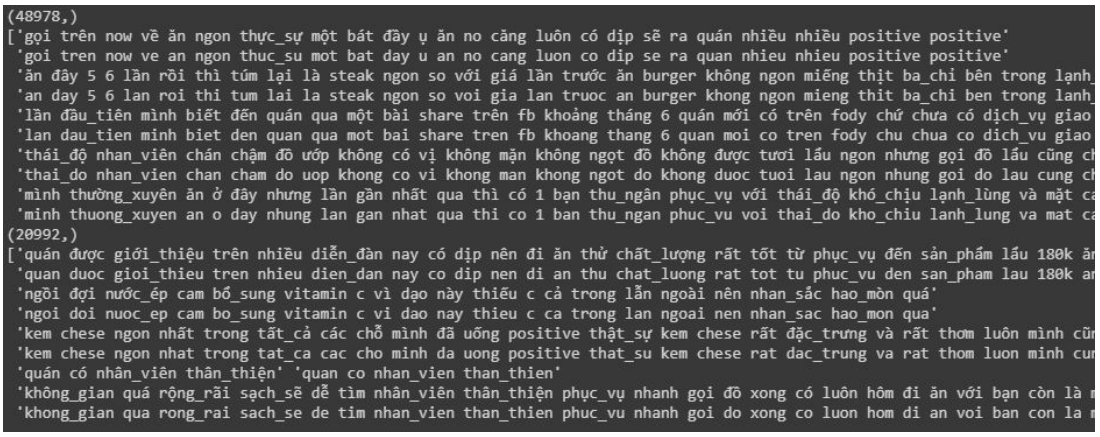


Trong hình minh hoạ ở trên, ta có thể hình dung dữ liệu đầu vào của thuật toán phân tích cảm xúc là một ma trận 16 x D chiều. Trong đó 16 là số lượng từ trong câu và D là số chiều của không gian vector để biểu diễn từ. Để ánh xạ từ một từ sang một vector, chúng ta sử dụng ma trận word embedding như đã thực hiện.

CHƯƠNG V: Xây dựng mô hình phân loại cảm xúc

1. Xây dựng tập train/validation

Sử dụng thư viện sklearn để chia tập train/validation tỉ lệ 70:30



Số lượng dữ liệu của tập train là 48978 và 10 bình luận đầu tiên của tập train được in ra bên dưới.

Số lượng dữ liệu của tập validation là 20992 và 10 bình luận đầu tiên của tập validation được in ra bên dưới.

2. MultinomialNB

MultinomialNB triển khai thuật toán Naive Bayes cho dữ liệu được phân phối đa thức và là một trong hai biến thể Naive Bayes cổ điển được sử dụng trong phân loại văn bản (trong đó dữ liệu thường được biểu diễn dưới dạng số lượng vector từ, mặc dù vector tf-idf cũng được biết là hoạt động tốt trong thực tế).

3. Logistic Regression

Logistic Regression là giải thuật tuyến tính được sử dụng khá phổ biến trong các bài toán phân loại 2 lớp, với activation function được tính

bằng hàm sigmoid sẽ cho ra kết quả trong khoảng từ 0.0 đến 1.0. Đặc biệt phù hợp với bài toán phân tích cảm xúc của 2 lớp tích cực và tiêu cực.

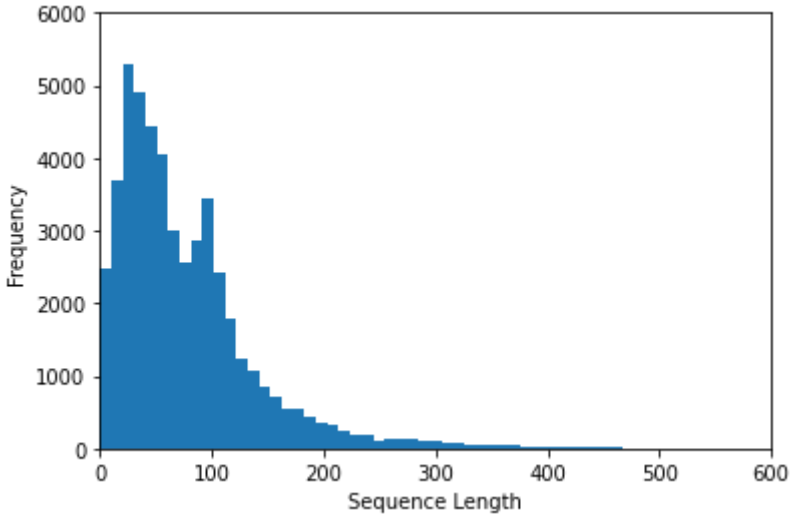
4. **LinearSVC**

Mục tiêu của Linear SVC (Support Vector Classifier) là phù hợp với dữ liệu cung cấp, trả về siêu mặt phẳng (hyperplane) "phù hợp nhất" để phân chia hoặc phân loại dữ liệu đó.

5. **Xây dựng model deep learning cho word embeddings**

Khảo sát tập dữ liệu: do khối lượng dữ liệu lớn (34985 mẫu), nếu chúng ta chọn số lượng từ tối đa cho một câu (MAX_SEQ_LENGTH) quá cao thì sẽ bị lãng phí khi biểu diễn ở những câu review quá ngắn. Ngược lại, nếu sử dụng số lượng từ tối đa quá ít thì sẽ bị bỏ lỡ những từ quan trọng giúp cho việc phân tích cảm xúc. Sau đây chúng ta sẽ tiến hành khảo sát độ dài của các mẫu dữ liệu huấn luyện.

```
The total number of samples is 48978
The total number of words in the files is 3783702
The average number of words in the files is 77.25309322552982
```



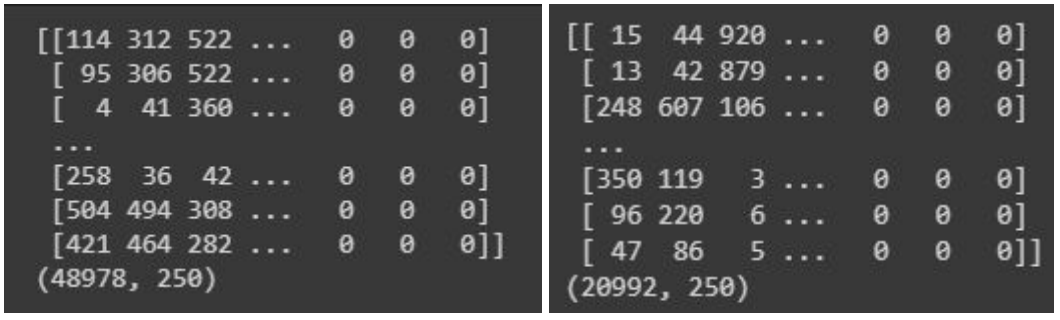
Nhìn vào bảng thống kê có thể thấy đa số các câu chứa khoảng 250 từ trở xuống chiếm phần lớn. Do đó độ dài quy định của các câu là 250 từ.

Khởi tạo các tham số:

```
vocab_size = 10000
embedding_dim = 100
max_length = 250
```

- vocab_size là số lượng từ trong tập từ điển
- embedding_dim là số chiều của embedding, giả sử một từ sẽ được đại diện bằng một vector n chiều, mà n ở đây là 100.
- max_length là số lượng từ trong 1 câu

Tiến hành tokenize và padding cho tập train và tập validation sử dụng thư viện keras của tensorflow. Sau khi hoàn tất thì tập train và tập validation sẽ trông như hình bên dưới



Từ ngữ trong câu đã được thay bằng các con số word index đại diện cho từ đó trong tập từ điển đã được khởi tạo ở trên. Trường hợp khi tokenize cho tập validation, nếu trong tập từ điển không có từ cần thay thế hay nói cách khác là khi tập từ điển gặp một từ mới thì mặc định từ đó sẽ được gán cho index của ký tự OOV (out of vocabulary).

Sau khi xử lý xong dữ liệu, ta sẽ xây dựng mô hình huấn luyện cho bài toán. Mô hình được sử dụng gồm 4 lớp:

- Lớp Embedding: tiến hành tạo khởi tạo vector cho các từ trong trong tập train. Lớp này được khởi tạo với trọng số ngẫu nhiên và sẽ học cách embedding cho tất cả các từ trong tập dữ liệu train.
- Lớp Flatten: sẽ duỗi các vector embedding, biến ma trận 2D thành 1D, chẳng hạn như ma trận embedding có chiều là 250 x 100, lớp Flatten sẽ duỗi ma trận ra thành 25000 x 1.

- Lớp Dense thứ nhất nhận vào 10 input và đưa ra output với hàm activation là relu
- Lớp Dense thứ nhất nhận vào 1 input và đưa ra output với hàm activation là sigmoid, đồng thời cũng là kết quả cuối cùng dự đoán tích cực / tiêu cực trên thang đo từ 0.0 đến 1.0

```
model = tf.keras.Sequential()
model.add(tf.keras.layers.Embedding(vocab_size,
embedding_dim, input_length = max_length))
model.add(tf.keras.layers.Flatten())
model.add(tf.keras.layers.Dense(10, activation="relu"))
model.add(tf.keras.layers.Dense(1, activation="sigmoid"))
model.compile(loss = 'binary_crossentropy', optimizer='adam',
metrics=['accuracy
```

CHƯƠNG VI: Tiến hành training

Áp dụng các thuật toán với tham số mặc định của sklearn.

1. **MultinomialNB**

   Kết quả trên tập dữ liệu imbalance

	precision	recall	f1-score	support
0	0.968	0.547	0.699	4964
1	0.876	0.994	0.932	16028
micro avg	0.889	0.889	0.889	20992
macro avg	0.922	0.771	0.815	20992
weighted avg	0.898	0.889	0.877	20992

   Kết quả trên tập dữ liệu đã khử imbalance với undersampling

	precision	recall	f1-score	support
1	0.860	0.927	0.892	4934
0	0.923	0.854	0.888	5106
micro avg	0.890	0.890	0.890	10040
macro avg	0.892	0.891	0.890	10040
weighted avg	0.892	0.890	0.890	10040

- f1-score trên dữ liệu imbalance (0.877) thấp hơn dữ liệu balance (0.890)
- f1-score của class 0 được cải thiện hơn rất nhiều (0.699 -> 0.888)
- f1-score của class 1 thì bị giảm do sự giảm thiểu dữ liệu trong cách tiếp cận undersampling (0.932 -> 0.892)

2. Logistic Regression

Kết quả trên tập dữ liệu imbalance

	precision	recall	f1-score	support
0	0.914	0.796	0.851	4964
1	0.939	0.977	0.958	16028
micro avg	0.934	0.934	0.934	20992
macro avg	0.926	0.886	0.904	20992
weighted avg	0.933	0.934	0.932	20992

Kết quả trên tập dữ liệu đã khử imbalance với undersampling

	precision	recall	f1-score	support
1	0.887	0.923	0.905	4934
0	0.923	0.886	0.904	5106
micro avg	0.904	0.904	0.904	10040
macro avg	0.905	0.905	0.904	10040
weighted avg	0.905	0.904	0.904	10040

- f1-score trên dữ liệu imbalance (0.932) cao hơn dữ liệu balance (0.904)
- f1-score của class 0 được cải thiện hơn (0.851 -> 0.904)
- f1-score của class 1 thì bị giảm do sự giảm thiểu dữ liệu trong cách tiếp cận undersampling (0.958 -> 0.905)

3. Linear SVC

Kết quả trên tập dữ liệu imbalance

	precision	recall	f1-score	support
0	0.878	0.810	0.843	4964
1	0.942	0.965	0.954	16028
micro avg	0.929	0.929	0.929	20992
macro avg	0.910	0.888	0.898	20992
weighted avg	0.927	0.929	0.928	20992

Kết quả trên tập dữ liệu đã khử imbalance với undersampling

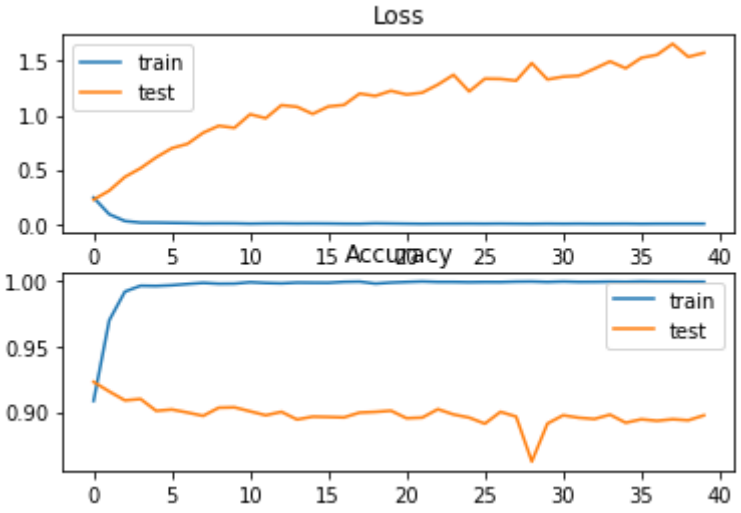
	precision	recall	f1-score	support
1	0.881	0.907	0.894	4934
0	0.907	0.882	0.894	5106
micro avg	0.894	0.894	0.894	10040
macro avg	0.894	0.894	0.894	10040
weighted avg	0.894	0.894	0.894	10040

- f1-score trên dữ liệu imbalance (0.928) cao hơn dữ liệu balance (0.894)
- f1-score của class 0 được cải thiện không đáng kể (0.843 -> 0.894)
- f1-score của class 1 thì bị giảm do sự giảm thiểu dữ liệu trong cách tiếp cận undersampling (0.954 -> 0.894)

4. **Deep learning model cho word embedding**

Sau khi huấn luyện với 40 epochs với dữ liệu imbalance thì kết quả nhận được là:

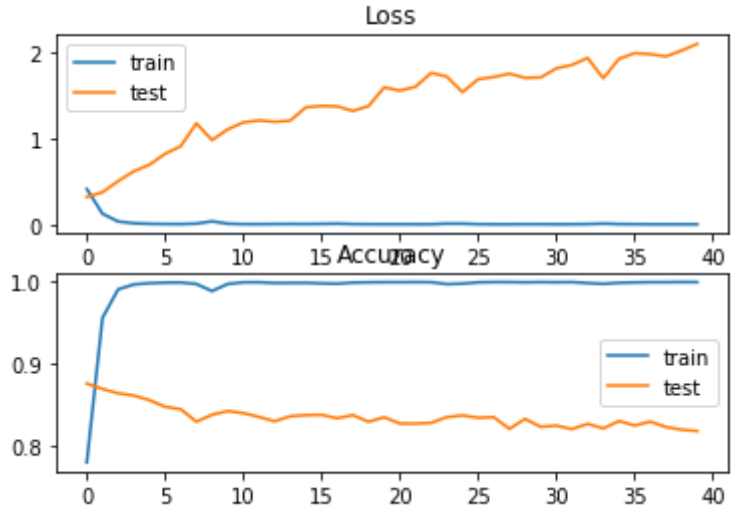
```
loss: 0.0026 - accuracy: 0.9990 - val_loss: 1.5734 -
val_accuracy: 0.8974
```



Nhìn vào biểu đồ loss, ta thấy loss của tập train và tập validation càng chạy xa nhau, trong khi ở biểu đồ accuracy thì khá ổn định đường accuracy của tập train và tập validation không bị biến động nhiều.

Còn với dữ liệu balance thì kết quả nhận được là

```
loss: 0.0017 - accuracy: 0.9993 - val_loss: 2.0965 -
val_accuracy: 0.8175
```



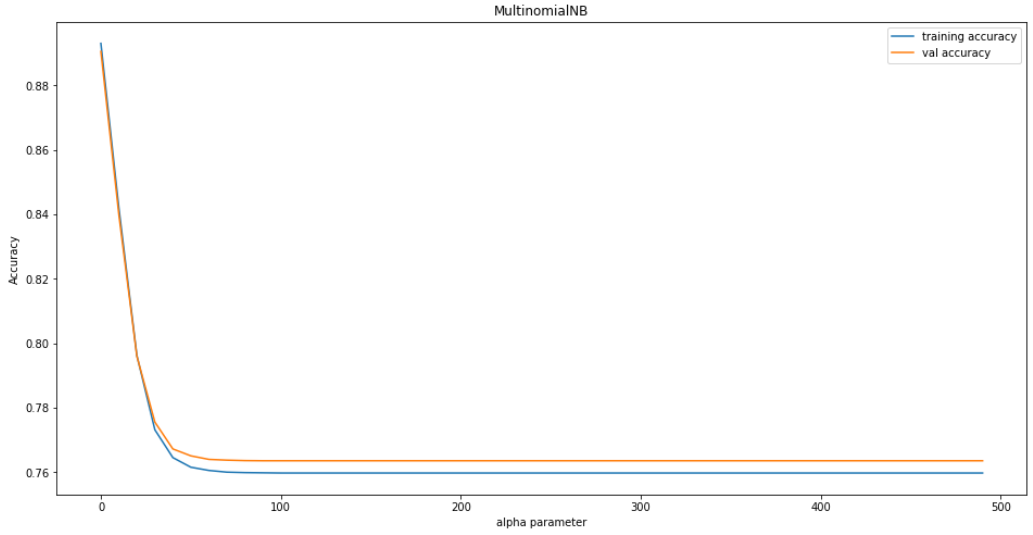
CHƯƠNG VII: Tinh chỉnh tham số

Tinh chỉnh tham số cho ba thuật toán MultinomialNB, Logistic Regression, LinearSVC sử dụng GridsearchCV của sklearn để thử các tham số

1. **MultinomialNB**

   Các tham số quan trọng:

alpha: tham số làm mịn Laplace/Lidstone



Các tham số tốt nhất sau khi tune

```
Best: 0.888678 using {'alpha': 0}
/usr/local/lib/python3.6/dist-packages/sklearn/n
  'setting alpha = %.1e' % _ALPHA_MIN)
```

Kết quả tập validation với tham số tốt nhất (trên tập dữ liệu imbalance)

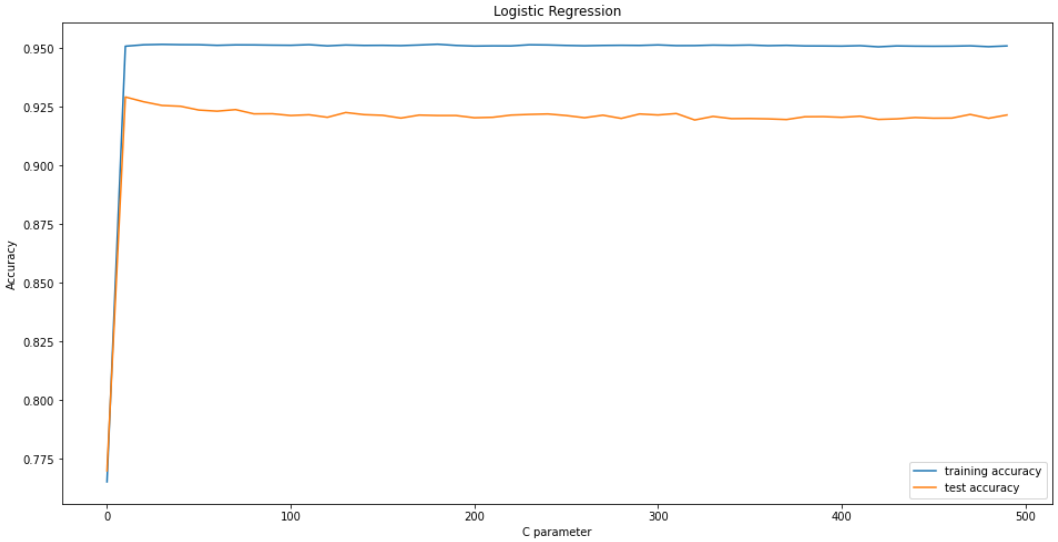
	precision	recall	f1-score	support
0	0.967	0.556	0.706	4964
1	0.878	0.994	0.933	16028
micro avg	0.890	0.890	0.890	20992
macro avg	0.923	0.775	0.819	20992
weighted avg	0.899	0.890	0.879	20992

2. Logistic Regression

C: tham số của regularization

penalty: lựa chọn regularization.

solver: thuật toán dùng tối ưu hóa vấn đề.



Các tham số tốt nhất sau khi tune

```
Best: 0.933678 using {'C': 10, 'solver': 'newton-cg'}
```

Kết quả tập validation với tham số tốt nhất (trên tập dữ liệu imbalance)

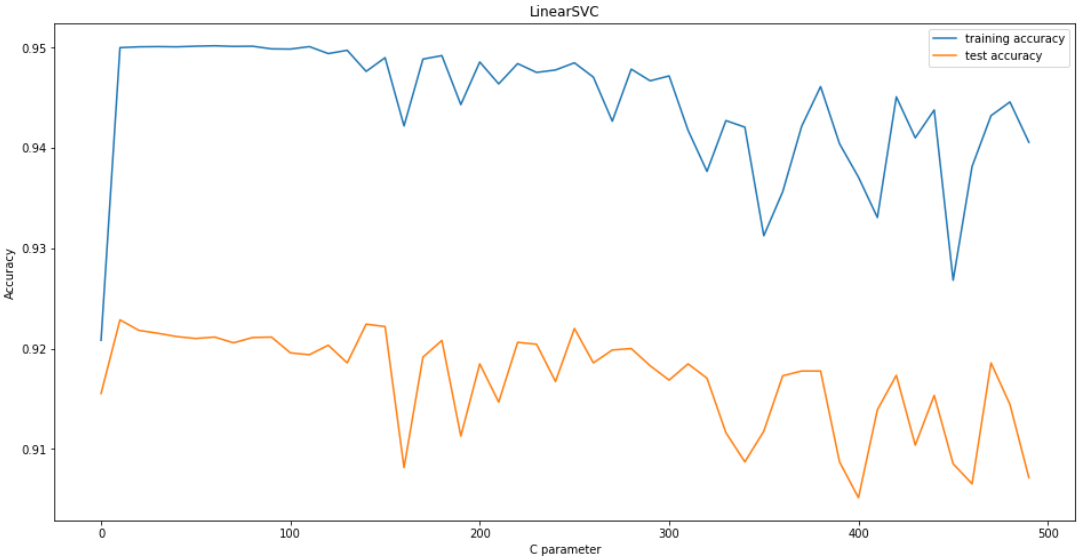
	precision	recall	f1-score	support
0	0.881	0.812	0.845	4964
1	0.943	0.966	0.954	16028
micro avg	0.930	0.930	0.930	20992
macro avg	0.912	0.889	0.900	20992
weighted avg	0.928	0.930	0.929	20992

3. LinearSVC

C: tham số của regularization

penalty: lựa chọn regularization.

solver: thuật toán dùng tối ưu hóa vấn đề.



Các tham số tốt nhất sau khi tune

```
Best: 0.929234 using {'C': 10, 'multi_class': 'crammer_singer'}
/usr/local/lib/python3.6/dist-packages/sklearn/svm/base.py:931: Co
  "the number of iterations.", ConvergenceWarning)
```

Kết quả tập validation với tham số tốt nhất (trên tập dữ liệu imbalance)

	precision	recall	f1-score	support
0	0.884	0.809	0.845	4964
1	0.942	0.967	0.955	16028
micro avg	0.930	0.930	0.930	20992
macro avg	0.913	0.888	0.900	20992
weighted avg	0.929	0.930	0.929	20992

Kết quả của các thuật toán sau khi tinh chỉnh cải thiện không đáng kể.

CHƯƠNG VIII: Kết quả, đánh giá trên tập test

Sử dụng các tham số tốt nhất của mỗi thuật toán để đánh giá trong tập dữ liệu test Quảng Nam, kết quả lần lượt nhận được theo thứ tự MultinomialNB, Logistic Regression, LinearSVC, Deep learning model là:

- **Dữ liệu imbalance**

	precision	recall	f1-score	support
0	0.986	0.500	0.664	1000
1	0.883	0.998	0.937	3798
micro avg	0.894	0.894	0.894	4798
macro avg	0.935	0.749	0.800	4798
weighted avg	0.905	0.894	0.880	4798

	precision	recall	f1-score	support
0	0.919	0.789	0.849	1000
1	0.946	0.982	0.964	3798
micro avg	0.941	0.941	0.941	4798
macro avg	0.932	0.885	0.906	4798
weighted avg	0.941	0.941	0.940	4798

	precision	recall	f1-score	support
0	0.874	0.811	0.841	1000
1	0.951	0.969	0.960	3798
micro avg	0.936	0.936	0.936	4798
macro avg	0.913	0.890	0.901	4798
weighted avg	0.935	0.936	0.935	4798

Và kết quả của custom model trên dữ liệu test Quảng Nam là:

	precision	recall	f1-score	support
0	0.752	0.718	0.735	1000
1	0.927	0.938	0.932	3798
micro avg	0.892	0.892	0.892	4798
macro avg	0.839	0.828	0.833	4798
weighted avg	0.890	0.892	0.891	4798

- **Dữ liệu balance**

	precision	recall	f1-score	support
0	0.766	0.836	0.800	1000
1	0.956	0.933	0.944	3798
micro avg	0.913	0.913	0.913	4798
macro avg	0.861	0.884	0.872	4798
weighted avg	0.916	0.913	0.914	4798

	precision	recall	f1-score	support
0	0.727	0.887	0.799	1000
1	0.968	0.912	0.940	3798
micro avg	0.907	0.907	0.907	4798
macro avg	0.848	0.900	0.869	4798
weighted avg	0.918	0.907	0.910	4798

	precision	recall	f1-score	support
0	0.702	0.877	0.780	1000
1	0.965	0.902	0.933	3798
micro avg	0.897	0.897	0.897	4798
macro avg	0.834	0.890	0.856	4798
weighted avg	0.910	0.897	0.901	4798

Và kết quả của custom model trên dữ liệu test Quảng Nam là:

	precision	recall	f1-score	support
0	0.654	0.819	0.727	1000
1	0.949	0.886	0.916	3798
micro avg	0.872	0.872	0.872	4798
macro avg	0.801	0.852	0.822	4798
weighted avg	0.887	0.872	0.877	4798

Kết quả cho thấy thuật toán Logistic Regression phân loại rất tốt khi đều thể hiện chỉ số f1-score khá cao khi được huấn luyện trên cả 2 tập dữ liệu (imbalance và balance).
f1-score của thuật toán MultinomialNB cải thiện rất nhiều sau khi đã khử imbalance cho tập dữ liệu, từ 0.88 tăng lên đến 0.914.
Còn về phía Linear SVC và deep learning model, f1-score của cả hai đều giảm khi model được được huấn luyện với dữ liệu balance.

TÀI LIỆU THAM KHẢO:

MultinomialNB.

https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html

Logistic Regression.
https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
Linear SVC
https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html
Word Embedding.
https://machinelearningmastery.com/what-are-word-embeddings/