TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**
**KHOA CÔNG NGHỆ THÔNG TIN**

# TÌM HIỂU H Ệ THỐNG PHÂN LOẠI TIN TỨC TỰ ĐỘNG

**HKIII (2020 – 2021)**

*Người hướng dẫn*: **THẦY DƯƠNG HỮU PHÚC**
*Người thực hiện*: **CAO MINH PHÁT -517H0153**
**NGUYỄN TRỌNG HIẾU -517H0121**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2021**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**
**KHOA CÔNG NGHỆ THÔNG TIN**

# TÌM HIỂU H Ệ THỐNG PHÂN LOẠI TIN TỨC TỰ ĐỘNG

**HKIII (2020 – 2021)**

*Người hướng dẫn*: **THẦY DƯƠNG HỮU PHÚC**
*Người thực hiện*: **CAO MINH PHÁT -517H0152**
**NGUYỄN TRỌNG HIẾU -517H0121**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2021**

# LỜI CẢM ƠN

Bài đồ án cuối kì được hoàn thành tại trường đại học Tôn Đức Thắng. Trong quá trình làm đồ án cuối kì, chúng em đã nhận được rất nhiều sự giúp đỡ để hoàn tất sản phẩm của mình.

Trước hết chúng em xin gửi lời cảm ơn chân thành đến thầy Dương Hữu Phúc đã tận tình hướng dẫn, truyền đạt những kiến thức, kinh nghiệm cho chúng em trong suốt quá trình thực hiện đồ án cuối kì này.

Sau cùng xin gửi lời cảm ơn đến thầy cô và các bạn sinh viên khác đã luôn động viên, giúp đỡ chúng em trong quá trình làm đồ án. Đồng thời xin gửi lời cám ơn đến các bạn sinh viên đã vui vẻ, nhiệt tình tham gia trả lời câu hỏi khảo sát giúp em hoàn thành sản phẩm này.

Một lần nữa, xin chân thành cảm ơn!

# ĐỒ ÁN ĐƯỢC HOÀN THÀNH TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

Chúng tôi xin cam đoan đây là sản phẩm đồ án của riêng chúng tôi và được sự hướng dẫn của thầy Dương Hữu Phúc. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày 23 tháng 6 năm 2021*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

# PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

**Phần xác nhận của GV hướng dẫn**

_____

_____

_____

_____

_____

_____

_____

_____

Tp. Hồ Chí Minh, ngày  tháng  năm
(kí và ghi họ tên)

**Phần đánh giá của GV chấm bài**

_____

_____

_____

_____

_____

_____

_____

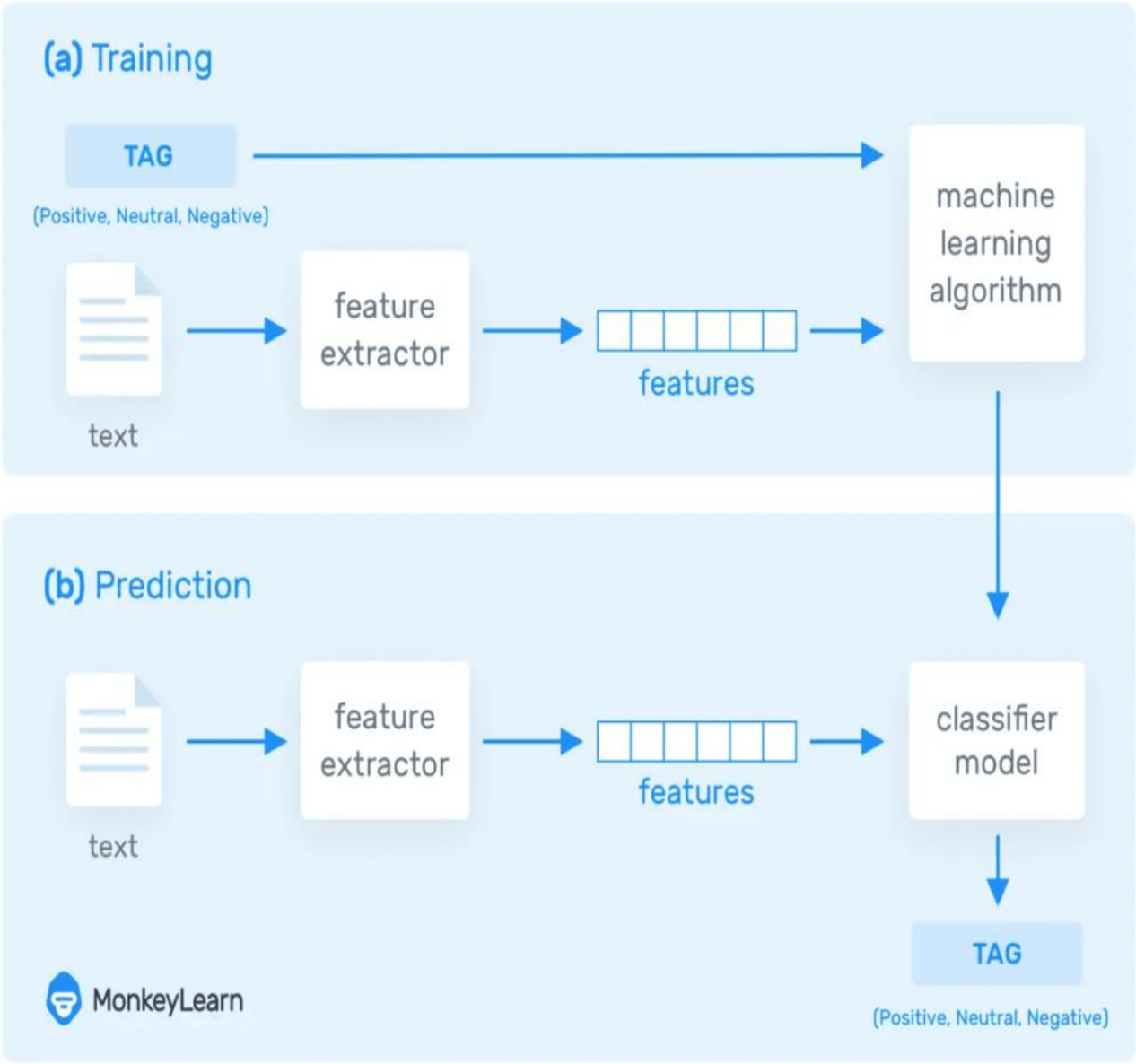
_____

Tp. Hồ Chí Minh, ngày  tháng  năm
(kí và ghi họ tên)

# MỤC LỤC

# I – TỔNG QUAN

Phân loại văn bản (Text Classification) là bài toán thuộc nhóm học có giám sát (Supervised learning) trong học máy. Bài toán này yêu cầu dữ liệu cần có nhãn (label). Mô hình sẽ học từ dữ liệu có nhãn đó, sau đó được dùng để dự đoán nhãn cho các dữ liệu mới mà mô hình chưa gặp.




- Giai đoạn (a): Huấn luyện (training) là giai đoạn học tập của mô hình phân loại văn bản. Ở bước này, mô hình sẽ học từ dữ liệu có nhãn (trong ảnh trên nhãn là Possitive, Negative, Neutral). Dữ liệu văn bản sẽ được số hóa thông qua bộ trích xuất đặc trưng (feature extractor) để mỗi mẫu

dữ liệu trong tập huấn luyện trở thành 1 vector nhiều chiều (đặc trưng). Thuật toán máy học sẽ học và tối ưu các tham số để đạt được kết quả tốt trên tập dữ liệu này. Nhãn của dữ liệu được dùng để đánh giá việc mô hình học tốt không và dựa vào đó để tối ưu.

- Giai đoạn (b): Dự đoán (prediction), là giai đoạn sử dụng mô hình học máy sau khi nó đã học xong. Ở giai đoạn này, dữ liệu cần dự đoán cũng vẫn thực hiện các bước trích xuất đặc trưng. Mô hình đã học sau đó nhận đầu vào là đặc trưng đó và đưa ra kết quả dự đoán.

# II – THỰC HÀNH PHÂN LOẠI VĂN BẢN

## 2.1 Tiền sử lý dữ liệu

Trong qui trình khai phá dữ liệu, công việc xử lý dữ liệu trước khi đưa vào các mô hình là rất cần thiết, bước này làm cho dữ liệu có được ban đầu qua thu thập dữ liệu (gọi là dữ liệu gốc original data) có thể áp dụng được (thích hợp) với các mô hình khai phá dữ liệu (data mining model) cụ thể. Các công việc cụ thể của tiền xử lý dữ liệu bao gồm những công việc như:

- Filtering Attributes: Chọn các thuộc tính phù hợp với mô hình
- Filtering samples: Lọc các mẫu (instances, patterns) dữ liệu cho mô hình
- Clean data: Làm sạch dữ liệu như xóa bỏ các dữ liệu bất thường (Outlier) Transformation: Chuyển đổi dữ liệu cho phù hợp với các mô hình như chuyển đổi dữ liệu từ numeric qua nomial hay ordinal
- Discretization (rời rạc hóa dữ liệu): Nếu bạn có dữ liệu liên tục nhưng một vài mô hình chỉ áp dụng cho các dữ liệu rời rạc (như luật kết hợp chẳn hạn) thì bạn phải thực hiện việc rời rạc hóa dữ liệu.

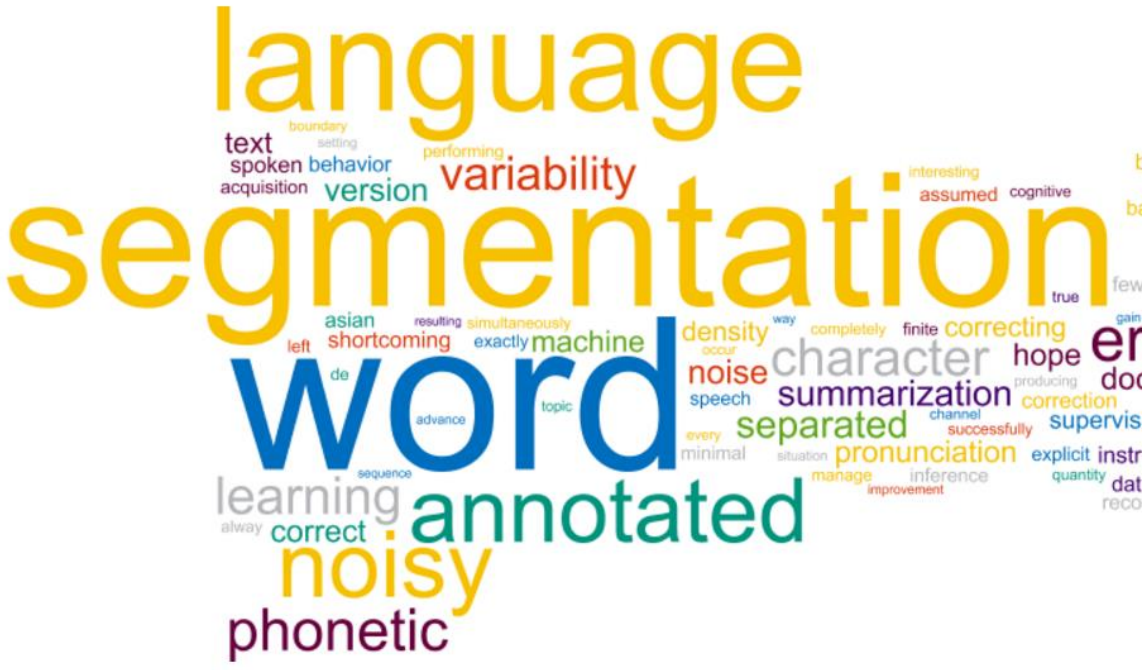
Bước tiền xử lý dữ liệu là bước đầu tiên cần làm, nó cần được làm trước bước feature extractor. Việc tiền sử lý dữ liệu là quá trình chuẩn hóa dữ liệu và loại bỏ các thành phần không có ý nghĩa cho việc phân loại văn bản.
Tiền xử lý dữ liệu tiếng Việt cho bài toán phân loại văn bản thường gồm các việc sau:

- Xóa HTML code (nếu có)
- Chuẩn hóa bảng mã Unicode (đưa về Unicode tổ hợp dựng sẵn)
- Chuẩn hóa kiểu gõ dấu tiếng Việt (dùng òa úy thay cho oà uý)
- Thực hiện tách từ tiếng Việt (sử dụng thư viện tách từ như pyvi, underthesea, vncorenlp,…)
- đưa về văn bản lower (viết thường)
- Xóa các ký tự đặc biệt: “.”, “,”, “;”, “)”, …

## 2.2 Tách từ tiếng Việt

Thuật ngữ "tách từ" trong Tiếng Anh là "word segmentation", dịch ra Tiếng Việt là "tách từ" hoặc "phân đoạn từ". Dùng từ nào cũng được, nhưng chúng tôi dùng từ "tách từ" trong bài viết này.

Tách từ, về mặt biểu hiện, là gom nhóm các từ đơn liền kề thành một cụm từ có ý nghĩa. Ví dụ: "Cách tách từ cho Tiếng Việt." sau khi tách từ thì thành "Cách tách từ cho Tiếng_Việt ." Về hình thức, các từ đơn được gom nhóm với nhau bằng cách nối với nhau bằng ký tự gạch dưới "_", trong trường hợp này là từ Tiếng_Việt. Sau khi thực hiện tách từ thì mỗi từ (token) trong câu được cách nhau bởi một khoảng trắng, trong trường hợp này như "Tiếng_Việt ." thì từ "Tiếng_Việt" cách đấu "." bởi 1 khoảng trắng. Đây là quy ước chung cho tất cả các ngôn ngữ của bài toán tách từ trong xử lý ngôn ngữ tự nhiên. Việc quy ước như vậy là để tạo thành chuẩn chung và để dễ xử lý hơn trong lập trình.




Tại sao phải tách từ cho Tiếng Việt?

Về mặt ngữ nghĩa, việc tách từ văn bản đầu vào trước khi đưa vào huấn luyện mô hình máy học là để giải quyết các bài toán liên quan đến ngữ nghĩa của văn bản, tức là kết quả đầu ra mang tính suy luận dựa trên việc hiểu ý nghĩa của văn bản đầu vào. Ví dụ như các dạng bài toán: phát hiện đạo văn, tóm tắt

văn bản, hỏi đáp tự động, hỗ trợ khách hàng tự động, phân tích cảm xúc văn bản, dịch máy, trợ lý ảo...

Mục tiêu của việc tách từ văn bản đầu vào là để khử tính nhập nhằng về ngữ nghĩa của văn bản. Tùy vào từng loại ngôn ngữ có những đặc điểm khác nhau mà việc tách từ văn bản cũng có độ khó khăn khác nhau. Đựa theo đặc điểm của ngôn ngữ tự nhiên mà ngôn ngữ được phân thành các loại:

- Ngôn ngữ hòa kết (flexional), ví dụ: Đức, Latin, Hi Lạp, Anh, Nga...
- Ngôn ngữ chắp dính (agglutinate), ví dụ: Thổ Nhĩ Kỳ, Mông Cổ, Nhật Bản, Triều Tiên,...
- Ngôn ngữ đơn lập (isolate), là ngôn ngữ phi hình thái, không biến hình, đơn âm tiết, ví dụ: Việt Nam, Hán,...

Với ngôn ngữ hòa kết như Tiếng Anh, thì việc tách từ khá đơn giản vì ranh giới từ được nhận diện bằng khoảng trắng và dấu câu.

Với ngôn ngữ Tiếng Việt, thuộc loại hình đơn lập, mang đặc điểm là từ Tiếng Việt không biến đổi hình thái, ranh giới từ không được xác định mặc nhiên bằng khoản trắng. Tiếng Việt có đặc điểm là ý nghĩa ngữ pháp nằm ở ngoài từ, phương thức ngữ pháp chủ yếu là trật tự từ và từ hư. Cho nên có trường hợp một câu có thể có nhiều ngữ nghĩa khác nhau tuỳ vào cách ta tách từ như thế nào, gây nhập nhằng về ngữ nghĩa của câu. Ví dụ:

Với câu "Xoài phun thuốc sâu không ăn." có thể được tách từ như sau, với ý nghĩa hoàn toàn khác nhau:
Xoài / phun thuốc / sâu / không / ăn.
Xoài / phun / thuốc sâu / không / ăn.

Với câu "Ăn cơm không được uống rượu.", có thể được tách từ như sau:
Ăn / cơm / không / được / uống / rượu.
Ăn / cơm không / được / uống / rượu.

Với câu "Mẹ vào ca ba con ngủ với dì.", có thể được tách từ như sau:
Mẹ / vào / ca ba / con / ngủ / với / dì.
Mẹ / vào ca / ba con / ngủ / với / dì.

Điều đó cho thấy, công việc tách từ trong Tiếng Việt không phải là chuyện dễ dàng, vì nó tạo ra các câu có ngữ nghĩa hoàn toàn khác nhau, gây ảnh hưởng đến chất lượng huấn luyện mô hình học. Chính vì vậy, công việc tách từ là rất

quan trọng đối với xử lý ngôn ngữ Tiếng Việt, nhất là khi giải quyết các bài toán liên quan đến ngữ nghĩa của văn bản.

## 2.3 Loại bỏ stopword

Trong máy tính, từ dừng là những từ được lọc ra trước hoặc sau khi dữ liệu ngôn ngữ tự nhiên (văn bản) được xử lý. Trong khi "stop words" thường đề cập đến các từ phổ biến nhất trong một ngôn ngữ, các công cụ xử lý ngôn ngữ hoàn toàn tự nhiên không sử dụng một danh sách các từ dừng phổ biến.

Trong máy tính, từ dừng là những từ được lọc ra trước hoặc sau khi dữ liệu ngôn ngữ tự nhiên (văn bản) được xử lý. Trong khi "các từ dừng" thường đề cập đến các từ phổ biến nhất trong một ngôn ngữ, các công cụ xử lý ngôn ngữ hoàn toàn tự nhiên không sử dụng một danh sách các từ dừng phổ biến.

Stopword là những từ trong bất kỳ ngôn ngữ nào không bổ sung nhiều ý nghĩa cho một câu. Chúng có thể được bỏ qua một cách an toàn mà không làm mất đi ý nghĩa của câu. Đối với một số công cụ tìm kiếm, đây là một số từ chức năng ngắn, phổ biến nhất, chẳng hạn như, is, at, which, and on. Trong trường hợp này, các từ dừng có thể gây ra vấn đề khi tìm kiếm các cụm từ bao gồm chúng, đặc biệt là trong các tên như "The Who" hoặc "Take That"


.

Khi nào loại bỏ stopword?

Nếu là nhiệm vụ phân loại văn bản hoặc phân tích tình cảm thì nên xóa các từ dừng vì chúng không cung cấp bất kỳ thông tin nào cho mô hình của chúng tôi, tức là loại bỏ các từ không mong muốn ra khỏi kho ngữ liệu của chúng tôi, nhưng nếu chúng tôi có nhiệm vụ dịch ngôn ngữ thì các từ dừng sẽ hữu ích, vì chúng phải được dịch cùng với các từ khác.

Không có quy tắc cứng và nhanh về thời điểm loại bỏ các từ dừng. Nhưng tôi khuyên bạn nên loại bỏ các từ dừng nếu nhiệm vụ của chúng ta phải thực hiện là một trong các Phân loại ngôn ngữ, Lọc thư rác, Tạo phụ đề, Tạo thẻ tự động, Phân tích tình cảm hoặc một thứ gì đó liên quan đến phân loại văn bản.

Mặt khác, nếu nhiệm vụ của chúng ta là một trong các vấn đề về Dịch máy, Trả lời câu hỏi, Tóm tắt văn bản, Lập mô hình ngôn ngữ, thì tốt hơn hết bạn không nên xóa các từ dừng vì chúng là một phần quan trọng của các ứng dụng này.

Ưu điểm:

- Các từ dừng thường bị xóa khỏi văn bản trước khi đào tạo mô hình học sâu và học máy vì các từ dừng xuất hiện rất nhiều, do đó cung cấp rất ít hoặc không có thông tin duy nhất có thể được sử dụng để phân loại hoặc phân cụm.
- Khi loại bỏ các từ dừng, kích thước tập dữ liệu giảm và thời gian đào tạo mô hình cũng giảm mà không ảnh hưởng lớn đến độ chính xác của mô hình.
- Loại bỏ từ khóa có khả năng giúp cải thiện hiệu suất, vì có ít hơn và chỉ còn lại các mã thông báo quan trọng. Do đó, độ chính xác phân loại có thể được cải thiện

Khuyết điểm:

- Việc lựa chọn và loại bỏ các từ dừng không đúng cách có thể thay đổi ý nghĩa của văn bản của chúng ta. Vì vậy, chúng ta phải cẩn thận trong việc lựa chọn từ dừng của mình.

Ví dụ: “Bộ phim này không hay.”
Nếu chúng ta loại bỏ (không phải) trong bước xử lý trước, câu (phim này hay) cho biết nó là khẳng định nhưng bị diễn giải sai.

## 2.4 Xây dựng tập train/test

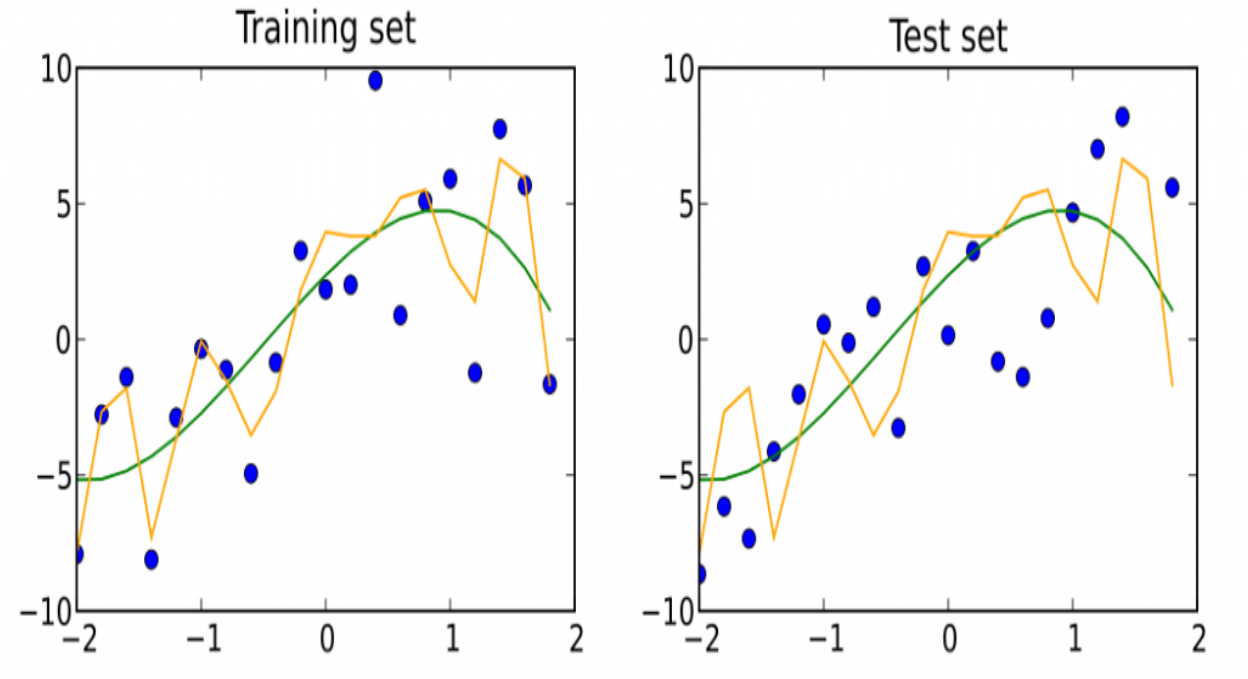
Training Set (Tập huấn luyện):

Tập huấn luyện (training set) là tập dữ liệu được sử dụng để huấn luyện mô hình. Các thuật toán học máy sẽ học các mô hình từ tập huấn luyện này. Việc học sẽ khác nhau tùy thuộc vào thuật toán và mô hình sử dụng. Ví dụ, khi sử dụng môt hình Hồi quy tuyến tính (Linear Regression), các điểm trong tập huấn luyện được sử dụng để tìm ra hàm số hay đường phù hợp nhất mô tả quan hệ giữa đầu vào và đầu ra của tập dữ liệu huấn luyện bằng cách sử dụng một số phương pháp tối ưu hóa như công thức nghiệm ở bài trước hoặc các thuật toán tối ưu gần đúng như gradient descent hay stochastic gradient descent. Trong thuật toán K-Nearest Neighbors (K-Hàng xóm gần nhất), các điểm trong tập huấn luyện là những điểm có thể là hàng xóm của nhau (gần nhau) được học theo các phương pháp tham lam. Trong thực tế, tập dữ liệu huấn luyện thường bao gồm các cặp vectơ đầu vào và vectơ đầu ra tương ứng, trong đó vector đầu ra thường được gọi là nhãn (label hoặc target). Các thuật toán nói chung sẽ tìm cách tối ưu sai số dự đoán trên tập huấn luyện này đến mức đủ tốt. Trong trường hợp overfitting sai số dự đoán của mô hình trên tập huấn luyện có thể rất thấp, thậm chí = 0%.

Testing Set (Tập kiểm thử)
Nhưng điều này chưa đủ. Mục tiêu của machine learning là tạo ra những mô hình có khả năng tổng quát hóa để dự đoán tốt trên cả dữ liệu chưa thấy bao giờ (nằm ngoài tập huấn luyện), do đó, để biết một thuật toán hay mô hình có tốt hay không thì sau khi được huấn luyện, mô hình cần được đánh giá hiệu quả thông qua bộ dữ liệu kiểm thử (testing set). Bộ dữ liệu này được sử dụng để tính độ chính xác hoặc sai số của mô hình dự đoán đã được huấn luyện. Chúng ta biết nhãn thực của mọi điểm trong tập hợp dữ liệu kiểm thử này, nhưng chúng ta sẽ tạm thời giả vờ như không biết và đưa các giá trị đầu vào của tập vào mô hình dự đoán để nhận kết quả dự đoán đầu ra. Sau đó chúng ta có thể nhìn vào các nhãn thực và so sánh nó với kết quả dự đoán của các đầu vào tương ứng này và xem liệu mô hình có dự đoán đúng hay không. Việc tính tổng trung bình của toàn bộ các lỗi này chúng ta có thể tính toán được lỗi dự đoán trên tập kiểm thử.

Các lỗi dự đoán này được đánh giá thông qua rất nhiều chỉ số khác nhau như độ chính xác (precision), độ hồi tưởng (recall), F1-Score, RMSE, MAE,…Các chỉ số này sẽ được trình bày chi tiết hơn trong bài tiếp theo. Một lưu ý là các chỉ số đo mức độ hiệu quả của mô hình này trên tập kiểm thử có thể khác với các lossfunction hay objective function sử dụng để tối ưu hóa mô hình trên tập huấn luyện. Nghĩa là quá trình kiểm thử và quá trình huấn luyện là hoàn toàn độc lập với nhau, cả về bộ dữ liệu lẫn cách thức so sánh chỉ số.

Tập dữ liệu kiểm thử tốt là một tập dữ liệu độc lập với tập dữ liệu huấn luyện (để ngoài và không được tham gia vào quá trình huấn luyện), nhưng tuân theo cùng một phân phối xác suất như tập dữ liệu huấn luyện. Điều này giúp cho việc đánh giá không bị thiên vị. Nếu một mô hình phù hợp với trainining set nhưng lại sai khác trên testing set, thì việc rất có khả năng nó bị overfitting (xem hình bên dưới). Ngược lại, sai số không quá nhiều thì thường chúng là một mô hình phù hợp.

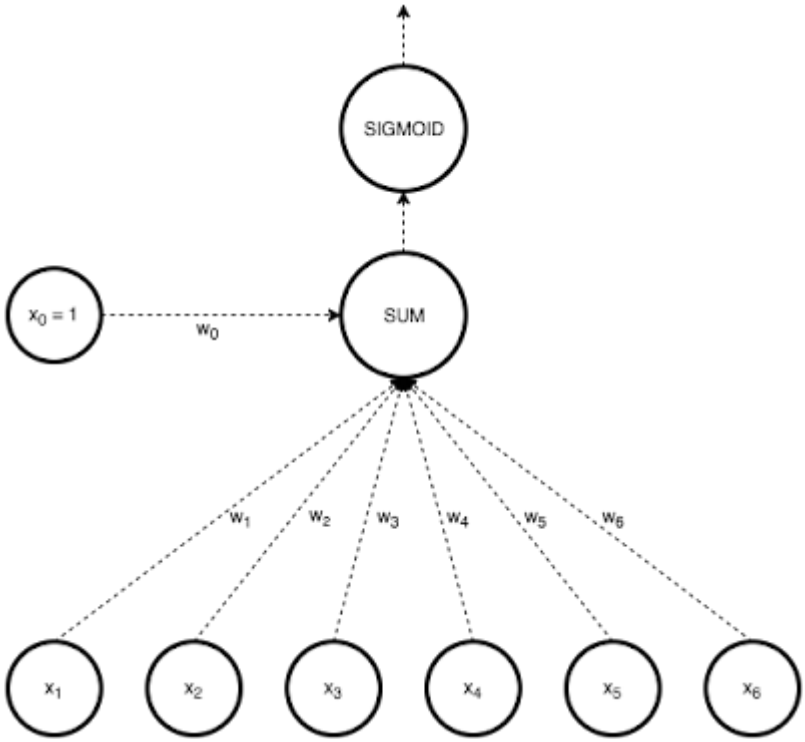



Với mô hình màu xanh lá, lỗi dự đoán trên cả training set và testing set là không chênh lệch nhau quá nhiều, ngược lại với mô hình màu cam, lỗi dự đoán trên testing set lớn hơn rất nhiều so với trên training set. Do đó mô hình màu cam nhiều khả năng bị overfitting hơn mô hình màu xanh

# III – THUẬT TOÁN

## 3.1 Logistic Regression

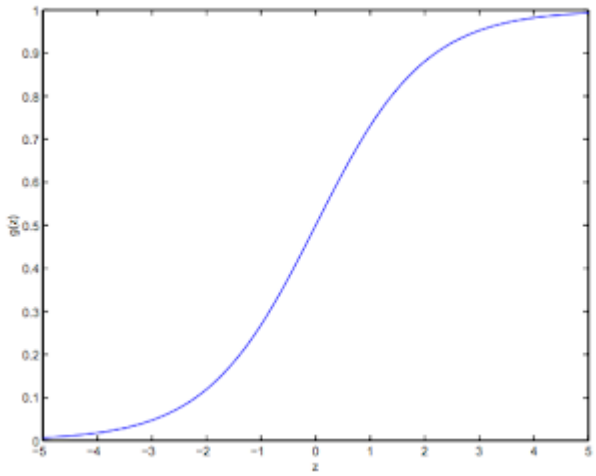
Logistic Regression là một kỹ thuật phân loại cơ bản. Nó thuộc nhóm linear classifers và hơi giống với polynomial và linear regression. Logistic Regression nhanh và tương đối không phức tạp, đồng thời thuận tiện cho bạn trong việc diễn giải kết quả. Mặc dù về cơ bản đây là một phương pháp để phân loại nhị phân, nhưng nó cũng có thể được áp dụng cho các bài toán đa lớp.




Hàm sigmoid:

Hãy xem xét một hàm $h(z)$ có dạng: $h(z) = \frac{1}{1+e^{-z}}$

$h(z)$ được gọi là hàm sigmoid. Đồ thị $h(z)$ với z được thể hiện trong hình. Chúng ta thấy rằng h (z) tăng đều từ 0 đến 1 khi z đi từ -inf đến + inf

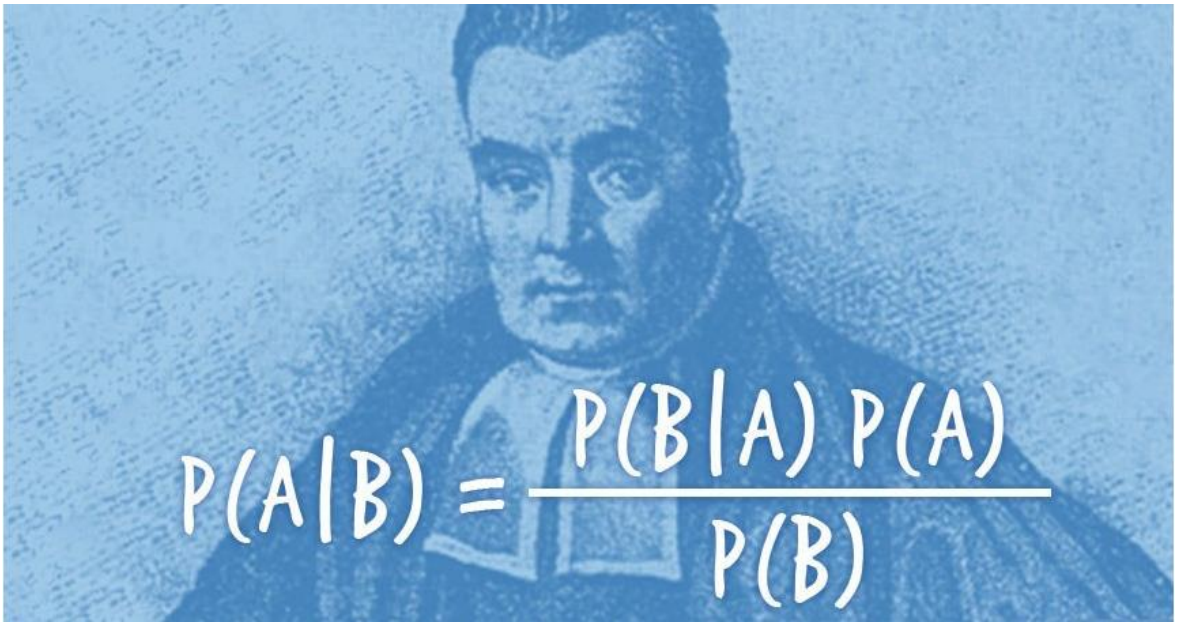



Trong Logistic Regression, chúng ta sử dụng hàm sigmoid này làm hàm kích hoạt giống như chúng ta đã sử dụng hàm ngưỡng trong perceptron. Hàm logistic được cung cấp một sự kết hợp tuyến tính có trọng số của các tính năng vì đầu vào và đầu ra là một số từ 0 đến 1

$$h(w,x) = \frac{1}{1 + e^{-w^T x}}$$

### 3.2 Naive Bayes

Naive Bayes Classification (NBC) – thuật toán phân loại Naive Bayes - là một thuật toán dựa trên định lý Bayes về lý thuyết xác suất để đưa ra các phán đoán cũng như phân loại dữ liệu dựa trên các dữ liệu được quan sát và thống kê, được ứng dụng rất nhiều trong các lĩnh vực Machine learning dùng để đưa các dự đoán có độ chính xác cao, dựa trên một tập dữ liệu đã được thu thập. NBC thuộc vào nhóm học máy có giám sát.




Định lý Bayes cho phép tính xác suất xảy ra của một sự kiện ngẫu nhiên A khi biết sự kiện liên quan B đã xảy ra. Xác suất này được ký hiệu là P(A|B), và đọc là "xác suất của A nếu có B". Đại lượng này được gọi xác suất có điều kiện hay xác suất hậu nghiệm vì nó được rút ra từ giá trị được cho của B hoặc phụ thuộc vào giá trị đó.

$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)}$$

Theo định lí Bayes, P(A|B) sẽ phụ thuộc vào 3 yếu tố:
- Xác suất xảy ra A của riêng nó, không quan tâm đến B. Kí hiệu là P(A).
- Xác suất xảy ra B của riêng nó, không quan tâm đến A. Kí hiệu là P(B).
- Xác suất xảy ra B khi biết A xảy ra. Kí hiệu là P(B|A). Đại lượng này gọi là khả năng (likelihood) xảy ra B khi biết A đã xảy ra.

Ở trên ta có thể thấy xác suất xảy ra của sự kiện A phụ thuộc và xác suất của sự kiện B, nhưng trong thực tế xác suất A có thể phụ thuộc vào xác suất của nhiều các giác thuyết khác có thể là B1, B2, B3 … Bn. Vậy định luật Bayes có thể được mở rộng bằng công thức sau:

$$P(A|B) = \frac{(P(B_1|A) \times P(B_2|A) \times P(B_3|A) \ldots \times P(B_n|A)) \times P(A)}{P(B_1) \times P(B_2) \times P(B_3) \ldots \times P(B_n)}$$

Trên thực tế thì ít khi tìm được dữ liệu mà các thành phần là hoàn toàn độc lập với nhau. Tuy nhiên giả thiết này giúp cách tính toán trở nên đơn giản, training data nhanh, đem lại hiệu quả bất ngờ với các lớp bài toán nhất định.

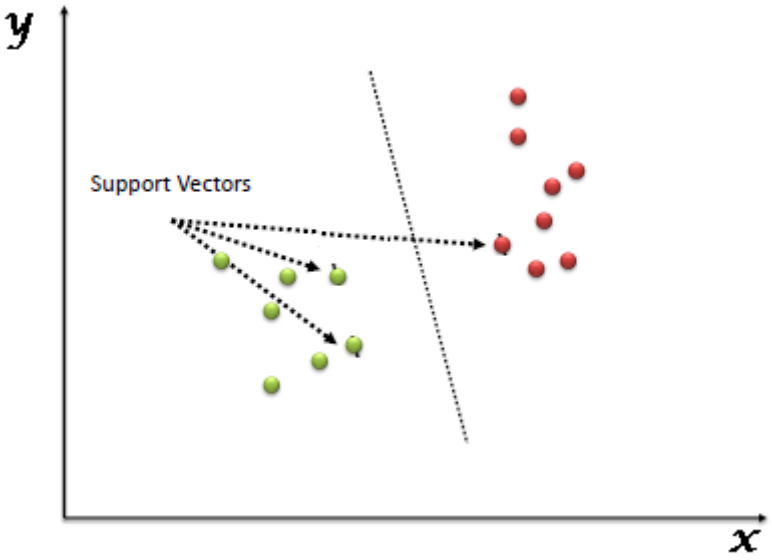
**Ứng dụng của thuật toán Naïve Bayes**

Thuật toán Naive Bayes Classification được áp dụng vào các loại ứng dụng sau:

- Real time Prediction: NBC chạy khá nhanh nên nó thích hợp áp dụng ứng dụng nhiều vào các ứng dụng chạy thời gian thực, như hệ thống cảnh báo phát hiện sự cố...
- Multi class Prediction: Nhờ vào định lý Bayes mở rộng ta có thể ứng dụng vào các loại ứng dụng đa dự đoán, tức là ứng dụng có thể dự đoán nhiều giả thuyết mục tiêu.
- Text classification/ Spam Filtering/ Sentiment Analysis: NBC cũng rất thích hợp cho các hệ thống phân loại văn bản hay ngôn ngữ tự nhiên vì tính chính xác của nó lớn hơn các thuật toán khác. Ngoài ra các hệ thống chống thư rác cũng rất ưu chuộng thuật toán này. Và các hệ thống phân tích tâm lý thị trường cũng áp dụng NBC để tiến hành phân tích tâm lý người dùng ưu chuộng hay không ưu chuộng các loại sản phẩm nào từ việc phân tích các thói quen và hành động của khách hàng.

## 3.3 Support Vector Machine (SVM)

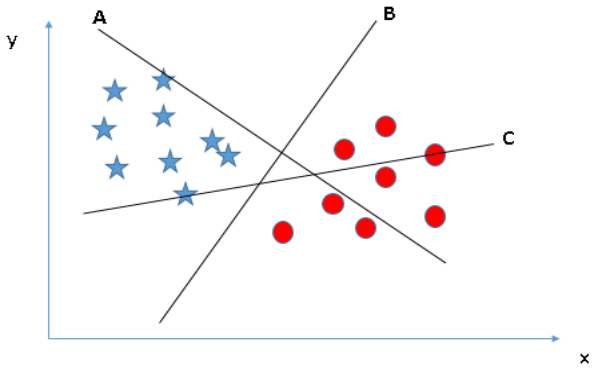
SVM là một thuật toán giám sát, nó có thể sử dụng cho cả việc phân loại hoặc đệ quy. Tuy nhiên nó được sử dụng chủ yếu cho việc phân loại. Trong thuật toán này, chúng ta vẽ đồi thị dữ liệu là các điểm trong n chiều ( ở đây n là số lượng các tính năng bạn có) với giá trị của mỗi tính năng sẽ là một phần liên kết. Sau đó chúng ta thực hiện tìm "đường bay" (hyper-plane) phân chia các lớp. Hyper-plane nó chỉ hiểu đơn giản là 1 đường thẳng có thể phân chia các lớp ra thành hai phần riêng biệt.




Support Vectors hiểu một cách đơn giản là các đối tượng trên đồ thị tọa độ quan sát, Support Vector Machine là một biên giới để chia hai lớp tốt nhất.

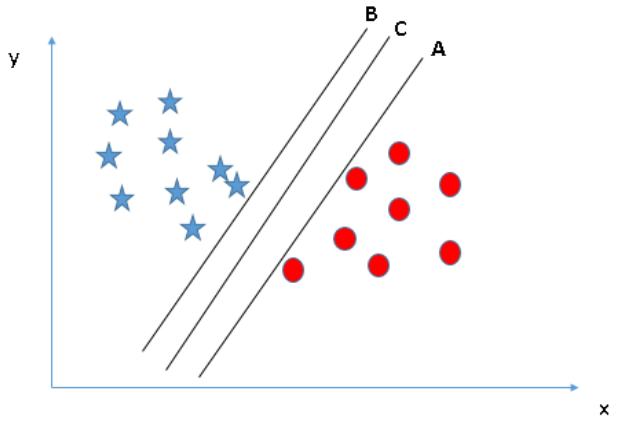
SVM làm việc như thế nào?
Ở trên, chúng ta đã thấy được việc chia hyper-plane. Bấy giờ làm thế nào chúng ta có thể xác định "Làm sao để vẽ-xác định đúng hyper-plane". Chúng ta sẽ theo các tiêu chí sau:

Identify the right hyper-plane (Scenario-1):
Ở đây, có 3 đường hyper-lane (A,B and C). Bây giờ đường nào là hyper-lane đúng cho nhóm ngôi sao và hình tròn.



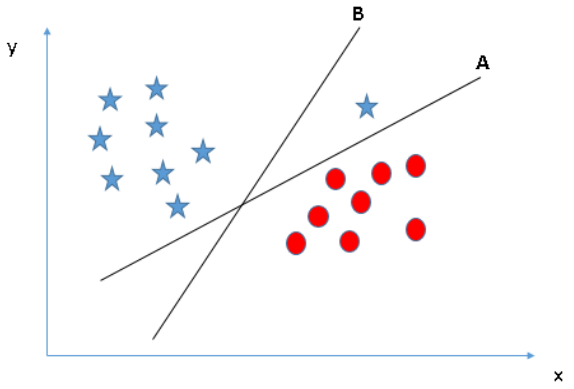

Quy tắc số một để chọn 1 hyper-lane, chọn một hyper-plane để phân chia hai lớp tốt nhất. Trong ví dụ này chính là đường B.

Identify the right hyper-plane (Scenario-2):
Ở đây chúng ta cũng có 3 đường hyper-plane (A,B và C), theo quy tắc số 1, chúng đều thỏa mãn.




Quy tắc thứ hai chính là xác định khoảng cách lớn nhất từ điểu gần nhất của một lớp nào đó đến đường hyper-plane. Khoảng cách này được gọi là "Margin", Hãy nhìn hình bên dưới, trong đấy có thể nhìn thấy khoảng cách margin lớn nhất đấy là đường C. Cần nhớ nếu chọn lầm hyper-lane có margin thấp hơn thì sau này khi dữ liệu tăng lên thì sẽ sinh ra nguy cơ cao về việc xác định nhầm lớp cho dữ liệu.

Identify the right hyper-plane (Scenario-3):
Sử dụng các nguyên tắc đã nêu trên để chọn ra hyper-plane cho trường hợp sau:

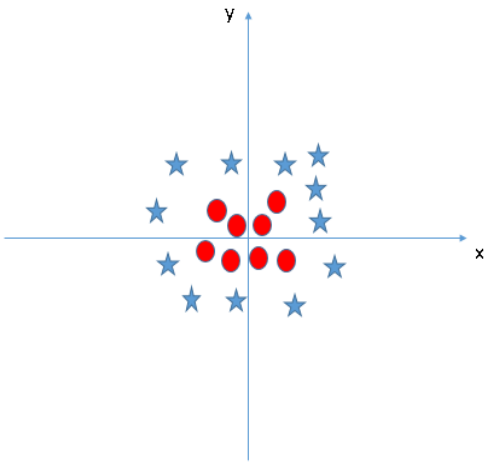



Có thể có một vài người sẽ chọn đường B bởi vì nó có margin cao hơn đường A, nhưng đấy sẽ không đúng bởi vì nguyên tắc đầu tiên sẽ là nguyên tắc số 1, chúng ta cần chọn hyper-plane để phân chia các lớp thành riêng biệt. Vì vậy đường A mới là lựa chọn chính xác.

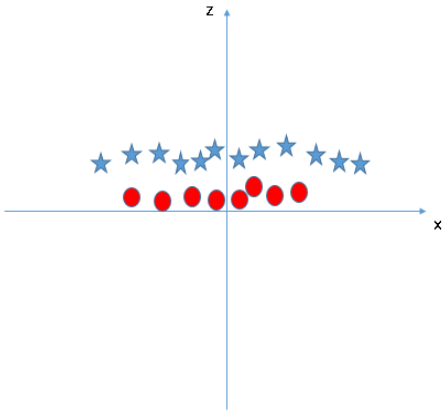
Can we classify two classes (Scenario-4)?
Tiếp the hãy xem hình bên dưới, không thể chia thành hai lớp riêng biệt với 1 đường thẳng, để tạo 1 phần chỉ có các ngôi sao và một vùng chỉ chứa các điểm tròn.


Ở đây sẽ chấp nhận, một ngôi sao ở bên ngoài cuối được xem như một ngôi sao phía ngoài hơn, SVM có tính năng cho phép bỏ qua các ngoại lệ và tìm ra hyper-plane có biên giới tối đa . Do đó có thể nói, SVM có khả năng mạnh trong việc chấp nhận ngoại lệ.

Find the hyper-plane to segregate to classes (Scenario-5)
Trong trường hợp dưới đây, không thể tìm ra 1 đường hyper-plane tương đối để chia các lớp, vậy làm thế nào để SVM phân tách dữ liệu thành hai lớp riêng biệt? Cho đến bây giờ chúng ta chỉ nhìn vào các đường tuyến tính hyper-plane.

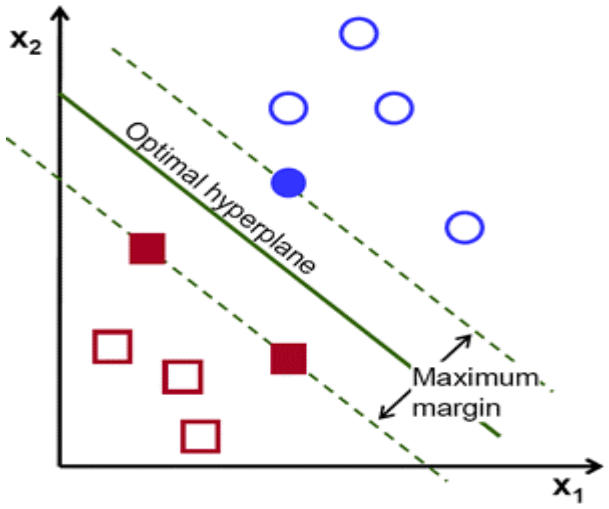



SVM có thể giải quyết vấn đề này, Khá đơn giản, nó sẽ được giải quyết bằng việc thêm một tính năng, Ở đây chúng ta sẽ thêm tính năng z = x^2+ y^2. Bây giờ dữ liệu sẽ được biến đổi theo trục x và z như sau




Trong sơ đồ trên, các điểm cần xem xét là: • Tất cả dữ liệu trên trục z sẽ là số dương vì nó là tổng bình phương x và y • Trên biểu đồ các điểm tròn đỏ xuất hiện gần trục x và y hơn vì thế z sẽ nhỏ hơn => nằm gần trục x hơn trong đồ thị (z,x) Trong SVM, rất dễ dàng để có một siêu phẳng tuyến tính (linear hyper-plane) để chia thành hai lớp, Nhưng một câu hỏi sẽ nảy sinh đấy là, chúng ta có cần phải thêm một tính năng phân chia này bằng tay hay không. Không, bởi vì SVM có một kỹ thuật được gọi là kernel trick ( kỹ thuật hạt nhân), đây là tính năng có không gian đầu vào có chiều sâu thấm và biến đổi nó thành không gian có chiều cao hơn, tức là nó không phân chia các vấn đề thành các vấn đề riêng biệt, các tính năng này được gọi là kernel. Nói một cách đơn giản nó thực hiện một số biết đổi dữ liệu phức tạp, sau đó tìm ra quá trình tách dữ liệu dựa trên các nhãn hoặc đầu ra mà chúng ra đã xác định trước.

**Margin trong SVM**



Margin là khoảng cách giữa siêu phẳng đến 2 điểm dữ liệu gần nhất tương ứng với các phân lớp. Trong ví dụ quả táo quả lê đặt trên mặt bán, margin chính là khoảng cách giữa cây que và hai quả táo và lê gần nó nhất. Điều quan trọng ở đây đó là phương pháp SVM luôn cố gắng cực đại hóa margin này, từ đó thu được một siêu phẳng tạo khoảng cách xa nhất so với 2 quả táo và lê. Nhờ vậy, SVM có thể giảm thiểu việc phân lớp sai (misclassification) đối với điểm dữ liệu mới đưa vào

# IV – PHÂN TÍCH ỨNG DỤNG

Ngôn ngữ lập trình: Python
Thư viện cài đặt: underthesea, regex, scikit-learn, pickle, fasttext, numpy
Môi trường thực thi: Jupyter notebook, Google colab

Cấu trúc chương trình cơ bản như sau: Tiền xử lí dữ liệu → Phân loại văn bản → Train data → Thử nghiệm mô hình → Đánh giá mô hình

## 4.1 Tiền xử lí và chuẩn hóa dữ liệu

Bước tiền xử lý dữ liệu là bước đầu tiên cần làm, nó cần được làm trước bước feature extractor. Việc tiền sử lý dữ liệu là quá trình chuẩn hóa dữ liệu và loại bỏ các thành phần không có ý nghĩa cho việc phân loại văn bản.

### 4.1.1 Xóa HTML code trong dữ liệu

```
def remove_html(txt):
    return re.sub(r'<[^>]*>', '', txt)

txt = "<p class=\"par\">This is an example</p>"
remove_html(txt)
```

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. Việc xóa các HTML code này cũng khá đơn giản, chúng ta có thể sử dụng regex trong Python để xóa chúng đi.

### 4.1.2 Chuẩn hóa Unicode tiếng Việt

Đôi lúc người dùng sử dụng bộ mã khác nhau khi gõ tiếng Việt. Cụ thể đa số chúng ta đang dùng Unikey với bộ mã Unicode (dựng sẵn). Nhưng ở đâu đó, bộ mã Unicode tổ hợp vẫn được sử dụng. Do đó, đối với các nguồn dữ liệu thu thập trên internet thường bị lẫn cả 2 cách gõ này. Chúng ta sẽ thay thế cách gõ Unicode tổ hợp bằng cách gõ của Unicode dựng sẵn tránh việc xuất hiện lỗi khi train.

```
import regex as re

uniChars = "àáảãạ…"
unsignChars = "aaaaaaaaaaa…"
def loaddicchar():
    dic = {}
    char1252 = 'à|á|ả|ã|ạ|… '.split(
        '|')
    charutf8 = "à|á|ả|ã|ạ|… ".split(
        '|')
    for i in range(len(char1252)):
        dic[char1252[i]] = charutf8[i]
    return dic

dicchar = loaddicchar()

def covert_unicode(txt):
    return re.sub(
        r'à|á|ả|ã|ạ|…',
        lambda x: dicchar[x.group()], txt)
```

### 4.1.3 Chuẩn hóa kiểu gõ dấu tiếng Việt

Đưa về kiểu gõ dấu cũ (ex: oà, uý) đối với các từ dùng kiểu gõ mới (ex: oà, uý)

```
def chuan_hoa_dau_tu_tieng_viet(word):
    if not is_valid_vietnam_word(word):
        return word

    chars = list(word)
    dau_cau = 0
    nguyen_am_index = []
    qu_or_gi = False
    for index, char in enumerate(chars):
        x, y = nguyen_am_to_ids.get(char, (-1, -1))
        if x == -1:
            continue
        elif x == 9:  # check qu
            if index != 0 and chars[index - 1] == 'q':
                chars[index] = 'u'
                qu_or_gi = True
        elif x == 5:  # check gi
            if index != 0 and chars[index - 1] == 'g':
```

```
                chars[index] = 'i'
                qu_or_gi = True
        if y != 0:
            dau_cau = y
            chars[index] = bang_nguyen_am[x][0]
        if not qu_or_gi or index != 1:
            nguyen_am_index.append(index)
    if len(nguyen_am_index) < 2:
        if qu_or_gi:
            if len(chars) == 2:
                x, y = nguyen_am_to_ids.get(chars[1])
                chars[1] = bang_nguyen_am[x][dau_cau]
            else:
                x, y = nguyen_am_to_ids.get(chars[2], (-1, -1))
                if x != -1:
                    chars[2] = bang_nguyen_am[x][dau_cau]
                else:
                    chars[1] = bang_nguyen_am[5][dau_cau] if chars[1] == 'i'
 else bang_nguyen_am[9][dau_cau]
            return ''.join(chars)
        return word

    for index in nguyen_am_index:
        x, y = nguyen_am_to_ids[chars[index]]
        if x == 4 or x == 8:  # ê, ơ
            chars[index] = bang_nguyen_am[x][dau_cau]
            return ''.join(chars)

    if len(nguyen_am_index) == 2:
        if nguyen_am_index[-1] == len(chars) - 1:
            x, y = nguyen_am_to_ids[chars[nguyen_am_index[0]]]
            chars[nguyen_am_index[0]] = bang_nguyen_am[x][dau_cau]
        else:
            x, y = nguyen_am_to_ids[chars[nguyen_am_index[1]]]
            chars[nguyen_am_index[1]] = bang_nguyen_am[x][dau_cau]
    else:
        x, y = nguyen_am_to_ids[chars[nguyen_am_index[1]]]
        chars[nguyen_am_index[1]] = bang_nguyen_am[x][dau_cau]
    return ''.join(chars)


def is_valid_vietnam_word(word):
    chars = list(word)
    nguyen_am_index = -1
    for index, char in enumerate(chars):
        x, y = nguyen_am_to_ids.get(char, (-1, -1))
```

```
        if x != -1:
            if nguyen_am_index == -1:
                nguyen_am_index = index
            else:
                if index - nguyen_am_index != 1:
                    return False
                nguyen_am_index = index
    return True


def chuan_hoa_dau_cau_tieng_viet(sentence):
    """
        Chuyển câu tiếng việt về chuẩn gõ dấu kiểu cũ.
        :param sentence:
        :return:
        """
    sentence = sentence.lower()
    words = sentence.split()
    for index, word in enumerate(words):
        cw = re.sub(r'(^\p{P}*)([p{L}.]*\p{L}+)(\p{P}*$)', r'\1/\2/\3', word
).split('/')
        # print(cw)
        if len(cw) == 3:
            cw[1] = chuan_hoa_dau_tu_tieng_viet(cw[1])
        words[index] = ''.join(cw)
    return ' '.join(words)

if __name__ == '__main__':
    print(chuan_hoa_dau_cau_tieng_viet('anh hoà, đang làm.. gì'))
```

Script python như trên cung cấp cho chúng ta khả năng chuẩn hóa kiểu gõ về kiểu cũ, đồng thời không làm thay đổi cấu trúc dữ liệu gốc (giữ nguyên hoa thường, dấu ngắt câu,…).

### 4.1.4 Tách từ tiếng Việt

Ở phần tiếp theo 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 (Ex: Học sinh học sinh học ⇒ Học_sinh học sinh_học).

Bài toán này là một bài toán cơ sở trong NLP – bài toán tách từ (word tokenize). 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.

```
>>> from underthesea import word_tokenize
>>> sentence = 'Chàng trai 9X Quảng Trị khởi nghiệp từ nấm sò'

>>> word_tokenize(sentence)
['Chàng trai', '9X', 'Quảng Trị', 'khởi nghiệp', 'từ', 'nấm', 'sò']

>>> word_tokenize(sentence, format="text")
'Chàng_trai 9X Quảng_Trị khởi_nghiệp từ nấm sò'
```

### 4.1.5 Đưa về viết thường (lowercase) và xóa các kí tự không cần thiết

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.

Ngoài ra 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

```
def text_preprocess(document):
    # xóa html code
    document = remove_html(document)
    # chuẩn hóa unicode
    document = convert_unicode(document)
    # chuẩn hóa cách gõ dấu tiếng Việt
    document = chuan_hoa_dau_cau_tieng_viet(document)
    # tách từ
    document = word_tokenize(document, format="text")
    # đưa về lower
    document = document.lower()
    # xóa các ký tự không cần thiết
    document = re.sub(r'[^\s\wáà…',' ',document)
    # xóa khoảng trắng thừa
    document = re.sub(r'\s+', ' ', document).strip()
    return document
```

### 4.1.6 Loại bỏ các stopword tiếng Việt

Bây giờ chúng ta sẽ duyệt qua từng bản ghi dữ liệu và xóa tất cả các từ trong dữ liệu mà có trong danh sách stopword.

```
# Danh sách stopword
stopword = set()

def remove_stopwords(line):
    words = []
    for word in line.strip().split():
        if word not in stopword:
            words.append(word)
    return ' '.join(words)
```

Sau khi thực hiện tiền xử lý dữ liệu xong, giờ chúng ta bắt tay vào xây dựng các mô hình học máy cho bài toán phân loại văn bản trên dữ liệu đã tiền xử lý.

## 4.2 Xây dựng mô hình phân loại văn bản

### 4.2.1 Xây dựng tập train/test

Sử dụng thư viện sklearn trong Python giúp mình tách dữ liệu làm 2 tập train/test riêng biệt:

- Đọc dữ liệu từ file và tách làm 2 list text (dữ liệu) và label (nhãn). Dữ liệu text[i] sẽ có nhãn là label[i].
- Chia làm 2 tập train (X_train, y_train) và test (X_test, y_test) theo tỉ lệ 80% train, 20% test.
- Lưu train/test data ra file để sử dụng cho việc train với thư viện Fasttext.
- Đưa label về dạng vector để tiện cho tính toán sử dụng LabelEncoder.

### 4.2.2 Phân loại văn bản với Logistic Regression

Logistic Regression sẽ giúp chúng ta phân loại được dùng để gán các đối tượng cho 1 tập hợp giá trị rời rạc.

```
from sklearn.linear_model import LogisticRegression
```

```
start_time = time.time()
text_clf = Pipeline([('vect', CountVectorizer(ngram_range=(1,1),
                                              max_df=0.8,
                                              max_features=None)),
                     ('tfidf', TfidfTransformer()),
                     ('clf', LogisticRegression(solver='lbfgs',
                                                multi_class='auto',
                                                max_iter=10000))
                    ])
text_clf = text_clf.fit(X_train, y_train)

train_time = time.time() - start_time
print('Done training Linear Classifier in', train_time, 'seconds.')

# Save model
pickle.dump(text_clf, open(os.path.join(MODEL_PATH, "linear_classifier.pkl")
, 'wb'))
```

### 4.3 Đánh giá mô hình phân loại văn bản

- Độ chính xác:

```
import numpy as np

# Linear Classifier
model = pickle.load(open(os.path.join(MODEL_PATH,"linear_classifier.pkl"), '
rb'))
y_pred = model.predict(X_test)
print('Linear Classifier, Accuracy =', np.mean(y_pred == y_test))
```

Và đây là kết quả sau khi chạy đoạn code đánh giá trên:

```
Linear Classifier, Accuracy = 0.883140531276778
```

- Chúng ta cũng có thể xem kết quả của từng nhãn:

```
                    precision    recall  f1-score   support

 __label__công_nghệ       0.91      0.92      0.92       532
   __label__du_lịch       0.79      0.88      0.83       551
  __label__giáo_dục       0.82      0.88      0.85       528
  __label__giải_trí       0.59      0.74      0.66       487
__label__kinh_doanh       0.78      0.84      0.81       498
```

| | | | | |
|---|---|---|---|---|
| __label__nhịp_sống | 0.85 | 0.49 | 0.62 | 497 |
| __label__phim_ảnh | 0.90 | 0.76 | 0.82 | 525 |
| __label__pháp_luật | 0.90 | 0.92 | 0.91 | 543 |
| __label__sống_trẻ | 0.62 | 0.64 | 0.63 | 510 |
| __label__sức_khỏe | 0.79 | 0.88 | 0.83 | 496 |
| __label__thế_giới | 0.91 | 0.83 | 0.87 | 549 |
| __label__thể_thao | 0.95 | 0.95 | 0.95 | 508 |
| __label__thời_sự | 0.82 | 0.77 | 0.79 | 496 |
| __label__thời_trang | 0.86 | 0.77 | 0.81 | 521 |
| __label__xe_360 | 0.97 | 0.94 | 0.96 | 502 |
| __label__xuất_bản | 0.87 | 0.93 | 0.90 | 519 |
| __label__âm_nhạc | 0.83 | 0.87 | 0.85 | 554 |
| __label__ẩm_thực | 0.90 | 0.94 | 0.92 | 520 |
| micro avg | 0.83 | 0.83 | 0.83 | 9336 |
| macro avg | 0.84 | 0.83 | 0.83 | 9336 |
| weighted avg | 0.84 | 0.83 | 0.83 | 9336 |

## 4.4 Nhận xét & đánh giá

- Một số nhãn cho độ chính xác phân loại thấp (giải trí, nhịp sống, sống trẻ). Nguyên nhân có thể do các chuyên mục này không thực sự quá rõ ràng, nổi bật so với các chuyên mục khác nên mô hình dễ bị sai hơn.

- Có thể train lại với pretrain word2vec hoặc sử dụng nhiều dữ liệu hơn để nó có thể cho kết quả tốt hơn.

- Tăng số lượng dữ liệu huấn luyện cho các mô hình sẽ giúp mô hình trên đây học tốt hơn.

# TÀI LIỆU THAM KHẢO

https://viblo.asia/p/phan-loai-van-ban-tieng-viet-tu-dong-phan-1-yMnKM3bal7P

https://nguyenvanhieu.vn/phan-loai-van-ban-tieng-viet/

https://tek4.vn/khoa-hoc/machine-learning-co-ban/training-set-va-testing-set

https://whitehat.vn/threads/thuat-toan-phan-loai-naive-bayes-va-ung-dung.13775/

https://viblo.asia/p/gioi-thieu-ve-support-vector-machine-svm-6J3ZgPVElmB