TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA TOÁN – CƠ – TIN HỌC

BÁO CÁO THỰC TẬP

Đề tài:

Sentiment Analysis Vietnamese

Môn: Phân tích thiết kế hệ thống thông tin

Người hướng dẫn: Phạm Thế Quyền Cty: Aimesoft
Người thực hiện: Đỗ Thị Diệu Thúy MSV: 16002591

Hà Nội, tháng 5 năm 2020

MỤC LỤC

I. Lời mở đầu

Chúng ta đang sống trong kỷ nguyên số, đặc biệt những năm gần đây nổi lên mạng xã hội, với hàng triệu người dùng trên thế giới, lượng thông tin nội dung được người dùng tạo ra hằng ngày cực kỳ lớn, với đa dạng các hình thức như dòng trạng thái, hình ảnh, video. Mạng xã hội có những đặc điểm là: thông tin do người dùng tạo ra, mang tính cá nhân cho nên chất lượng nội dung hay tính đúng đắn, xác thực là tương đối; một thông tin mới được tạo lại có sức lan tỏa nhanh đến đông đảo các người dùng khác, so với các kênh thông tin truyền thống như truyền hình, truyền thanh, báo chí, diễn đàn, blog...

Điều này đặt ra cho các doanh nghiệp lớn giải quyết bài toán quản trị thương hiệu doanh nghiệp, quản trị thương hiệu sản phẩm trước các dư luận không tốt trên mạng xã hội rất khó khăn, cả về nguồn xuất phát thông tin, cả về khối lượng thông tin cần xử lý. Chưa kể việc các đối thủ cạnh tranh trên thương trường lợi dụng mạng xã hội để cố ý tạo các thông tin bất lợi cho nhau. Điều này đòi hỏi phải có một công cụ hỗ trợ đắc lực, mà chỉ có áp dụng công nghệ thông tin mới giải quyết được, chứ không lực lượng con người nào có thể làm xuể.

Các doanh nghiệp lớn của Việt Nam như: Vinamilk, các hãng hàng không như VietNam AirLines, Vietjest Art, Jetstar Pacific Airlines… hiện nay đã đặt hàng các doanh nghiệp công nghệ thông tin Việt Nam giải quyết vấn đề này. Giải pháp công nghệ hiện nay được gọi là "lắng nghe mạng xã hội", tức là các doanh nghiệp CNTT mua các dữ liệu thời gian thực (real time) từ các công ty mạng xã hội về để xử lý các thông tin liên quan đến doanh nghiệp hay các sản phẩm mà doanh nghiệp đó kinh doanh, nhằm phát hiện và ngăn chặn sớm sự lan rộng các thông tin bất lợi trên mạng xã hội, có hình thức đính chính phản hồi đến các khách hàng của mình, đồng thời thương lượng, ngăn chặn tận gốc những người tạo ra các nội dung đó.

Điều cốt yếu của giải pháp này chính là phân tích tình cảm của các dòng trạng thái trên mạng xã hội nhằm lọc ra các thông tin bất lợi để xử lý.

Phân tích tình cảm (Sentiment analysis) là nhằm phát hiện ra thái độ mang tính lâu dài, màu sắc tình cảm, khuynh hướng niềm tin vào các đối tượng hay người nào đó. Phát biểu theo góc nhìn của học máy thì nó 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 của N lớp cảm xúc mà ta cần xác định. Nó được ứng dụng trong hàng loạt các vấn đề như: phân tích thị trường, dự đoán biến động giá cổ phiếu, quản trị thương hiệu doanh nghiệp, thương hiệu sản phẩm, khảo sát ý kiến xã hội học, khảo sát sự hài lòng của khách hàng, phân tích trạng thái tâm lý con người...

Trong bài viết này, tôi xin được trình bày về một hệ thống đơn giản và quá trình tôi thực hiện. Đó là hệ thống phân tích cảm xúc trong văn bản thành 2 lớp: tích cực và tiêu cực, với bộ dữ liệu huấn luyện gồm khoảng 3000 câu bình luận trích xuất trực tiếp trên một trang web bán hàng.

II. Giới thiệu chung

1. Giới thiệu về công ty thực tập

Công ty Cổ phần Aimesoft là công ty công nghệ chuyên nghiên cứu, phát triển, phân phối và cung cấp dịch vụ liên quan đến phần mềm Trí tuệ nhân tạo đa thể thức. Aimesoft là công ty đầu tiên ở Việt Nam phát triển trí tuệ nhân tạo đa thể thức (Multimodal AI). Công nghệ trí tuệ nhân tạo đa thể thức của Aimesoft cho phép kết hợp và bổ trợ lẫn nhau giữa các thuật toán đơn lẻ như: Xử lý ảnh, khai phá dữ liệu, xử lý ngôn ngữ tự nhiên, xử lý tiếng nói và hợp nhất thông tin (Information Fusion), để giải quyết các vấn đề khó với độ chính xác cao. Các bài toán này thường không giải quyết được nếu chỉ dùng một trong các nguồn dữ liệu và thuật toán xử lý đơn lẻ.

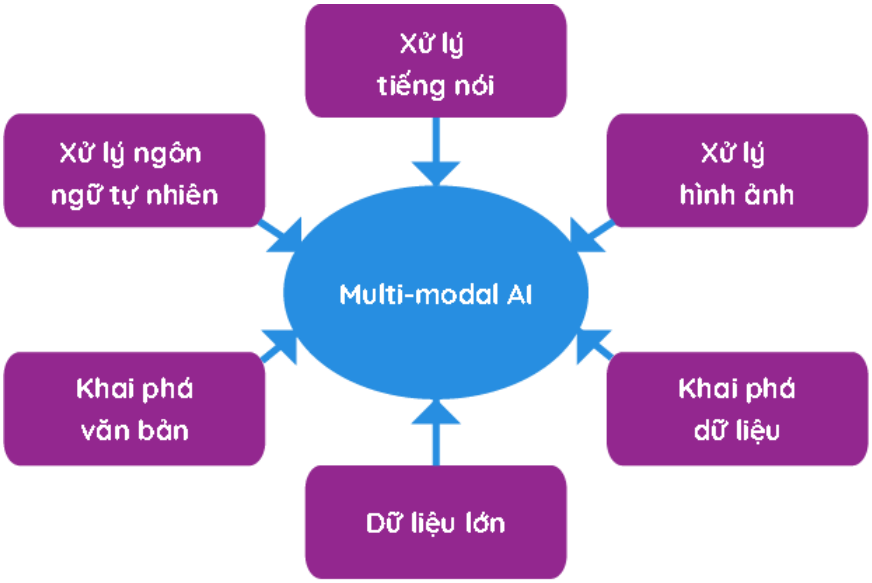
Tài sản lớn nhất của Aimesoft là đội ngũ hơn 30 chuyên gia và kĩ sư nghiên cứu với nền tảng lập trình vững chắc và kinh nghiệm thực chiến các dự án "khó nhằn". Dưới dự dẫn dắt của TS. Nguyễn Tuấn Đức (CEO) và cố vấn của các chuyên gia đến từ Đại học Tokyo, Nhật Bản, Aimesoft đang có những bước chuyển mình mạnh mẽ nhằm khẳng định vị trí số 1 về cung cấp giải pháp ứng dụng Trí tuệ nhân tạo.

Các công nghệ và thuật toán mà Aimesoft sử dụng để xây dựng Trí tuệ nhân tạo đa thể thức:

- ✓ Công nghệ Xử Lý Ảnh
  Phân mảnh, lý giải hình ảnh, nhận diện khuôn mặt, nhận diện tuổi, giới tính, nhận diện chữ viết, tìm kiếm dựa trên hình ảnh, phân tích dự đoán dựa trên ảnh
- ✓ Công nghệ Xử lý ngôn ngữ tự nhiên
  Tách từ, gán nhãn từ loại, tìm từ khoá, từ gần nghĩa, cùng nghĩa, đối

lập ngữ nghĩa, phân tích và trích xuất thông tin, trích xuất quan hệ, tìm kiếm dựa trên quan hệ, lý giải ngôn ngữ tự nhiên

- ✓ Công nghệ Xử lý tiếng nói
  Xây dựng mô hình ngôn ngữ, mô hình âm học, tự động nhận diện hot word (trigger word), lọc nhiễu.
- ✓ Công nghệ Khai phá dữ liệu
  Thu thập và xử lý dữ liệu rất lớn, dự đoán KPI, phân tích dự đoán dựa trên dữ liệu lớn, tự động sinh dữ liệu cho học máy.



2. Giới thiệu về đề tài được giao

Đề tài: Sentiment analysis Vietnamese

Phân tích cảm xúc (Sentiment analysis) là nhằm phát hiện ra thái độ mang tính lâu dài, màu sắc tình cảm, khuynh hướng niềm tin vào các đối tượng hay người nào đó.

Các vấn đề xung quanh việc phân tích cảm xúc:

- Nguồn gốc của cảm xúc.
- Mục tiêu của cảm xúc.
- Các loại cảm xúc: thích, yêu, ghét, đánh giá, mong mỏi…
- Về mức độ cảm xúc: tích cực, tiêu cực, trung tính.
- Văn bản hàm chứa cảm xúc: một câu hoặc một đoạn văn bản.

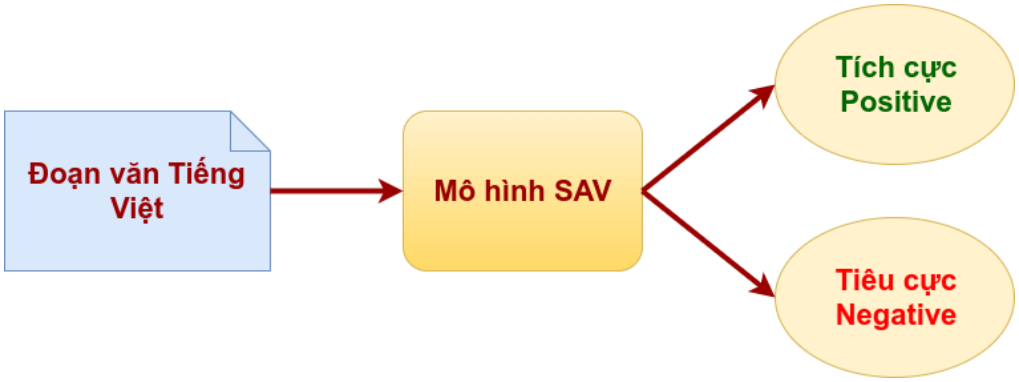
Bài toán phân tích cảm xúc thuộc dạng bài toán phân tích ngữ nghĩa văn bản. Vì vậy, ta cần phải xây dựng một mô hình để hiểu được ý nghĩa của câu văn, đoạn văn để quyết định xem câu văn đó hoặc đoạn văn đó mang màu sắc cảm xúc chủ đạo nào.

Phát biểu theo góc nhìn của máy học (Machine Learning) thì phân tích cảm xúc 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.

Trong loại bài toán phân tích cảm xúc được phân thành các bài toán có độ khó khác nhau như sau:

* Đơn giản: Phân tích cảm xúc (thái độ) trong văn bản thành 2 lớp: tích cực (positive) và tiêu cực (negative).
* Phức tạp hơn: Xếp hạng cảm xúc (thái độ) trong văn bản từ 1 đến 5.
* Khó: Phát hiện mục tiêu, nguồn gốc của cảm xúc (thái độ) hoặc các loại cảm xúc (thái độ) phức tạp.

Hiện tại thì cộng đồng khoa học mới chỉ giải quyết tốt bài toán phân tích cảm xúc ở cấp độ đơn giản, tức là phân tích cảm xúc với 2 lớp cảm xúc tiêu cực và tích cực. Vì vậy, bài toán phân tích cảm xúc trong Tiếng Việt trình bày trong bài viết này là kết quả của nghiên cứu phân tích cảm xúc văn bản Tiếng Việt với 2 lớp cảm xúc là: tiêu cực (negative) và tích cực (positive). Sơ đồ phân tích cảm xúc như sau:



Đầu vào của mô hình xử lý Sentiment Analysis Vietnamese (SAV) là một đoạn văn Tiếng Việt, đầu ra là 2 giá trị mà đoạn văn đầu vào thuộc về lớp cảm xúc: tiêu cực (negative) hay tích cực (positive).

3. Giới thiệu về ngôn ngữ, công cụ lập trình

a) Ngôn ngữ Python

Python là một ngôn ngữ lập trình bậc cao cho các mục đích lập trình đa năng, do Guido van Rossum tạo ra và lần đầu ra mắt vào năm 1991. Python được thiết kế với ưu điểm mạnh là dễ đọc, dễ học và dễ nhớ. Python là ngôn ngữ có hình thức rất sáng sủa, cấu trúc rõ ràng. Cấu trúc của Python còn cho phép người sử dụng viết mã lệnh với số lần gõ phím tối thiểu

Đặc điểm:

- Vừa hướng thủ tục (procedural-oriented), vừa hướng đối tượng (object-oriented)
- Hỗ trợ module và hỗ trợ gói (package)
- Xử lý lỗi bằng ngoại lệ (Exception)
- Kiểu dữ liệu động ở mức cao.
- Có các bộ thư viện chuẩn và các module ngoài, đáp ứng tất cả các nhu cầu lập trình.

- Có khả năng tương tác với các module khác viết trên C/C++ (Hoặc Java cho Jython, hoặc .Net cho IronPython).
- Có thể nhúng vào ứng dụng như một giao tiếp kịch bản (scripting interface).

Ưu điểm:

- ✓ Là một ngôn ngữ có hình thức sáng sủa, cấu trúc rõ ràng, cú pháp ngắn gọn
- ✓ Có trên tất cả các nền tảng hệ điều hành từ UNIX, MS – DOS, Mac OS, Windows và Linix và các OS khác thuộc họ Unix.
- ✓ Tương thích mạnh mẽ với Unix, hardware, thirt-party software với số lượng thư viện khổng lồ (400 triệu người sử dụng)
- ✓ Python với tốc độ xử lý cực nhanh, python có thể tạo ra những chương trình từ những script siêu nhỏ tới những phần mềm cực lớn như Biender 3D.

Với những đặc điểm trên, python đang là ngôn ngữ được rất nhiều người lựa chọn cho việc lập trình AI. Hơn nữa, Python còn có một số lượng lớn các thư viện hữu ích có thể được sử dụng trong AI. Ví dụ: Numpy, pandas, gensim , BeautifulSoup, matplotlib, Scypy và nhiều thư viện dành cho máy học như: sklearn, Tensorflow , Keras, NTLK….

b) Công cụ lập trình

❖ Visual Studio Code

Visual Studio Code là một trình biên tập mã được phát triển bởi Microsoft dành cho Windows, Linux và macOS. Visual Studio Code hay viết tắt là VS Code hay VSC lần đầu tiên được Microsoft giới thiệu vào ngày 29 tháng 4 năm 2015 tại Build Conference diễn ra tại San Francisco. Microsoft đã thiết kế VS Code như một trình soạn thảo mã nguồn đa nền tảng để viết các ứng dụng web và Cloud. Nó hỗ trợ nhiều ngôn ngữ và chức năng tùy vào ngôn ngữ sử dụng. Đặc điểm của nó là code rất nhanh, rất nhẹ và cũng rất mạnh nên nó là một trong những trình soạn thảo mã nguồn phổ biến nhất được sử dụng bởi các lập trình viên

Trình soạn thảo hay bị nhầm lẫn với IDE. Tuy nhiên, có một số khác biệt nhất định giữa trình soạn thảo và IDE là:

- Một IDE sẽ cung cấp môi trường hoàn chỉnh cho các lập trình viên để lập trình, giúp họ làm việc hiệu quả hơn. Chủ yếu, nó bao gồm một trình soạn thảo mã nguồn, trình biên dịch và trình gỡ lỗi cùng với rất nhiều các tính năng khác.
- Trong khi đó, một trình soạn thảo cung cấp ít chức năng hơn, ít môi trường để chạy, kiểm tra và debug code chung một nơi. Nhưng các trình soạn thảo này rất nhẹ, ngốn ít RAM hơn và một số trình soạn thảo như Visual Studio Code hoặc Sublime Text còn đi kèm với các tính năng bổ sung tương tự như các IDE.

Ưu điểm Visual Studio Code:

- Visual Studio Code là trình soạn thảo Cross-Platform-soạn thảo đa nền tảng, mã nguồn mở và miễn phí, hoạt động trên Windows, Linux và macOS.
- Visual Studio Code hỗ trợ nhiều ngôn ngữ lập trình như: Python, JavaScript, HTML, CSS, TypeScript, C ++, Java, PHP, Go, C, PHP, SQL, Ruby, Objective-C và thậm chí nhiều hơn thế nữa ...
- Có thể thay đổi ngôn ngữ cho tệp đã chọn trên Visual Studio Code bất cứ khi nào thích.
- Trang web của Visual Studio Code bao gồm các tài liệu dành riêng cho các ngôn ngữ phổ biến mà Visual Studio Code hỗ trợ. Một số trong đó là C ++, C, CSS, Go, Python, PHP, Java …
- Visual Studio Code đi kèm với tính năng Debug tích hợp sẵn, giúp tăng tốc bất kỳ chỉnh sửa vòng lặp nào, biên dịch và Debug.
- Visual Studio Code tích hợp sẵn Git hoàn chỉnh. Tính năng này giúp các lập trình viên thấy được các thay đổi ngay lập tức mà không cần rời khỏi màn hình làm việc của VCS.
- Tính năng IntelliSense được cung cấp sẵn cho các ngôn ngữ lập trình JavaScript, CSS, HTML, TypeScript, JSON, Sass và Less. Đối với các ngôn ngữ khác, có thể sử dụng IntelliSense bằng cách cài hêm các tiện ích mở rộng của nó.
- VS Code có khả năng tùy biến cao, nhờ các tùy chọn cài đặt linh hoạt và vô số tiện ích mở rộng. VS Code cung cấp cho bạn các tùy chọn để thay đổi theme, thay đổi phím tắt, điều chỉnh cài đặt, tạo snipet và nhiều hơn thế nữa.

❖ Jupyter notebook

Jupyter Notebook là một ứng dụng web mã nguồn mở - một dự án spin-off từ dự án IPython, trước đây từng có một dự án IPython Notebook. Tên Jupyter xuất phát từ các ngôn ngữ lập trình được hỗ trợ cốt lõi mà nó hỗ trợ: Julia, Python và R.

Jupyter notebook cung cấp một ứng dụng dựa trên web phù hợp để nắm bắt toàn bộ quá trình tính toán: phát triển, ghi lại tài liệu và thực thi mã, cũng như truyền đạt kết quả. Jupyter notebook kết hợp cả hai thành phần:

- Một ứng dụng web: một công cụ dựa trên trình duyệt web để lập trình, soạn thảo tương tác các tài liệu kết hợp văn bản giải thích, toán học, tính toán và đầu ra đa phương tiện của chúng.
- Tài liệu sổ ghi chép: đại diện cho tất cả nội dung hiển thị trong ứng dụng web, bao gồm đầu vào và đầu ra của các tính toán, văn bản giải thích, toán học, hình ảnh và biểu diễn đa phương tiện của các đối tượng.

III. Cơ sở lý thuyết

1. Một số mô hình học máy

a) Mô hình Naïve Bayes

Xét bài toán classification với C classes 1,2,3,…,C. Giả sử có một điểm dữ liệu $x \in R^d$. Hãy tính xác suất để điểm dữ liệu này rơi vào class c. Nói cách khác, hãy tính:

$$p(y=c|x) \qquad (1)$$

hoặc viết gọn thành p(c|x).

Tức tính xác suất để đầu ra là class c biết rằng đầu vào là vector x.

Biểu thức này, nếu tính được, sẽ giúp chúng ta xác định được xác suất để điểm dữ liệu rơi vào mỗi class. Từ đó có thể giúp xác định class của điểm dữ liệu đó bằng cách chọn ra class có xác suất cao nhất:

$$c = \arg \max_{c \in \{1,\dots,C\}} p(c|X) \qquad (2)$$

Biểu thức (2) thường khó được tính trực tiếp. Thay vào đó, quy tắc Bayes thường được sử dụng:

$$c = \arg \max_{c} p(c|X) \qquad (3)$$

$$= \arg \max_{c} \frac{p(X|c)p(c)}{p(X)} \qquad (4)$$

$$= \arg \max_{c} p(X|c)p(c) \qquad (5)$$

Từ (3) sang (4) là vì quy tắc Bayes. Từ (4) sang (5) là vì mẫu số p(x) không phụ thuộc vào c.

Tiếp tục xét biểu thức (5), p(c) có thể được hiểu là xác suất để một điểm rơi vào class c. Giá trị này có thể được tính bằng MLE, tức tỉ lệ số điểm dữ liệu trong tập training rơi vào class này chia cho tổng số lượng dữ liệu trong tập traing; hoặc cũng có thể được đánh giá bằng MAP estimation. Trường hợp thứ nhất thường được sử dụng nhiều hơn.

Thành phần còn lại p(x|c), tức phân phối của các điểm dữ liệu trong class c, thường rất khó tính toán vì x là một biến ngẫu nhiên nhiều chiều, cần rất rất nhiều dữ liệu training để có thể xây dựng được phân phối đó. Để giúp cho việc tính toán được đơn giản, người ta thường giả sử một cách đơn giản nhất rằng các thành phần của biến ngẫu nhiên x là độc lập với nhau, nếu biết c (given c.. Tức là:

$$p(X|c) = p(x_1, x_2, \dots, x_d|c) \prod_{i=1}^{d} p(x_i|c) \qquad (6)$$

Giả thiết các chiều của dữ liệu độc lập với nhau, nếu biết cc, là quá chặt và ít khi tìm được dữ liệu mà các thành phần hoàn toàn độc lập với nhau. Tuy nhiên, giả thiết ngây ngô này lại mang lại những kết quả tốt bất ngờ. Giả thiết về sự độc lập của các chiều dữ liệu này được gọi là Naive Bayes (xin không dịch). Cách xác định class của dữ liệu dựa trên giả thiết này có tên là Naive Bayes Classifier (NBC).

NBC, nhờ vào tính đơn giản một cách ngây thơ, có tốc độ training và test rất nhanh. Việc này giúp nó mang lại hiệu quả cao trong các bài toán large-scale.

Ở bước training, các phân phối p(c) và $p(x_i|c)$, i = 1,…, d sẽ được xác định dựa vào training data. Việc xác định các giá trị này có thể dựa vào Maximum Likelihood Estimation hoặc Maximum A Posteriori.

Ở bước test, với một điểm dữ liệu mới xx, class của nó sẽ được xác đinh bởi:

$$c = \arg \max_{c \in \{1,\dots,C\}} p(c) \prod_{i=1}^{d} p(x_i|c) \qquad (7)$$

Khi d lớn và các xác suất nhỏ, biểu thức ở vế phải của (7) sẽ là một số rất nhỏ, khi tính toán có thể gặp sai số. Để giải quyết việc này, (7) thường được viết lại dưới dạng tương đương bằng cách lấy log của vế phải

$$c = \arg \max_{c \in \{1,\dots,C\}} = \log(p(c)) + \sum_{i=1}^{d} \log(p(x_i|c)) \qquad (7.1)$$

Việc này không ảnh hưởng tới kết quả vì log là một hàm đồng biến trên tập các số dương.

Mặc dù giả thiết mà Naive Bayes Classifiers sử dụng là quá phi thực tế, chúng vẫn hoạt động khá hiệu quả trong nhiều bài toán thực tế, đặc biệt là trong các bài toán phân loại văn bản, ví dụ như lọc tin nhắn rác hay lọc email spam. Trong phần sau của bài viết, chúng ta cùng xây dựng một bộ lọc email spam tiếng Anh đơn giản.

Cả việc training và test của NBC là cực kỳ nhanh khi so với các phương pháp classification phức tạp khác. Việc giả sử các thành phần trong dữ liệu là độc lập với nhau, nếu biết class, khiến cho việc tính toán mỗi phân phối $p(x_i|c)$ trở nên cực kỳ nhanh.

Mỗi giá trị $p(c)$,$c=1,2,\ldots,C$ có thể được xác định như là tần suất xuất hiện của class c trong training data.

Việc tính toán $p(x_i|c)$ phụ thuộc vào loại dữ liệu. Có ba loại được sử dụng phổ biến là: Gaussian Naive Bayes, , và Bernoulli Naive. Nhưng trong bài này mình chỉ sử dụng Multinomial Naive Bayes

❖ Multinomial Naive Bayes

Mô hình này chủ yếu được sử dụng trong phân loại văn bản mà feature vectors được tính bằng Bags of Words. Lúc này, mỗi văn bản được biểu diễn bởi một vector có độ dài d chính là số từ trong từ điển. Giá trị của thành phần thứ i trong mỗi vector chính là số lần từ thứ I xuất hiện trong văn bản đó.

Khi đó, $p(x_i|c)$ tỉ lệ với tần suất từ thứ i (hay feature thứ i cho trường hợp tổng quát) xuất hiện trong các văn bản của class cc. Giá trị này có thể được tính bằng cách:

$$\lambda_{ci} = p(x_i|c) = \frac{N_{ci}}{N_c} \qquad (8)$$

Trong đó:

- N_{ci} là tổng số lần từ thứ i xuất hiện trong các văn bản của class c, nó được tính là tổng của tất cả các thành phần thứ i của các feature vectors ứng với class c.
- N_c là tổng số từ (kể cả lặp) xuất hiện trong class c. Nói cách khác, nó bằng tổng độ dài của toàn bộ các văn bản thuộc vào class cc. Có thể suy ra rằng $N_c = \sum_{i=1}^{d} N_{ci}$, từ đó $\sum_{i=1}^{d} \lambda_{ci} = 1$

Cách tính này có một hạn chế là nếu có một từ mới chưa bao giờ xuất hiện trong class c thì biểu thức (8) sẽ bằng 0, điều này dẫn đến vế phải của (7) bằng 0 bất kể các giá trị còn lại có lớn thế nào. Việc này sẽ dẫn đến kết quả không chính xác (xem thêm ví dụ ở mục sau).

Để giải quyết việc này, một kỹ thuật được gọi là Laplace smoothing được áp dụng:

$$\hat{\lambda}_{ci} = \frac{N_{ci+\alpha}}{N_c + d\alpha} \qquad (9)$$

Với α là một số dương, thường bằng 1, để tránh trường hợp tử số bằng 0. Mẫu số được cộng với dα để đảm bảo tổng xác suất $\sum_{i=1}^{d} \hat{\lambda}_{ci} = 1$.

Như vậy, mỗi class c sẽ được mô tả bởi bộ các số dương có tổng bằng 1: $\hat{\lambda}_{ci} = \{\hat{\lambda}_{c1}, \dots, \hat{\lambda}_{cd}\}$

b) Mô hình Logistic

Hai mô hình tuyến tính (linear models) Linear Regression và Perceptron Learning Algorithm (PLA) chúng ta đã biết đều có chung một dạng:

$$y = f(w^T x)$$

trong đó f() được gọi là activation function, và x được hiểu là dữ liệu mở rộng với x_0=1 được thêm vào để thuận tiện cho việc tính toán. Với linear regression thì f(s) = s, với PLA thì f(s) = sgn(s). Trong linear regression, tích vô hướng $w^T x$ được trực tiếp sử dụng để dự đoán output y, loại này phù hợp nếu chúng ta cần dự đoán một giá trị thực của đầu ra không bị chặn trên và dưới. Trong PLA, đầu ra chỉ nhận một trong hai giá trị 1hoặc −1, phù hợp với các bài toán binary classification.

Trong bài này, tôi sẽ giới thiệu mô hình thứ ba với một activation khác, được sử dụng cho các bài toán flexible hơn. Trong dạng này, đầu ra có thể được thể hiện dưới dạng xác suất (probability). Ví dụ: xác suất thi đỗ nếu biết thời gian ôn thi, xác suất ngày mai có mưa dựa trên những thông tin đo được trong ngày hôm nay,… Mô hình mới này của chúng ta có tên là logistic regression. Mô hình này giống với linear regression ở khía cạnh đầu ra là số thực, và giống với PLA ở việc đầu ra bị chặn (trong đoạn [0,1]). Mặc dù trong tên có chứa từ regression, logistic regression thường được sử dụng nhiều hơn cho các bài toán classification.

❖ Mô hình Logistic Regression

Đầu ra dự đoán của:

- Linear Regression:

$$f(x) = (w^T x)$$

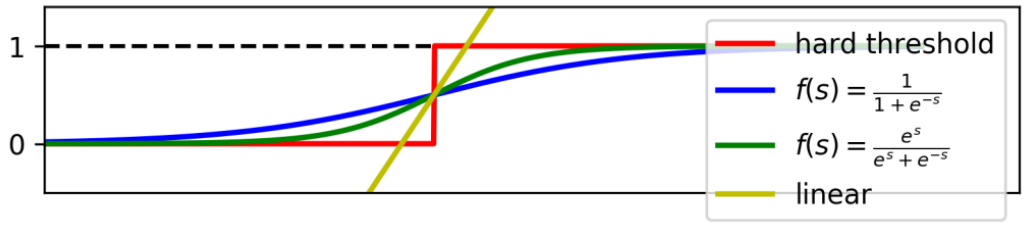
- PLA:

$$f(x) = sgn(w^T x)$$

Đầu ra dự đoán của logistic regression thường được viết chung dưới dạng:

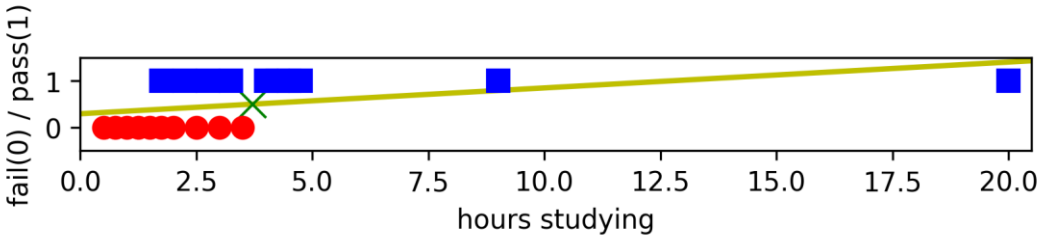
$$f(x) = \theta(w^T x)$$

Trong đó θ được gọi là logistic function. Một số activation cho mô hình tuyến tính được cho trong hình dưới đây:



Hình 2: Các activation function khác nhau.

- Đường màu vàng biểu diễn linear regression. Đường này không bị chặn nên không phù hợp cho bài toán này. Có một trick nhỏ để đưa nó về dạng bị chặn: cắt phần nhỏ hơn 0 bằng cách cho chúng bằng 0, cắt các phần lớn hơn 1 bằng cách cho chúng bằng 1. Sau đó lấy điểm trên đường thẳng này có tung độ bằng 0.5 làm điểm phân chia hai class, đây cũng không phải là một lựa chọn tốt.



Hình 3: Tại sao Linear Regression không phù hợp?

- Đường màu đỏ (chỉ khác với activation function của PLA ở chỗ hai class là 0 và 1 thay vì -1 và 1) cũng thuộc dạng ngưỡng cứng (hard threshold). PLA không hoạt động trong bài toán này vì dữ liệu đã cho không linearly separable.
- Các đường màu xanh lam và xanh lục phù hợp với bài toán của chúng ta hơn. Chúng có một vài tính chất quan trọng sau:
  - Là hàm số liên tục nhận giá trị thực, bị chặn trong khoảng (0,1)(0,1).
  - Nếu coi điểm có tung độ là 1/2 làm điểm phân chia thì các điểm càng xa điểm này về phía bên trái có giá trị càng gần 0. Ngược lại, các điểm càng xa điểm này về phía phải có giá trị càng gần 1. Điều này khớp với nhận xét rằng học càng nhiều thì xác suất đỗ càng cao và ngược lại.
  - Mượt (smooth) nên có đạo hàm mọi nơi, có thể được lợi trong việc tối ưu.

❖ Sigmoid function

Trong số các hàm số có 3 tính chất nói trên thì hàm sigmoid:

$$f(s) = \frac{1}{1+e^{-s}} \triangleq \sigma(s)$$

được sử dụng nhiều nhất, vì nó bị chặn trong khoảng (0,1)(0,1). Thêm nữa:

$$\lim_{s\to-\infty}\sigma(s)=0;\ \lim_{s\to+\infty}\sigma(s)=1$$

Đặc biệt hơn nữa:

$$\begin{aligned}\sigma'(s)&=\frac{e^{-s}}{(1+e^{-s})^2}\\&=\frac{1}{1+e^{-s}}\frac{e^{-s}}{1+e^{-s}}\\&=\sigma(s)(1-\sigma(s))\end{aligned}$$

Công thức đạo hàm đơn giản thế này giúp hàm số này được sử dụng rộng rãi. Ở phần sau, tôi sẽ lý giải việc người ta đã tìm ra hàm số đặc biệt này như thế nào.

Ngoài ra, hàm tanh cũng hay được sử dụng:

$$\tanh(s)=\frac{e^{s}-e^{-s}}{e^{s}e+^{-s}}$$

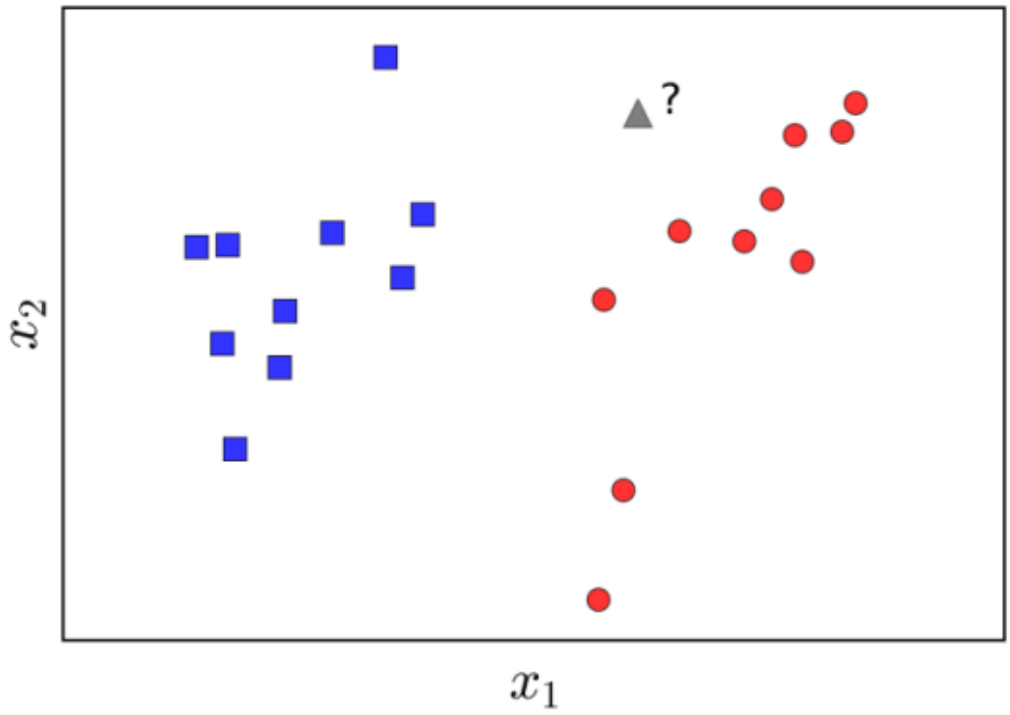
Hàm số này nhận giá trị trong khoảng (−1,1)(−1,1) nhưng có thể dễ dàng đưa nó về khoảng (0,1)(0,1). Bạn đọc có thể chứng minh được:

$$\tanh(s)=2\sigma(2s)-1$$

c) Mô hình Perceptron

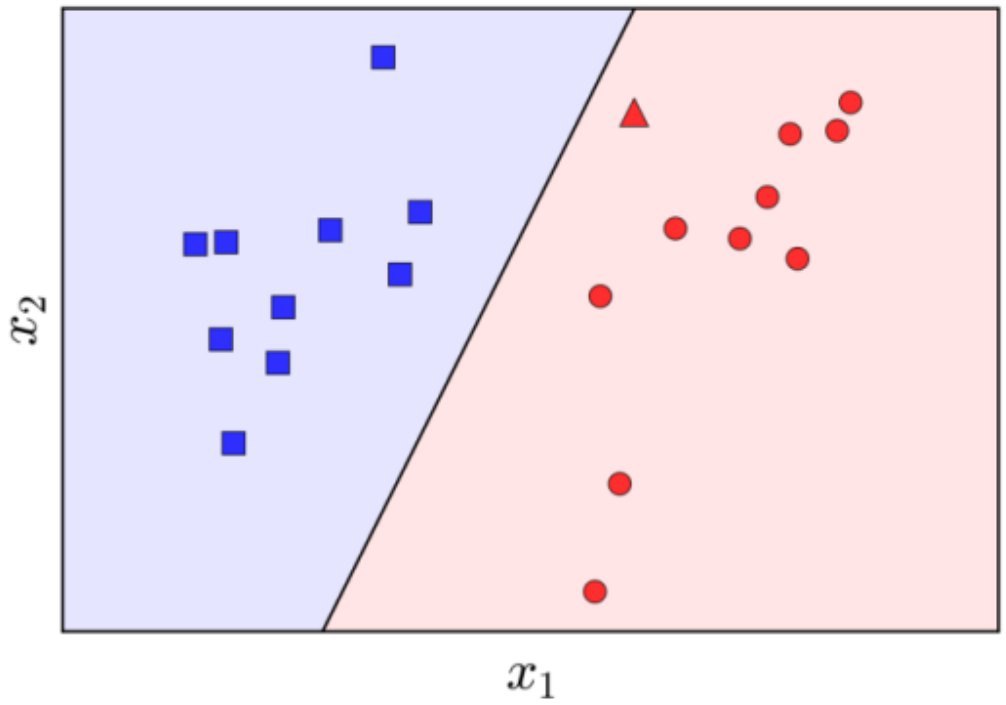
Perceptron là một thuật toán Classification cho trường hợp đơn giản nhất: chỉ có hai class (lớp) (bài toán với chỉ hai class được gọi là binary classification) và cũng chỉ hoạt động được trong một trường hợp rất cụ thể. Tuy nhiên, nó là nền tảng cho một mảng lớn quan trọng của Machine Learning là Neural Networks và sau này là Deep Learning.

Giả sử chúng ta có hai tập hợp dữ liệu đã được gán nhãn được minh hoạ trong Hình 1 dưới đây. Hai class của chúng ta là tập các điểm màu xanh và tập các điểm màu đỏ. Bài toán đặt ra là: từ dữ liệu của hai tập được gán nhãn cho trước, hãy xây dựng một classifier (bộ phân lớp) để khi có một điểm dữ liệu hình tam giác màu xám mới, ta có thể dự đoán được màu (nhãn) của nó.



Hình 1: Bài toán Perceptron

Hiểu theo một cách khác, chúng ta cần tìm lãnh thổ của mỗi class sao cho, với mỗi một điểm mới, ta chỉ cần xác định xem nó nằm vào lãnh thổ của class nào rồi quyết định nó thuộc class đó. Để tìm lãnh thổ của mỗi class, chúng ta cần đi tìm biên giới (boundary) giữa hai lãnh thổ này. Vậy bài toán classification có thể coi là bài toán đi tìm boundary giữa các class. Và boundary đơn giản nhất trong không gian hai chiều là một đường thẳng, trong không gian ba chiều là một mặt phẳng, trong không gian nhiều chiều là một siêu mặt phẳng (hyperplane) (gọi chung những boundary này là đường phẳng). Những boundary phẳng này được coi là đơn giản vì nó có thể biểu diễn dưới dạng toán học bằng một hàm số đơn giản có dạng tuyến tính, tức linear. Tất nhiên, chúng ta đang giả sử rằng tồn tại một đường phẳng để có thể phân định lãnh thổ của hai class. Hình 2 minh họa một đường thẳng phân chia hai class trong mặt phẳng. Phần có nền màu xanh được coi là lãnh thổ của lớp xanh, phần có nên màu đỏ được coi là lãnh thổ của lớp đỏ. Trong trường hợp này, điểm dữ liệu mới hình tam giác được phân vào class đỏ.



Hình 2: Bài toán Perceptron

❖ Bài toán Perceptron

Bài toán Perceptron được phát biểu như sau: Cho hai class được gán nhãn, hãy tìm một đường phẳng sao cho toàn bộ các điểm thuộc class 1 nằm về 1 phía, toàn bộ các điểm thuộc class 2 nằm về phía còn lại của đường phẳng đó. Với giả định rằng tồn tại một đường phẳng như thế.

Nếu tồn tại một đường phẳng phân chia hai class thì ta gọi hai class đó là linearly separable. Các thuật toán classification tạo ra các boundary là các đường phẳng được gọi chung là Linear Classifier.

❖ Thuật toán Perceptron (PLA)

Cũng giống như các thuật toán lặp trong K-means Clustering và Gradient Descent, ý tưởng cơ bản của PLA là xuất phát từ một nghiệm dự đoán nào đó, qua mỗi vòng lặp, nghiệm sẽ được cập nhật tới một ví trí tốt hơn. Việc cập nhật này dựa trên việc giảm giá trị của một hàm mất mát nào đó.

Giả sử $X=[x_1,x_2,\ldots,x_N] \in \mathbb{R}^{dxN}$ là ma trận chứa các điểm dữ liệu mà mỗi cột $x_i \in \mathbb{R}^{dxN}$ là một điểm dữ liệu trong không gian d chiều. (Chú ý: khác với các bài trước tôi thường dùng các vector hàng để mô tả dữ liệu, trong bài này tôi dùng vector cột để biểu diễn. Việc biểu diễn dữ liệu ở dạng hàng hay cột tùy thuộc vào từng bài toán, miễn sao cách biểu diễn toán học của nó khiến cho người đọc thấy dễ hiểu).

Giả sử thêm các nhãn tương ứng với từng điểm dữ liệu được lưu trong một vector hàng $y=[y_1,y_2,\ldots, y_N] \in \mathbb{R}^{1xN}$, với $y_i=1$ nếu x_i thuộc class 1 (xanh) và

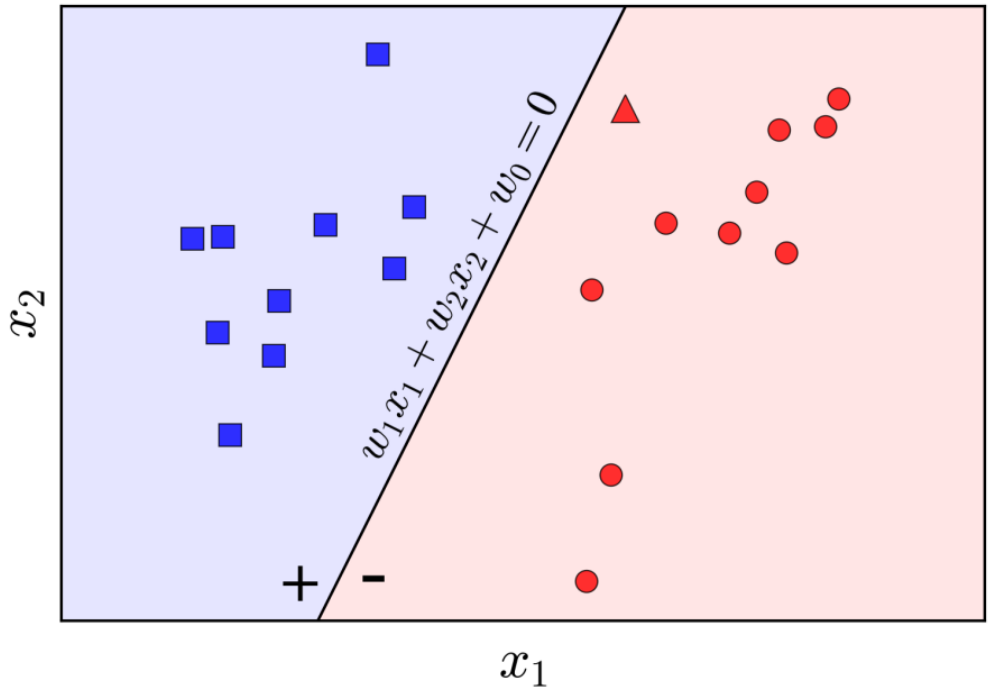
y_i=−1 nếu x_i thuộc class 2 (đỏ).

Tại một thời điểm, giả sử ta tìm được boundary là đường phẳng có phương trình:

$$f_w(x) = \quad w_1x_1 + w_2x_2 + \cdots + \quad w_dx_d = w^T\bar{x} = 0$$

với $\bar{x}$ là điểm dữ liệu mở rộng bằng cách thêm phần tử x_0=1 lên trước vector x tương tự như trong Linear Regression. Và từ đây, khi nói x, tôi cũng ngầm hiểu là điểm dữ liệu mở rộng.

Để cho đơn giản, chúng ta hãy cùng làm việc với trường hợp mỗi điểm dữ liệu có số chiều d=2. Giả sử đường thẳng $w_1x_1 + w_2x_2 + w_0 = 0$ chính là nghiệm cần tìm như Hình 3 dưới đây:



Hình 3: Phương trình đường thẳng boundary.

Nhận xét rằng các điểm nằm về cùng 1 phía so với đường thẳng này sẽ làm cho hàm số $f_w(x)$ mang cùng dấu. Chỉ cần đổi dấu của w nếu cần thiết, ta có thể giả sử các điểm nằm trong nửa mặt phẳng nền xanh mang dấu dương (+), các điểm nằm trong nửa mặt phẳng nền đỏ mang dấu âm (-). Các dấu này cũng tương đương với nhãn y của mỗi class. Vậy nếu w là một nghiệm của bài toán Perceptron, với một điểm dữ liệu mới x chưa được gán nhãn, ta có thể xác định class của nó bằng phép toán đơn giản như sau:

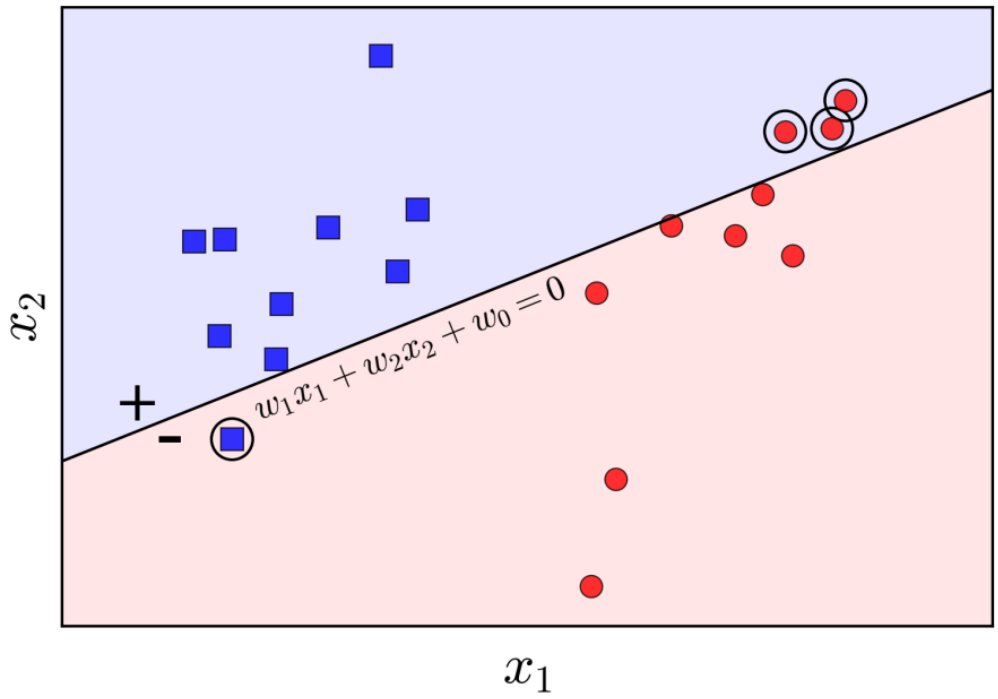
$$label(x) = 1\ if\ \ w^Tx\ \geq 0, otherwise - 1$$

Ngắn gọn hơn: $label(x) = sgn(w^Tx)$

trong đó, sgnsgn là hàm xác định dấu, với giả sử rằng sgn(0)=1.

❖ Xây dựng hàm mất mát

Tiếp theo, chúng ta cần xây dựng hàm mất mát với tham số ww bất kỳ. Vẫn trong không gian hai chiều, giả sử đường thẳng $w_1x_1 + w_2x_2 + w_0 = 0$ được cho như Hình 4 dưới đây:



Hình 4: Đường thẳng bất kỳ và các điểm bị misclassified được khoanh tròn.

Trong trường hợp này, các điểm được khoanh tròn là các điểm bị misclassified (phân lớp lỗi). Điều chúng ta mong muốn là không có điểm nào bị misclassified. Hàm mất mát đơn giản nhất chúng ta nghĩ đến là hàm đếm số lượng các điểm bị misclassied và tìm cách tối thiểu hàm số này:

$$J_1(w) = \sum_{x_i \in \mathrm{M}} (-y_i sgn(w^T x))$$

trong đó M là tập hợp các điểm bị misclassifed (tập hợp này thay đổi theo ww). Với mỗi điểm $x_i \in M$, vì điểm này bị misclassified nên y_i và $sgn(w^T x)$ khác nhau, và vì thế $-y_i sgn(w^T x) = 1$. Vậy $J_1(w)$ chính là hàm đếm số lượng các điểm bị misclassified. Khi hàm số này đạt giá trị nhỏ nhất bằng 0 thì ta không còn điểm nào bị misclassified.

Một điểm quan trọng, hàm số này là rời rạc, không tính được đạo hàm theo w nên rất khó tối ưu. Chúng ta cần tìm một hàm mất mát khác để việc tối ưu khả thi hơn.

Xét hàm mất mát sau đây:

$$J(w) = \sum_{x_i \in M} (-y_i w^T x)$$

Hàm J() khác một chút với hàm $J_1()$ở việc bỏ đi hàm sgn. Nhận xét rằng khi một điểm misclassified x_i nằm càng xa boundary thì giá trị $-y_i w^T x$ sẽ càng lớn, nghĩa là sự sai lệch càng lớn. Giá trị nhỏ nhất của hàm mất mát này cũng bằng 0 nếu không có điểm nào bị misclassifed. Hàm mất mát này cũng được cho là tốt hơn hàm $J_1()$vì nó trừng phạt rất nặng

những điểm lấn sâu sang lãnh thổ của class kia. Trong khi đó, $J_1()$trừng phạt các điểm misclassified như nhau (đều = 1), bất kể chúng xa hay gần với đường biên giới.

Tại một thời điểm, nếu chúng ta chỉ quan tâm tới các điểm bị misclassified thì hàm số J(w) khả vi (tính được đạo hàm), vậy chúng ta có thể sử dụng Gradient Descent hoặc Stochastic Gradient Descent (SGD) để tối ưu hàm mất mát này. Với ưu điểm của SGD cho các bài toán large-scale, chúng ta sẽ làm theo thuật toán này.

Với một điểm dữ liệu x_i bị misclassified, hàm mất mát trở thành:

$$J(w; x_i, y_i) = -y_i w^T x_i$$

Đạo hàm tương ứng:

$$\nabla_w J(w; x_i, y_i) = -y_i x_i$$

Vậy quy tắc cập nhật là:

$$w = w + \eta y_i x_i$$

với η là learning rate được chọn bằng 1. Ta có một quy tắc cập nhật rất gọn là: $w_{t+1} = w_t + y_i x_i$. Nói cách khác, với mỗi điểm x_i bị misclassifed, ta chỉ cần nhân điểm đó với nhãn y_i của nó, lấy kết quả cộng vào w ta sẽ được w mới.

Ta có một quan sát nhỏ ở đây:

$$w_{t+1}^T x_i = (w_t + y_i x_i)^T x_i$$
$$= w_t^T x_i + y_i ||x_i||_2^2$$

Nếu y_i=1, vì x_i bị misclassifed nên $w_t^T x_i < 0$. Cũng vì y_i=1 nên $y_i ||x_i||_2^2 = ||x_i||_2^2 > 0$ (chú ý x_0 =1), nghĩa là $w_{t+1}^T x_i > w_t^T x_i$. Lý giải bằng lời, w_{t+1} tiến về phía làm cho x_i được phân lớp đúng. Điều tương tự xảy ra nếu y_i=−1.

Tóm lại, thuật toán Perceptron có thể được viết như sau:

❖ Tóm tắt PLA

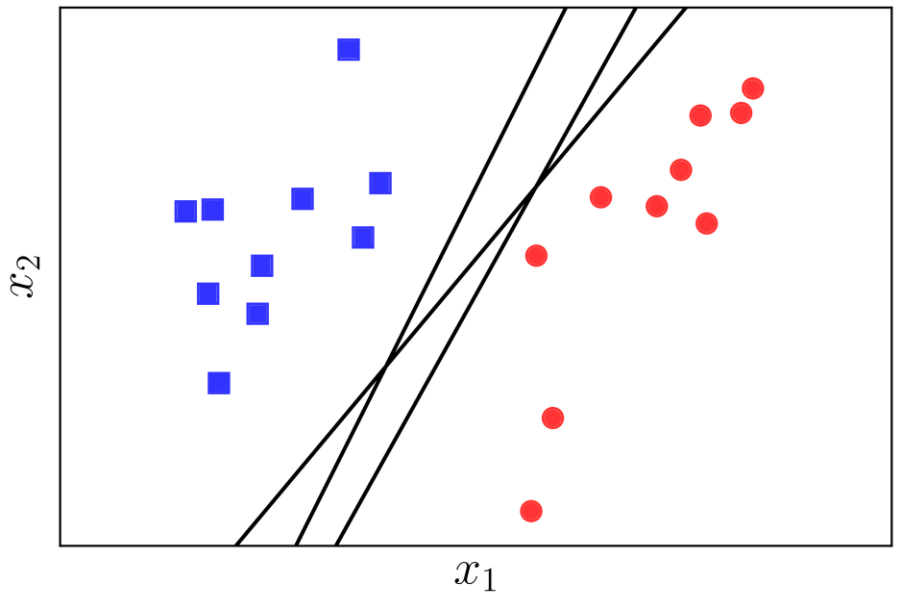
1. Chọn ngẫu nhiên một vector hệ số w với các phần tử gần 0.
2. Duyệt ngẫu nhiên qua từng điểm dữ liệu x_i:
   - Nếu x_i được phân lớp đúng, tức $\text{sgn}(w^T x_i) = y_i$, chúng ta không cần làm gì.
   - Nếu x_i bị misclassifed, cập nhật w theo công thức:

$$w = w + x_i y_i$$

3. Kiểm tra xem có bao nhiêu điểm bị misclassifed. Nếu không còn điểm nào, dừng thuật toán. Nếu còn, quay lại bước 2.

d) Mô hình SVM

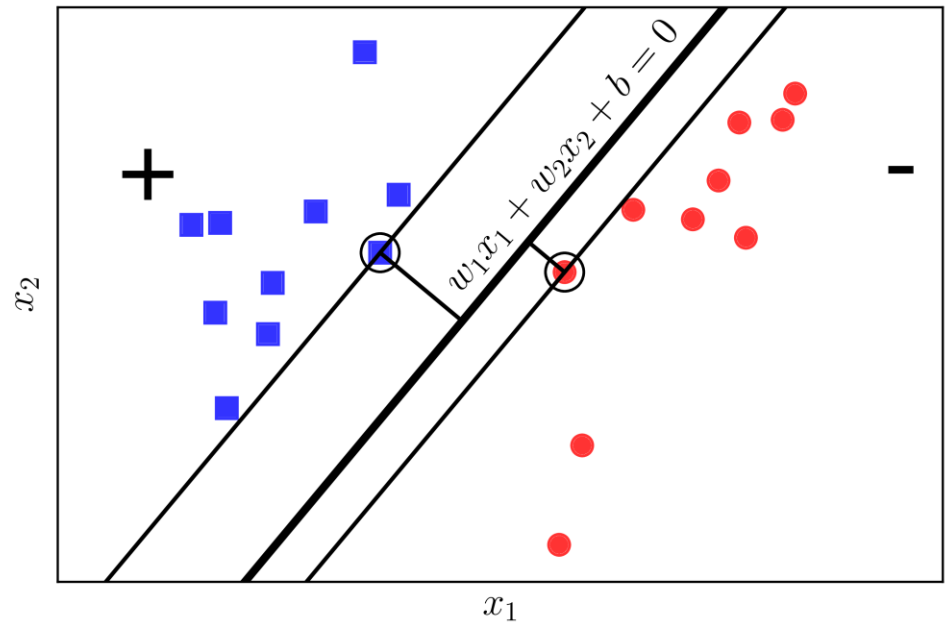
Quay lại với bài toán trong Perceptron Learning Algorithm (PLA). Giả sử rằng có hai class khác nhau được mô tả bởi các điểm trong không gian nhiều chiều, hai classes này linearly separable, tức tồn tại một siêu phẳng phân chia chính xác hai classes đó. Hãy tìm một siêu mặt phẳng phân chia hai classes đó, tức tất cả các điểm thuộc một class nằm về cùng một phía của siêu mặt phẳng đó và ngược phía với toàn bộ các điểm thuộc class còn lại. Chúng ta đã biết rằng, thuật toán PLA có thể làm được việc này nhưng nó có thể cho chúng ta vô số nghiệm như Hình 1 dưới đây:

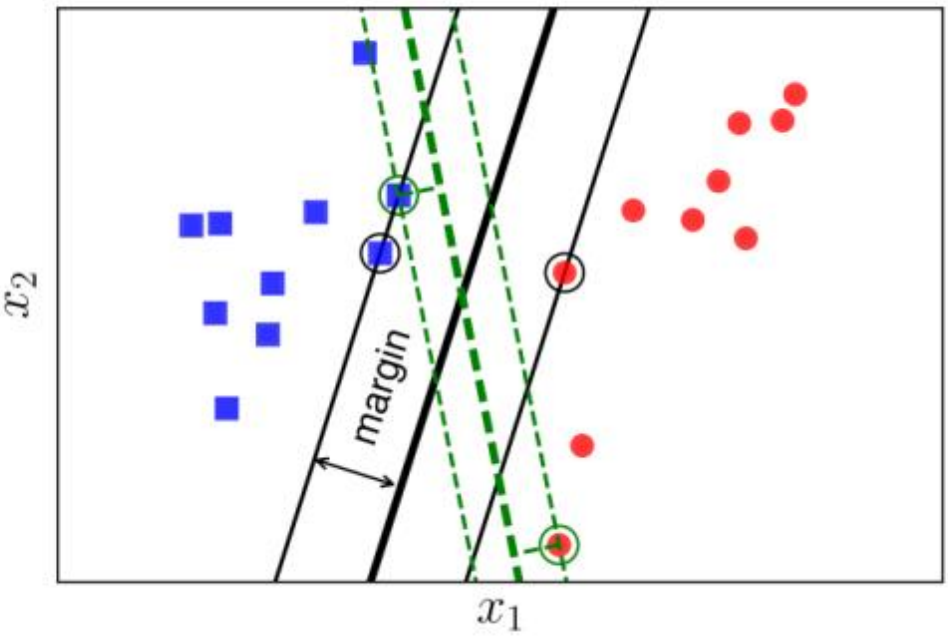


Hình 1: Các mặt phân cách hai classes linearly separable.

Câu hỏi đặt ra là: trong vô số các mặt phân chia đó, đâu là mặt phân chia tốt nhất theo một tiêu chuẩn nào đó? Trong ba đường thẳng minh họa trong Hình 1, có hai đường thẳng khá lệch về phía class hình tròn đỏ. Điều này có thể khiến cho lớp màu đỏ không vui vì lãnh thổ xem ra bị lấn nhiều quá. Liệu có cách nào để tìm được đường phân chia mà cả hai classes đều cảm thấy công bằng và hạnh phúc nhất hay không?

Chúng ta cần tìm một tiêu chuẩn để đo sự hạnh phúc của mỗi class. Hãy xem Hình 2 dưới đây:





Hình 2: Margin của hai classes là bằng nhau và lớn nhất có thể.

Nếu ta định nghĩa mức độ hạnh phúc của một class tỉ lệ thuận với khoảng cách gần nhất từ một điểm của class đó tới đường/mặt phân chia, thì ở Hình 2, class tròn đỏ sẽ không được hạnh phúc cho lắm vì đường phân chia gần nó hơn class vuông xanh rất nhiều. Chúng ta cần một đường phân chia sao cho khoảng cách từ điểm gần nhất của mỗi class (các điểm được khoanh tròn) tới đường phân chia là như nhau, như thế thì mới công bằng. Khoảng cách như nhau này được gọi là margin (lề).

Đã có công bằng rồi, chúng ta cần văn minh nữa. Công bằng mà cả hai đều kém hạnh phúc như nhau thì chưa phải là văn minh cho lắm.

Chúng ta xét tiếp Hình 2 khi khoảng cách từ đường phân chia tới các điểm gần nhất của mỗi class là như nhau. Xét hai cách phân chia bởi đường nét liền màu đen và đường nét đứt màu lục, đường nào sẽ làm cho cả hai class hạnh phúc hơn? Rõ ràng đó phải là đường nét liền màu đen vì nó tạo ra một margin rộng hơn.

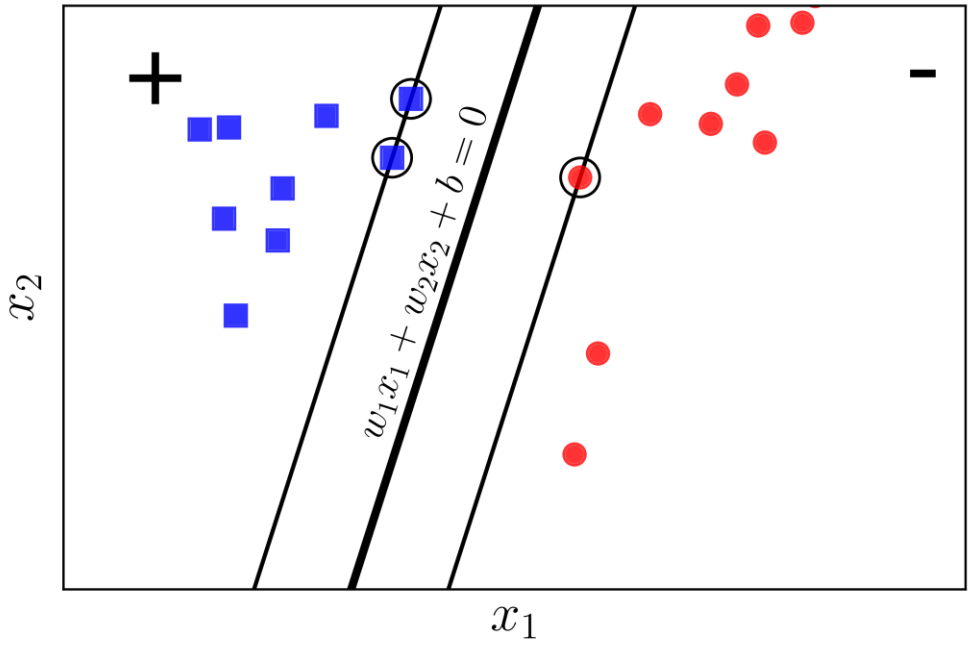
Việc margin rộng hơn sẽ mang lại hiệu ứng phân lớp tốt hơn vì sự phân chia giữa hai classes là rạch ròi hơn. Việc này, sau này các bạn sẽ thấy, là một điểm khá quan trọng giúp Support Vector Machine mang lại kết quả phân loại tốt hơn so với Neural Network với 1 layer, tức Perceptron Learning Algorithm.

Bài toán tối ưu trong Support Vector Machine (SVM) chính là bài toán đi tìm đường phân chia sao cho margin là lớn nhất. Đây cũng là lý do vì sao SVM còn được gọi là Maximum Margin Classifier. Nguồn gốc của tên gọi Support Vector Machine sẽ sớm được làm sáng tỏ.

❖ Xây dựng bài toán tối ưu cho SVM

Giả sử rằng các cặp dữ liệu của training set là (x_1, y_1), (x_2, y_2),…, (x_N, y_N) với vector $x_i \in \mathbb{R}^d$ thể hiện đầu vào của một điểm dữ liệu và y_i là nhãn của điểm dữ liệu đó. d là số chiều của dữ liệu và N là số điểm dữ liệu. Giả sử rằng nhãn của mỗi điểm dữ liệu được xác định bởi $y_i=1$ (class 1) hoặc $y_i = -1$ (class 2) giống như trong PLA.

Để giúp các bạn dễ hình dung, chúng ta cùng xét trường hợp trong không gian hai chiều dưới đây. Không gian hai chiều để các bạn dễ hình dung, các phép toán hoàn toàn có thể được tổng quát lên không gian nhiều chiều.



Hình 3: Phân tích bài toán SVM.

Giả sử rằng các điểm vuông xanh thuộc class 1, các điểm tròn đỏ thuộc class -1 và mặt $w^Tx + b = w_1x_1 + w_2x_2 + b = 0$ là mặt phân chia giữa hai classes (Hình 3). Hơn nữa, class 1 nằm về phía dương, class -1 nằm về phía âm của mặt phân chia. Nếu ngược lại, ta chỉ cần đổi dấu của w và b. Chú ý rằng chúng ta cần đi tìm các hệ số w và b.

Ta quan sát thấy một điểm quan trọng sau đây: với cặp dữ liệu (x_n, y_n) bất kỳ, khoảng cách từ điểm đó tới mặt phân chia là:

$$\frac{y_n(w^Tx_n + b)}{||w||_2}$$

Điều này có thể dễ nhận thấy vì theo giả sử ở trên, y_n luôn cùng dấu với phía của x_n. Từ đó suy ra y_n cùng dấu với $(w^T x_n + b)$, và tử số luôn là 1 số không âm.

Với mặt phần chia như trên, margin được tính là khoảng cách gần nhất từ 1 điểm tới mặt đó (bất kể điểm nào trong hai classes):

$$\text{margin} = \min_n \frac{y_n(w^Tx_n + b)}{||w||_2}$$

Bài toán tối ưu trong SVM chính là bài toán tìm w và b sao cho margin này đạt giá trị lớn nhất

$$(w, b) = \arg\max_{w,b} \left\{ \min_n \frac{y_n(w^Tx_n + b)}{||w||_2} \right\}$$

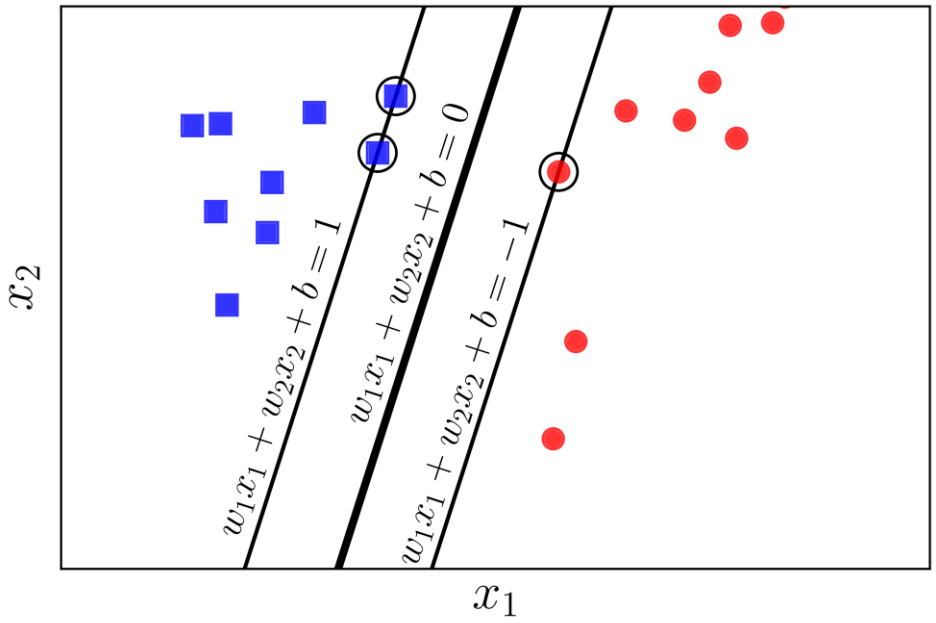
$$= \arg \max_{w,b} \left\{ \frac{1}{||w||_2} \min_n y_n(w^T x_n + b) \right\}$$

Việc giải trực tiếp bài toán này sẽ rất phức tạp, nhưng các bạn sẽ thấy có cách để đưa nó về bài toán đơn giản hơn.

Nhận xét quan trọng nhất là nếu ta thay vector hệ số w bởi kw và b bởi kb trong đó k là một hằng số dương thì mặt phân chia không thay đổi, tức khoảng cách từ từng điểm đến mặt phân chia không đổi, tức margin không đổi. Dựa trên tính chất này, ta có thể giả sử:

$$y_n(w^T x_n + b) = 1$$

với những điểm nằm gần mặt phân chia nhất như Hình 4 dưới đây:



Hình 4: Các điểm gần mặt phân cách nhất của hai classes được khoanh tròn.

Như vậy, với mọi n, ta có:

$$y_n(w^T x_n + b) \geq 1$$

Vậy bài toán tối ưu (1)có thể đưa về bài toán tối ưu có ràng buộc sau đây:

$$(w, b) = \arg \max_{w,b} \frac{1}{||w||_2}$$

subject to:

$$y_n(w^T x_n + b) \geq 1 \; \forall n = 1,2, \dots, N \qquad (2)$$

Bằng 1 biến đổi đơn giản, ta có thể đưa bài toán này về bài toán dưới đây:

$$(w, b) = \arg\min_{w,b} \frac{1}{2}||w||_2^2$$

subject to:

$$1 - y_n(w^T x_n + b) \leq 0 \; \forall n = 1,2, \dots, N \quad (3)$$

Ở đây, chúng ta đã lấy nghịch đảo hàm mục tiêu, bình phương nó để được một hàm khả vi, và nhân với $\frac{1}{2}$ để biểu thức đạo hàm đẹp hơn.

Quan sát quan trọng: Trong bài toán (3), hàm mục tiêu là một norm, nên là một hàm lồi. Các hàm bất đẳng thức ràng buộc là các hàm tuyến tính theo w và b, nên chúng cũng là các hàm lồi. Vậy bài toán tối ưu (3) có hàm mục tiêu là lồi, và các hàm ràng buộc cũng là lồi, nên nó là một bài toán lồi. Hơn nữa, nó là một Quadratic Programming. Thậm chí, hàm mục tiêu là strictly convex vì $||w||_2^2 = w^T I w$ và I là ma trận đơn vị - là một ma trận xác định dương. Từ đây có thể suy ra nghiệm cho SVM là duy nhất.

Đến đây thì bài toán này có thể giải được bằng các công cụ hỗ trợ tìm nghiệm cho Quadratic Programing, ví dụ CVXOPT.

Tuy nhiên, việc giải bài toán này trở nên phức tạp khi số chiều d của không gian dữ liệu và số điểm dữ liệu N tăng lên cao.

Người ta thường giải bài toán đối ngẫu của bài toán này. Thứ nhất, bài toán đối ngẫu có những tính chất thú vị hơn khiến nó được giải hiệu quả hơn. Thứ hai, trong quá trình xây dựng bài toán đối ngẫu, người ta thấy rằng SVM có thể được áp dụng cho những bài toán mà dữ liệu không linearly separable, tức các đường phân chia không phải là một mặt phẳng mà có thể là các mặt có hình thù phức tạp hơn.

Xác định class cho một điểm dữ liệu mới: Sau khi tìm được mặt phân cách $w^T x + b = 0$, class của bất kỳ một điểm nào sẽ được xác định đơn giản bằng cách:

$$\text{class}(x) = \text{sgn}(w^T x + b)$$

Trong đó hàm sgn là hàm xác định dấu, nhận giá trị 1 nếu đối số là không âm và -1 nếu ngược lại.

2. Một số thư viện hỗ trợ

a) Numpy

NumPy viết tắt của Numerical Python là thư viện cốt lõi cho scientific computing, nó chứa một đối tượng mảng n chiều mạnh mẽ, cung cấp các công cụ để tích hợp C, C ++, v.v. Nó cũng hữu ích trong đại số tuyến tính, random number capability, ... Các loại tính toán số này được sử dụng rộng rãi trong các nhiệm vụ như:

- ✓ Mô hình học máy: trong khi viết các thuật toán Machine Learning, người ta phải thực hiện các phép tính số khác nhau trên ma trận. Ví dụ, nhân ma trận, hoán vị, cộng, v.v. NumPy cung cấp một thư viện tuyệt vời để dễ dàng (về mã viết) và tính

toán nhanh (về tốc độ). Mảng NumPy được sử dụng để lưu trữ cả dữ liệu đào tạo cũng như các tham số của các mô hình Machine Learning.

- ✓ Xử lý hình ảnh và đồ họa máy tính: Hình ảnh trong máy tính được thể hiện dưới dạng các mảng số đa chiều, NumPy trở thành sự lựa chọn tự nhiên nhất. Trên thực tế, nó cung cấp một số chức năng thư viện tuyệt vời để thao tác nhanh chóng với hình ảnh. Một số ví dụ đang phản chiếu một hình ảnh, xoay một hình ảnh theo một góc nhất định, v.v.
- ✓ Các nhiệm vụ toán học: NumPy khá hữu ích để thực hiện các nhiệm vụ toán học khác nhau như tích hợp số, phân biệt, nội suy, ngoại suy và nhiều nhiệm vụ khác. Như vậy, nó tạo thành một sự thay thế nhanh chóng dựa trên MATLAB dựa trên Python khi nói đến các nhiệm vụ toán học.

b) Pandas

Pandas là thư viện mã nguồn mở với hiệu năng cao cho phân tích dữ liệu trong Python được phát triển bởi Wes McKinney trong năm 2008. Là bộ công cụ phân tích và xử lý dữ liệu mạnh mẽ. Thư viện này được sử dụng rộng rãi trong cả nghiên cứu lẫn phát triển các ứng dụng về khoa học dữ liệu. Pandas sử dụng một cấu trúc dữ liệu riêng là Dataframe và cung cấp rất nhiều chức năng xử lý và làm việc trên cấu trúc dữ liệu này. Một số tính năng nổi bật của pandas như:

- ✓ Có thể xử lý tập dữ liệu khác nhau về định dạng: chuỗi thời gian, bảng không đồng nhất, ma trận dữ liệu
- ✓ Khả năng import dữ liệu từ nhiều nguồn khác nhau như : tệp CSV và văn bản, Microsoft Excel, cơ sở dữ liệu SQL và định dạng HDF5 nhanh;
- ✓ DataFrame đem lại sự linh hoạt và hiệu quả trong thao tác dữ liệu và lập chỉ mục;
- ✓ Liên kết dữ liệu thông minh, xử lý được trường hợp dữ liệu bị thiếu. Tự động đưa dữ liệu lộn xộn về dạng có cấu trúc;
- ✓ Lập chỉ mục theo các chiều của dữ liệu giúp thao tác giữa dữ liệu cao chiều và dữ liệu thấp chiều;
- ✓ Tối ưu hóa cao cho hiệu suất , với các đường dẫn quan trọng được viết bằng Cython hoặc C.

c) Gensim

Gensim là một thư viện xử lý ngôn ngữ tự nhiên. Gensim đưa ra các công cụ làm việc với vector space modeling và topic modeling. Thư viện này được thiết kế phù hợp với các text lớn, chỉ có trong xử lý in-memory. Sử dụng NumPy data structures và SciPy operations sẽ mang đến hiệu quả cực tốt. Cả hai đều hiệu quả và dễ sử dụng. Gensim được thiết kế để dùng với các digital text thô và chưa theo cấu trúc. Gensim đưa ra các thuật toán như Dirichlet processes theo thứ bậc (HDP), phân tích ngữ nghĩa tiềm ẩn - latent semantic analysis (LSA) và phân bổ Dirichlet tiềm ẩn (LDA), cũng như tf-idf, projections ngẫu nhiên, word2vec và document2vec hỗ trợ việc

kiểm tra các text để tái hiện lại các patterns words trong set documents (thường được đề cấp đến như 1 corpus). Tất cả các thuật toán đều là unsupervised- input duy nhất là corpus.

Các tính năng chính của Gensim:

- ✓ Gensim độc lập với bộ nhớ, nó có thể dễ dàng xử lý khối lượng lớn và quy mô web bằng cách sử dụng các thuật toán đào tạo trực tuyến gia tăng của nó. Nó có khả năng mở rộng về bản chất, vì không cần toàn bộ kho dữ liệu đầu vào cư trú hoàn toàn trong Bộ nhớ truy cập ngẫu nhiên (RAM) bất cứ lúc nào
- ✓ Gensim cung cấp các triển khai đa lõi hiệu quả của các thuật toán phổ biến khác nhau như Phân tích ngữ nghĩa tiềm ẩn (LSA), Phân bổ Dirichlet tiềm ẩn (LDA), Dự đoán ngẫu nhiên (RP), Quy trình Dirichlet phân cấp (HDP) và học sâu word2vec,
- ✓ Gensim có bản chất mạnh mẽ: có thể dễ dàng cắm vào kho dữ liệu đầu vào hoặc luồng dữ liệu của riêng mình, cũng rất dễ dàng để mở rộng với các thuật toán không gian Vector khác.

d) Beautisoup

Beautiful Soup là gói Python để phân tích cú pháp các tài liệu HTML và XML (bao gồm cả việc đánh dấu không đúng định dạng, tức là các thẻ không đóng). Nó tạo ra một cây phân tích cho các trang được phân tích cú pháp có thể được sử dụng để trích xuất dữ liệu từ HTML , rất hữu ích cho việc quét web .

e) Flask

Flask là một Web Framework rất nhẹ của Python được tạo ra bởi Armin Ronacher của Pocoo, một nhóm những người đam mê Python quốc tế được thành lập vào năm 2004. Nó được phân loại là một microframework vì nó không yêu cầu các công cụ hoặc thư viện cụ thể. Nó không có lớp trừu tượng hóa cơ sở dữ liệu, xác thực mẫu hoặc bất kỳ thành phần nào khác nơi các thư viện bên thứ ba tồn tại trước đó cung cấp các chức năng chung. Tuy nhiên, Flask hỗ trợ các tiện ích mở rộng có thể thêm các tính năng ứng dụng như thể chúng được triển khai trong chính Flask

Điểm mạnh của flask đó chính là sự nhỏ gọn, dễ cài đặt và triển khai**.** Flask tập trung vào sự tối giản, cho phép chúng ta xây dựng ứng dụng nhanh hơn

Flask có kiến trúc nhỏ, gọn nên ta hoàn toàn không bị bó buộc bởi bộ khung cồng kềnh, không gặp bất cứ khó khăn nào khi cấu hình hay tổ chức ứng dụng. Không những thế, Flask còn có các ưu điểm nổi bật như: cực kỳ linh hoạt, tối giản, dễ tìm hiểu và sử dụng, định tuyến dễ dàng, rất dễ mở rộng. . Flask cung cấp rất nhiều tài liệu từ cài đặt đến thực hiện và triển khai, từ hướng dẫn nhanh đến hướng dẫn chi tiết. Với Flask, việc chọn component nào cho ứng dụng là việc của chúng ta. Điều này thật tuyệt, vì mỗi web application có những đặc điểm và tính năng riêng, nó không phần phải chứa các component mà nó không dùng.

f) Scikit-learn

Dự án scikit-learn (trước đây là scikits.learn và còn được gọi là sklearn) là một dự án Google Summer of Code của David Cournapeau. Scikit-learn là một thư viện máy học phần mềm miễn phí cho ngôn ngữ lập trình Python và sử dụng rộng rãi cho đại số tuyến tính và mảng hiệu suất cao.

Hơn nữa, một số thuật toán cốt lõi được viết bằng Cython để cải thiện hiệu suất. . [3] Nó có các thuật toán phân loại , hồi quy và phân cụm khác nhau bao gồm các máy vectơ hỗ trợ , rừng ngẫu nhiên , tăng cường độ dốc , k -means và DBSCAN , và được thiết kế để tương tác với các thư viện khoa học và số của Python NumPy và Khoa học viễn tưởng .

Điểm nổi bật của Scikit-Learn:

- ✓ Khả năng trích xuất các đặc trưng từ hình ảnh và văn bản
- ✓ Tái sử dụng trong một số hoàn cảnh
- ✓ Một số phương thức để kiểm tra tính chính xác của các mô hình được giám sát trên dữ liệu chưa thấy
- ✓ Một loạt các thuật toán, bao gồm phân cụm, phân tích nhân tố, phân tích thành phần chính cho các mạng thần kinh không giám sát
- ✓ Scikit-học tích hợp tốt với nhiều thư viện Python khác, chẳng hạn như matplotlib và plotly, numpy cho vector mảng, pandas dataframes, scipy , và nhiều hơn nữa.

IV. Phân tích và giải quyết bài toán

1. Thu thập và phân tích đặc điểm dữ liệu

Dữ liệu hệ thống sử dụng thực tế trong quá trình vận hành là dữ liệu trên chính những website, fanpage của cửa hàng, nơi mà họ đang muốn phân tích đánh giá của người dùng.

Ở đây ta cần giải quyết bài toán phân loại văn bản thành 2 tích cực và tiêu cực(0: Negative, 1: Positive) và lượng dữ liệu cần cho quá trình huấn luyện được tốt thì không hề ít.

Do đó để tận dụng lượng dữ liệu có nhãn sẵn, tôi crawl dữ liệu trên trang web bán hàng của thế giới di động. Dữ liệu được crawl về bằng beautisoup dưới dạng HTML. Sau đó, trích suất những dữ liệu có đủ 2 thành phần là lời bình luận và nhãn từ đó ta thu được 3097 comment được đánh giá từ 1 đến 5 sao. Như vậy, bộ dữ liệu này hoàn toàn phù hợp cho nhiệm vụ classification nơi bạn có đầu vào là text và đầu ra là số lượng star, nơi thể hiện cảm xúc review của người dùng. Mình phân loại dữ liệu theo quy tắc:

- Nhỏ hơn 3 sao: Negative
- Từ 3 đến 5 sao: Positive

Từ đó, ta có bộ dữ liệu gồm 2364 bình luận tích cực và 733 bình luận tiêu cực, ví dụ như sau:

	Comment	Stars	Label
0	Lâu lâu âm thanh video bị nhỏ lại sau khi gọi điện thoại bang mocha call out...Điên thoại dùng được chỉ có lỗi đó làm sao khắc phục được là tốt	2	0
1	May quá mk mua xong là hết khuyến mại...! Giá bây giờ thì hơi chát...! Máy rất ok..! Chụp ảnh hay bị tối.....! Ko đc sáng ảnh...!	5	1
2	Máy rất đẹp. Ở ngoài đẹp hơn so với trong ảnh. Mọi thứ ok nhất là khoản pin khá trâu. Mua lúc giảm 1tr. Còn giá 4tr190k như bây giờ thì hơi chát	5	1
3	K dc ngon cho lắm nên giảm giá mạnh mong muốn samung thiết kế mặt kính sau lưng với giá tầm trung cho xịn và đẹp	1	0
4	Nói chung là tiền nào của đó ok trong tầm giá bền bỉ lước wep thì cũng tạm game online thì quá ok bin thì trâu khỏi nói luôn	5	1
5	Sau 5 ngày sử dụng mình có nhận xét như sau:	5	1
6	Mua đt đc 1 tuần, xài rất tuyệt nhưng lại thấy bắt đầu sạc chậm đi... buồn quá hichic. Cho em hỏi có cách nào khắc phục không ạ	4	1
7	Mua máy lúc 12h trưa tại điện máy xanh 211 khâm thiên đống đa hà nội,sau 12h sử dụng chơi liên quân pin tụt còn 68 % như vậy pin rất tốt,cột sóng sim 1 luôn chập chờn 1 đến 2 vạch trong khi sim 2 lại được 4 vạch,màn hình cảm ứng nhạy,cho 5 sao vì giảm giá 1 triệu và thái độ nhân viên phục tốt,nhất là nhân viên bảo vệ ngoài cửa,đi từ cửa vào đã muốn mua máy vì anh bảo vệ tươi cười thân thiện...	5	1
8	May sai rất ok dù mỗi nguoi hay thay lag j đó. Thì cũng có chút ít. Nhưng tien nao cua đó. Nhe sai bthuong thi may phan hồi rất ok..nhân viên tgdd..phu vu rất ok.nhất là mấy a.chi quang lý tư vân cho mk rất nhiệt tình...***** 5 sao cho tgdd.	5	1
9	Chán nhất khi quay trực tiếp cam nhòe nhoẹt, quay song xem lại ko rõ mặt. Làm thế nào để nhìn nét hơn ko	3	1
10	Lúc mua máy từ tháng 8 tới nay là được 5 tháng lúc đầu mua về mượn lắm nhưng sau có lúc bị lỗ giật vuốt k đi lun	1	0

❖ Đánh giá dữ liệu.

Nhận xét dữ liệu này là không tốt lắm vì số bình luận tiêu cực chỉ khoảng 733 trên tổng số 3096 comment. Như vậy chỉ có 24% là tiêu cực khi đó dữ liệu sẽ không cân bằng . Cũng dễ hiểu vì để trở thành một cửa hàng lớn thì nó phải có đủ nhiều khách hàng tin tưởng, hoặc có thể chính cửa

hàng tạo nên những bình luận tích cực như thuê người comment chẳng hạn. Vì cả 2 nhãn tích cực và tiêu cực có thể xem là quan trọng như nhau. không có biến nào trội hơn nên model của chúng ta trên data này có lẽ sẽ không tốt. Hơn nữa score này chỉ mang tính chất tượng trưng nên không dám chắc nó là tiêu chí phân loại tích cực và tiêu cực tốt.

Thêm vào đó, dữ liệu bình luận sản phẩm là văn nói, không ràng buộc, nên không có khuôn khổ về cách diễn đạt, thường xảy ra những lỗi câu chữ, lỗi chính tả, viết tắt và câu văn thường mang tính chất cá nhân,…. Do đó, bài toán phân tích tình cảm là bài toán khó đạt đến độ chính xác cao nhất. Bởi ngay cả trong đời sống, chính con người cũng khó phân biệt cảm xúc, quan điểm của đối phương chỉ qua một câu nói.

2. Tiền xử lý dữ liệu

Như đã phân tích dữ liệu ở trên, ta phải xử lý dữ liệu thô khá nhiều để có được một bộ dữ liệu huấn luyện đạt kết quả tốt:

- ✓ Chuẩn hóa về chữ thường
- ✓ Thay thế các url trong dữ liệu bởi nhãn URL_Change
- ✓ Thay thế các cấu trúc ngày tháng trong dữ liệu bởi nhãn DATE _Change
- ✓ Xóa các số lớn hơn 5(vì các số thường không quan trọng trong phân tích tình cảm trừ trường hợp nhiều comment sẽ nhắc đến cho 1* hay 5*)
- ✓ Loại bỏ dấu câu và các ký tự đặc biệt không phải alphabet
- ✓ Những từ thường xuyên xuất hiện sẽ không có nhiều thông tin nhưng vẫn có tỉ trọng(weight) ngang với các từ khác, cần giảm tỉ trọng về mặt thông tin nó xuống vì thông tin không mang nhiều giá trị.
- ✓ Những từ hiếm (rare word) không có sự khác biệt về tỉ trọng thông tin
- ✓ Dữ liệu sẽ được tách từ bằng ViTokenizer.tokenize .

	Comment	Stars	Label
0	lâu_lâu âm_thanh video bị nhỏ lại sau khi gọi điện_thoại_bang mocha call out . điên thoại dùng được chỉ có lỗi đó làm_sao khắc_phục được là tốt	2	0
1	may quá mk mua xong là hết khuyến_mại giá bây_giờ thì hơi chát máy rất ok ! chụp ảnh hay bị tối ko đc sáng ảnh	5	1
2	máy rất đẹp . ở ngoài đẹp hơn so với trong ảnh . mọi thứ ok nhất_là khoản pin khá trâu . mua lúc giảm 1tr . còn giá 4tr190k như bây_giờ thì hơi chát	5	1
3	k dc ngon cho lắm nên giảm_giá mạnh mong_muốn samung thiết_kế mặt kính sau lưng với giá tầm trung cho xịn và đẹp	1	0
4	nói_chung là tiền nào của đó ok trong tầm giá bền_bỉ lướt wep thì cũng tạm game online thì quá ok bin thì trâu khỏi nói luôn	5	1

5	sau 5 ngày sử_dụng mình có nhận_xét như sau :	5	1
6	mua đt đc 1 tuần , xài rất tuyệt nhưng lại thấy bắt_đầu sạc chậm đi . buồn quá hichic . cho em hỏi có cách nào khắc_phục không ạ	4	1
7	mua máy lúc 12h trưa tại điện_máy xanh 211 khâm thiên đống đa hà_nội , sau 12h sử_dụng chơi liên_quân pin tụt còn 68 % như_vậy pin rất tốt , cột sóng sim 1 luôn chập_chờn 1 đến 2 vạch trong khi sim 2 lại được 4 vạch , màn_hình cảm_ứng nhạy , cho 5 sao vì giảm_giá 1 triệu và thái_độ nhân_viên phục tốt , nhất là nhân_viên bảo_vệ ngoài cửa , đi từ cửa vào đã muốn mua máy vì anh bảo_vệ tươi_cười thân_thiện .	5	1
8	may sai rất ok dù mỗi nguoi hay thay lag j đó . thì cũng có chút_ít . nhưng tien nao cua đó . nhe sai bthuong thi may phan_hồi rất oknhân_viên tgddphu vu rất ok . nhất là mấy a . chi quang_lý tư vấn cho mk rất nhiệt_tình . * * * * * 5 sao cho tgdd .	5	1
9	chán nhất khi quay trực_tiếp cam nhòe_nhoẹt , quay song xem_lại ko rõ mặt . làm thế_nào để nhìn nét hơn ko	3	1
10	lúc mua máy từ tháng 8 tới nay là được 5 tháng lúc đầu mua về mượn lắm nhưng sau có lúc bị lỗ giật vuốt k đi lun	1	0

Sau khi thực hiện tuần tự và đầy đủ theo quy trình trên, ta thu được bộ dữ liệu sạch cho pha tiếp theo của mô hình. Chia dữ liệu theo tỉ lệ 80:20 để có được dữ liệu train và test.

3. Vector hóa dữ liệu

Bag of words (BoW) là cách biểu diễn bộ dữ liệu truyền thống phổ biến nhất được sử dụng trong xử lý ngôn ngữ tự nhiên. BoW được hiểu là một túi chứa tất cả các từ xuất hiện trong bộ dữ liệu, được sắp xếp một cách nhất định mà thường là theo thứ tự alphabet. Khi đó, mỗi câu trong bộ dữ liệu đều có thể biểu diễn được bằng một vector có số chiều bằng đúng số từ trong bộ từ vựng. Tại vị trí tương ứng với vị trí của từ đó trong túi từ, phần tử trong vector đó có thể sẽ được đánh dấu là 1/0 để thể hiện sự có hay không xuất hiện của từ đó trong câu, hoặc bằng chính số lần xuất hiện của từ đó trong câu.

Phương pháp này có nhược điểm là ta không thể xác định được nghĩa thực của mỗi từ và các từ tương quan với chúng. Các từ giống nhau sẽ được đánh trọng số như nhau. Phương pháp này không xét đến tần suất xuất hiện của từ hay những cảnh từ. Và trong thực tế, để cần hiểu được nghĩa của mỗi từ, ta cần xác định từ đó trong văn cảnh hơn là xét nghĩa độc lập từ. Tuy nhiên, phương pháp này khá cơ bản, dễ hiểu, và nhanh nên vẫn được sử dụng rộng rãi.

Trong bộ dữ liệu, ta xây dựng BoW sao cho túi từ không chứa các từ xuất hiện quá nhiều hoặc quá ít lần vì đó là những từ không mang nhiều ý nghĩa trong bài toán. Từ đó, ta giảm được đáng kể số chiều của vector, tăng tốc độ xử lý bài toán mà không ảnh hưởng nhiều đến độ chính xác của bài.

Sau đó, vector hóa từng câu trong bộ dữ liệu bằng túi từ, sao cho mỗi điểm vector là số lần xuất hiện của từ đó trong câu. Từ đó, ta được bộ dữ liệu dạng ma trận với số hàng là số câu trong bộ dữ liệu và số cột là số từ trong túi từ.

Ngoài ra, tôi cũng thử nghiệm vector hóa dữ liệu với thư viện Scikit-learn để so sánh kết quả giữa 2 cách xử lý.

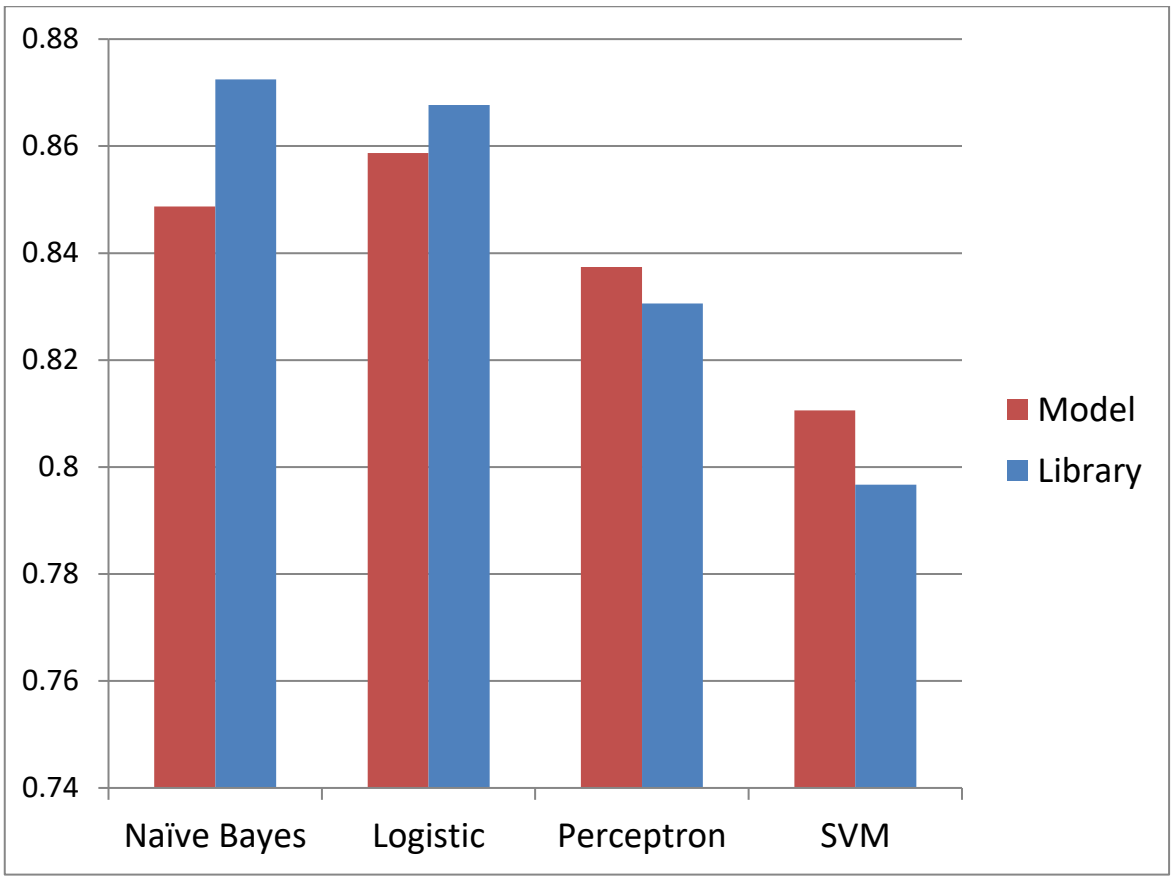
4. Xây dựng và huấn luyện mô hình

Như đã giới thiệu về các mô hình học máy chung ở trên, trong bài này tôi xử dụng 4 mô hình rất cơ bản của học máy đó là: Naïve Bayes, Logistic, Perceptron, SVM.

Đầu tiên, làm hoàn toàn bằng python và các thuật toán học máy, vector hóa dữ liệu bằng túi từ, lập trình từng bước với mô hình Logistic. Kết quả đạt được khá cao, trung bình độ chính xác bộ test đạt 84%. Trong đó, mô hình dự đoán được đúng 85% bình luận tích cực và 81% bình luận tiêu cực.

Sau đó, sử dụng thư viện Scikit-learn để huấn luyện tất cả 4 mô hình với 2 bộ dữ liệu vector hóa bằng túi từ và vector hóa bằng thư viện, tôi được kết quả sau:

		Naïve Bayes	Logistic	Perceptron	SVM
Traning	Model	0.8953	0.9591	0.9841	1
	Library	0.9144	0.9612	0.9911	0.9996
Testing	Model	0.8487	0.8587	0.8374	0.8106
	Library	0.8725	0.8677	0.8306	0.7967



5. Định hướng phát triển bài toán

Trong quá trình thực hiện, tôi nhận được nhiều lời nhận xét và gợi ý từ người hướng dẫn, thầy cô và cả cộng đồng lập trình,…Tôi nhận thấy bài làm của mình còn nhiều thiếu sót nhưng chưa đủ thời gian để hoàn thiện. Có những hướng khả quan để tôi có thể phát triển bài của mình:

- Thu thập dữ liệu nhiều và đa dạng hơn:

Dữ liệu tôi sử dụng là chuyên về bình luận sản phẩm điện thoại của một của hàng. Nếu tiếp tục theo hướng phân tích cảm xúc về mặt hàng này, ta có thể thu thập thêm dữ liệu từ nhiều cửa hàng điện thoại khác. Hoặc bài toán phân tích thái độ của khách hàng trên mọi mặt hàng nói chung, ta có thể thu thập thêm dữ liệu từ bất kỳ cửa hàng nào khác.

- Xử lý dữ liệu thô hiệu quả hơn:

Tiền xử lý dữ liệu cho bài toán này là một trong những vẫn đề quan trọng nhất, bởi như tôi đã nói về đặc tính dữ liệu của bài toán này. Quá trình thực hiện tôi mới chỉ xử lý được một phần mục tiêu của mình. Còn lại, tôi sẽ cần xử lý nhiều hơn và mức độ khó hơn với dữ liệu mình thu được:

- Loại bỏ những ký tự kéo dài: Ví dụ: Áo đẹp quáaaaaaaa--> Áo đẹp quá.
- Tiếng Việt có 2 cách bỏ dấu nên đưa về 1 chuẩn. Ví dụ, chữ "Hòa" và "Hoà" đều được chấp nhận trong tiếng Việt. Ngoài ra còn một số trường hợp lỗi font chữ cũng cần chuẩn hóa lại. (các

trường hợp dính chữ như: "Giao hàngnhanh" xử ý đc sẽ tốt hơn).

- Chuẩn hóa một số sentiment word: "okie"-->"ok", "okey"-->"ok", authentic--> "chuẩn chính hãng",vv...
- Emoj quy về 2 loại: emojs mang ý nghĩa tích cực (positive) và emojs mang nghĩa tiêu cực (nagative)
- Xử lý vấn đề phủ định. Ví dụ: Cái áo này rất đẹp và Cái áo này chẳng đẹp sẽ không khác nhau nhiều khi ta vector hóa bằng BoW
- Augmentation data bằng cách thêm vào các sample của chính tập train nhưng không dấu. (Bình luận không dấu khá phổ biến).
- Ngoài ra, ta có thể bổ sung vào tập train các sample mới lấy từ chính 2 từ điển positive và nagative.

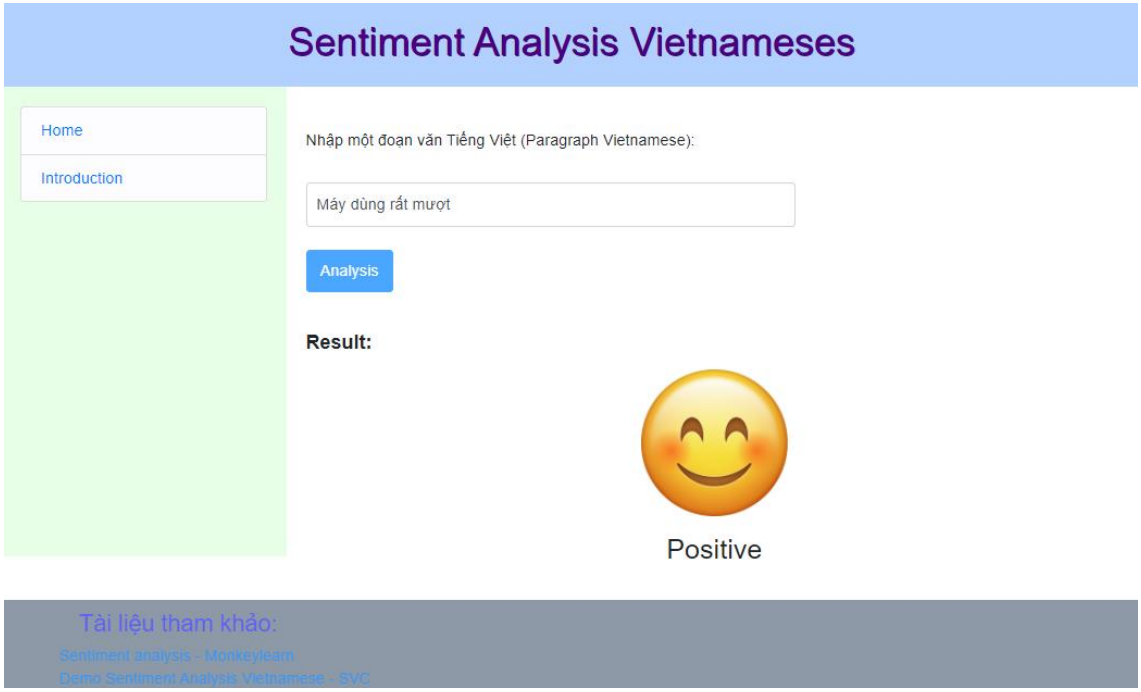
➢ Training với các mô hình khác, thư viện khác để tăng độ chính xác.

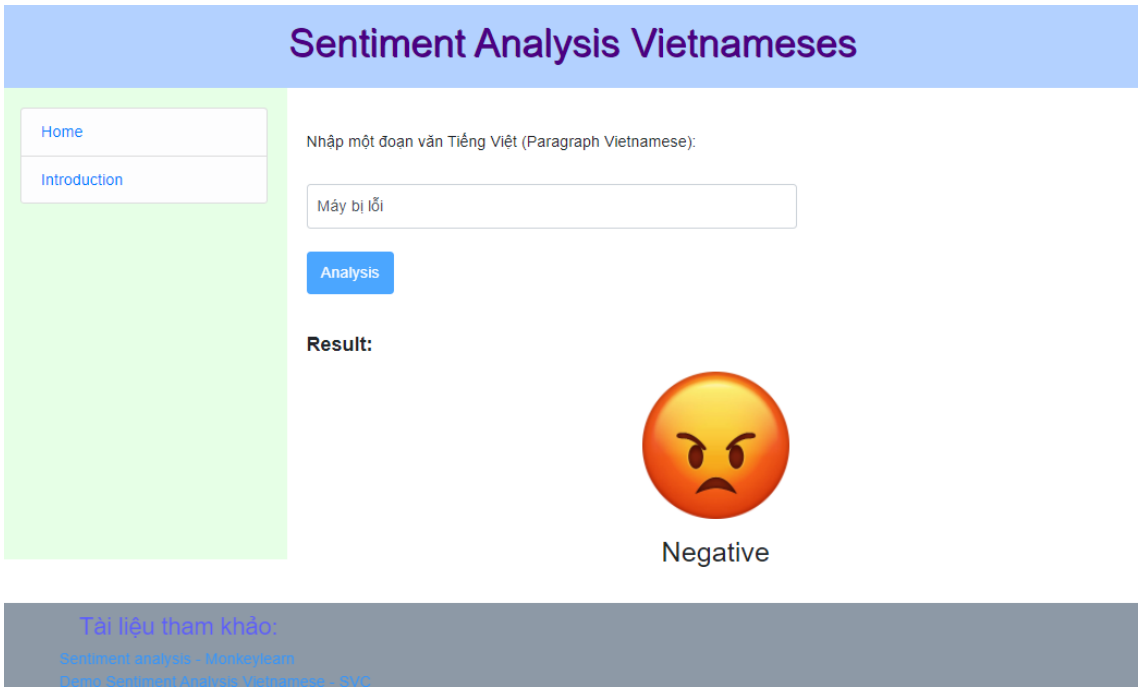
Sử dụng các thư viện học máy khác hay học sâu như TensorFlow, Keras, NTLK

Phương pháp Deep Learning Neural Network. Những thập niên gần đây, với sự phát triển nhanh chóng tốc độ xử lý của CPU, GPU và chi phí cho phần cứng ngày càng giảm, các dịch vụ hạ tầng điện toán đám mây ngày càng phát triển, làm tiền đề và cơ hội cho phương pháp học sâu Deep Learning Neural Network phát triển mạnh mẽ. Trong đó, bài toán phân tích cảm xúc đã được giải quyết bằng mô hình học Recurrent Neural Network (RNN) với một biến thể được dùng phổ biến hiện nay là Long Short Term Memory Neural Network (LSTMs), kết hợp với mô hình vector hóa từ (vector representations of words) Word2Vector với kiến trúc Continuous Bag-of-Words (CBOW). Ưu điểm của phương pháp này là văn bản đầu vào có thể là 1 câu hay 1 đoạn văn. Để thực hiện mô hình này đòi hỏi phải có dữ liệu văn bản càng nhiều càng tốt để tạo Word2Vector CBOW chất lượng cao và dữ liệu gán nhãn lớn để huấn luyện (training), xác minh (validate) và kiểm tra (test) mô hình học có giám sát (Supervise Learning) LSTMs.

➢ Viết api và giao diện cho dự án

Để trực quan hóa, tôi đã viết web api cho bài toán bằng flask với giao diện đơn giản như sau:


Sentiment Analysis Vietnameses
Home
Introduction
Nhập một đoạn văn Tiếng Việt (Paragraph Vietnamese):
Máy dùng rất mượt
Analysis
Result:
Positive
Tài liệu tham khảo:
Sentiment analysis - Monkeylearn
Demo Sentiment Analysis Vietnamese - SVC


Sentiment Analysis Vietnameses
Home
Introduction
Nhập một đoạn văn Tiếng Việt (Paragraph Vietnamese):
Máy bị lỗi
Analysis
Result:
Negative
Tài liệu tham khảo:
Sentiment analysis - Monkeylearn
Demo Sentiment Analysis Vietnamese - SVC

V. Kết quả đạt được và lời cảm ơn

Trong quá trình thực tập tại công ty Aimesof, tôi đã học được rất nhiều thứ. Khác với những lý thuyết khô khan trên giảng đường về học máy, thì thực tập là cơ hội để ta áp dụng những lý thuyết vào bài toán thực tế. Đó là cách trực quan nhất để ta hiểu sâu hơn về nó và ứng dụng của nó. Tại đây, lần đầu tiên tôi học được một quy trình đầy đủ của một dự án nhỏ. Bắt đầu từ việc phân tích, đánh giá để đưa ra giải pháp cho đến việc thực hành. Từ đó, tôi rèn luyện cho mình tính tỉ mỉ và khoa học để không bỏ sót những vấn đề quan trọng trong bài toán.

Cũng từ công ty, tôi học được ở các anh chị rất nhiều về chuyên môn, về kỹ năng làm việc và rèn luyện cho bản thân cả những kỹ năng mềm cần thiết. Các nhân viên ở đó dù say mê công việc nhưng cũng rất hòa đồng và nhiệt tình. Bởi vậy, đến đây thực tập tôi đã được quan tâm và giúp đỡ rất nhiều để kì thực tập kết thúc bổ ích. Sau khóa thực tập tôi tự tin hơn khi giao tiếp; khả năng trình bày vấn đề, hiểu vấn đề cho đến khả năng đặt câu hỏi, hay tìm kiếm cũng tiến bộ hơn nhiều.

Cảm ơn Aimesoft đã giúp tôi hoàn thành khóa thực tập với sự tiến bộ lớn . Em cũng xin gửi lời cảm ơn chân thành đến thầy cô, đã tạo điều kiện cho chúng em biết đến và được làm việc trong những môi trường tốt nhất. Cảm ơn các thầy cô đã quan tâm, hỏi han và giúp đỡ chúng em những khó khăn trong quá trình thực tập.

VI. Nhận xét của công ty thực tập

- Công ty Cổ phần Aimesoft: Tầng 3, tòa nhà Hoàng Ngọc, số 4 ngõ 82 phố Dịch Vọng Hậu, Cầu Giấy, Hà Nội.
- Người hướng dẫn: anh Phạm Thế Quyền .

  Số điện thoại: 0347385423

  Email: quyenpt@aimesoft.com

❖ Nhận xét về quá trình thực tập:

➢ **Xác nhận về công việc được giao trong báo cáo:**

.....

.....

.....

.....

➢ **Xác nhận về kết quả đạt được:**

.....

.....

.....

.....

.....

➢ **Ý thức và thái độ làm việc trong quá trình làm việc tại công ty:**

Nhận xét của người hướng dẫn: .....

.....

.....

.....

Nhận xét từ ban quản lý nhân sự của công ty: .....

.....

.....

.....

.....

- **Góp ý với sinh viên nhà trường để công việc đạt hiệu quả tốt hơn:**

.....

.....

.....

Điểm số đánh giá theo thang 10: .....

Xác nhận của công ty Xác nhận của người hướng dẫn