TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN - ĐHQGHN
KHOA TOÁN - CƠ - TIN HỌC

—————o0o—————

BÁO CÁO CUỐI KỲ

KỸ THUẬT LỌC TIN NHẮN RÁC SMS

Môn học: Khai phá dữ liệu

Giảng viên hướng dẫn: **PGS.TS. Lê Hoàng Sơn**

Nhóm 11: **Nguyễn Chí Dũng - 20002040**

Đinh Tiến Dũng - 20002039

Nguyễn Hoàng Giang - 20002048

Nguyễn Thị Khánh Linh - 20002066

Lớp: **K65A5 Khoa học dữ liệu**

HÀ NỘI, 5/2023

Lời nói đầu

Trong những năm gần đây, sự bùng nổ của mạng di động tại Việt Nam đã kích thích việc sử dụng dịch vụ tin nhắn ngắn (SMS). Thư rác qua SMS đã tăng lên vì chi phí cho người gửi thư rác thấp và tỷ lệ phản hồi cao hơn email. Spam SMS đang là một vấn đề nghiêm trọng ở Việt Nam do các gói SMS trả trước không giới hạn và rất rẻ.

Theo một báo cáo vào năm 2015 của BKAV, một hãng bảo mật thông tin, có 13,9 triệu tin nhắn rác được gửi đến người dùng điện thoại di động mỗi ngày ở Việt Nam và cứ hai người thì có một người nhận được tin nhắn rác. Trong khi đó, có ít nhất 120 triệu người dùng di động tích cực trên thị trường theo một báo cáo của Bộ Thông tin và Truyền thông, được công bố vào năm 2015.

Spam SMS gây ra nhiều vấn đề cho người dùng di động. Họ có thể bị tổn thất tài chính từ những tin nhắn này bằng cách phản ứng lại chúng. Người dùng có thể vô tình gọi đến các số giá cước cao cấp hoặc đăng ký các dịch vụ đắt tiền bằng cách trả lời các tin nhắn này. Hơn nữa, họ có thể gặp phải một số rủi ro khi truy cập các trang web có hại hoặc tải xuống phần mềm độc hại. Các nhà khai thác mạng di động cũng bị thiệt hại về tài chính vì họ có thể mất người dùng hoặc chi nhiều hơn cho việc ngăn chặn thư rác.

Chúng ta đã và đang sống trong thời đại 4.0, khi công nghệ thông tin phát triển mạnh mẽ và số lượng người dùng điện thoại thông minh ngày càng tăng. Sau một ngày dài làm việc, chúng ta không hề muốn những tin nhắn rác gây ra những phiền toái. Chính vì lẽ đó, việc lọc tin nhắn rác là rất quan trọng, giúp bảo vệ thông tin cá nhân, tiết kiệm thời gian và tăng hiệu quả sử dụng điện thoại, tối ưu hóa dịch vụ của nhà cung cấp, đảm bảo an toàn và bảo mật và cải thiện trải nghiệm người dùng.

Vậy, một quy trình lọc tin nhắn rác được xây dựng và hoạt động ra sao? Câu hỏi này sẽ được lý giải trong đề tài của chúng em: "Kỹ thuật lọc tin nhắn rác SMS".

Nội dung đề tài gồm các chương:

Chương 1. Cơ sở lý thuyết

Chương 2. Phân tích xử lý bộ dữ liệu

Chương 3. Xây dựng và thực thi chương trình

Chương 4. Kết luận và hạn chế - giải pháp

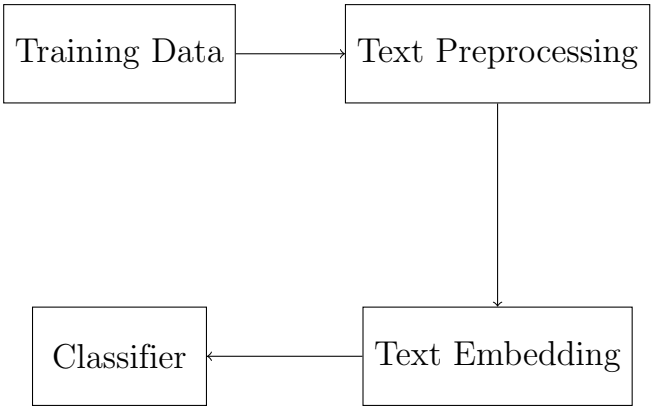
Chúng em kính mong các thầy/cô góp ý thêm để báo cáo của chúng em đạt gần với thực tế hơn. Chúng em xin chân thành cảm ơn.

Lý do lựa chọn đề tài

Ngày nay, các tin nhắn rác mà người dùng nhận được thường có nội dung liên quan đến quảng cáo sản phẩm, dịch vụ. Tuy nhiên nó cũng có thể bị kẻ xấu lợi dụng nhằm lừa đảo và chiếm đoạt thông tin. Do đó, việc nhận diện và phân loại được các tin nhắn này là rất quan trọng để giảm thiểu sự phiền toái và nguy hiểm cho người dùng.

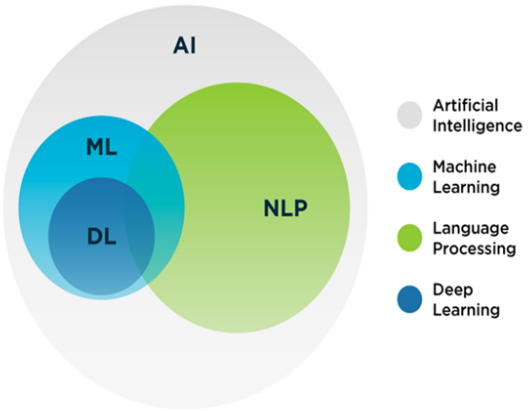
Một trong số những phương pháp phát hiện tin nhắn rác là áp dụng học máy bằng cách lấy mẫu các tin nhắn đã được nhận dạng là tin nhắn rác hoặc thường. Sau đó, huấn luyện một mô hình xử lý ngôn ngữ tự nhiên (NLP) để nhận diện tin nhắn rác. Mô hình này sẽ phân tích đặc trưng của các tin nhắn và phân loại chúng dựa trên các tiêu chí đã được định nghĩa.

NLP (Natural Language Processing) là một lĩnh vực của trí tuệ nhân tạo liên quan đến việc xử lý và phân tích ngôn ngữ tự nhiên của con người. Quá trình này giúp máy tính có thể hiểu được ngữ nghĩa, cú pháp của ngôn ngữ tự nhiên. Một dự án NLP trải qua 4 giai đoạn:



Trong đề tài này, NLP được sử dụng để phân tích và xác định các đặc điểm của tin nhắn SMS, từ đó đưa ra quyết định xem tin nhắn đó có phải là spam hay không.

Chúng ta sẽ tìm hiểu về các kỹ thuật NLP để xử lý và phân loại tin nhắn, từ tiền xử lý dữ liệu đến các phương pháp trích xuất đặc trưng và xây dựng mô hình học máy để nhận diện Spam SMS. Sau đó, thông qua thực nghiệm để đánh giá độ chính xác và hiệu suất của các mô hình được đề xuất.



Danh mục viết tắt

STT	Từ viết tắt	Tên đầy đủ	Giải thích
1	ANN	Artificial Neural Network	Mạng nơron
2	BBN	Bayesian Belief Network	Mô hình xác suất dạng đồ thị.
3	SVM	Support Vector Machine	Mô hình học máy có giám sát được sử dụng cho các bài toán phân loại và hồi quy
4	MNB	Multinomial Naive Bayes	Thuật toán phân lớp được mô hình hoá dựa trên định lý Bayes trong xác suất thống kê
5	MLP	Multi Layer Perceptron	Mạng nơ-ron được sử dụng cho bài toán phân loại, hồi quy
6	EDA	Exploratory Data Analysis	Phân tích khai phá dữ liệu

Danh mục các hình ảnh

Mục lục

Chương 1

Cơ sở lý thuyết

1.1 Khai phá dữ liệu

1.1.1 Khai phá dữ liệu là gì?

Khai phá dữ liệu (Data Mining) là quá trình sắp xếp, phân loại một tập hợp các dữ liệu lớn để xác định các mẫu và thành lập một mối quan hệ nhằm giải quyết nhiều vấn đề thông qua việc phân tích dữ liệu. Quá trình để khai phá dữ liệu là một quá trình phức tạp đòi hỏi dữ liệu cần phải chuyên sâu và yêu cầu nhiều kỹ năng tính toán khác nhau. Hơn nữa, khai phá dữ liệu không chỉ giới hạn trong việc trích xuất các dữ liệu mà còn sử dụng để làm sạch, chuyển đổi, tích hợp dữ liệu và phân tích các mẫu.

1.1.2 Quy trình khai phá dữ liệu

Quy trình khai phá dữ liệu bao gồm 7 bước như sau, cùng với đó quy trình khai phá dữ liệu được trình bày dưới dạng sơ đồ hình vẽ thông qua hình 1.1:

- *Bước 1: Làm sạch dữ liệu.*
  Đây là bước đầu tiên trong quy trình khai phá dữ liệu.Bước này được đánh giá là khá quan trọng vì những dữ liệu bẩn nếu được sử dụng trựctiếp trong khai phá dữ liệu có thể sẽ gây ra kết quả nhầm lẫn, dự báo và tạo ra các kết quả không được chính xác.

- *Bước 2: Tích hợp dữ liệu.*
  Ở bước này, dữ liệu của chúng ta được cải thiện về độ chính xác cũng như tốc độ của quá trình khai phá dữ liệu.

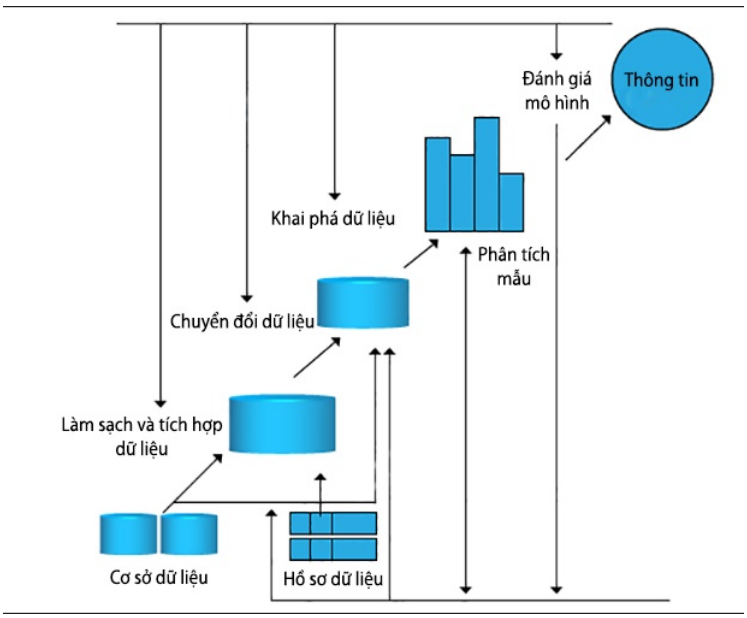
- *Bước 3: Làm giảm dữ liệu.*
  Mục đích ở bước này là giúp kích thước của dữ liệu có khối lượng nhỏ hơn nhưng nó vẫn đảm bảo và vẫn duy trì tính toàn vẹn.

- *Bước 4: Chuyển đổi dữ liệu.*
  Trong bước này, dữ liệu được chuyển thành một dạng phù hợp với quy trình khai phá dữ liệu. Dữ liệu được hợp nhất để quy trình khai phá dữ liệu có thể hiệu quả hơn và các mẫu dễ hiểu hơn.

- *Bước 5: Khai thác dữ liệu.*
  Ở bước này, chúng ta đi khai thác dữ liệu là để xác định các mẫu và một lượng lớn dữ liệu từ những suy luận.

- *Bước 6: Đánh giá mẫu.*
  Bước này bao gồm việc xác định các mẫu đại diện cho nhiều kiến thức dựa trên những thước đo, cho biết những kiến thức nào là cần thiết, kiến thức nào là dư thừa và sẽ bị loại bỏ. Các phương pháp trực quan hóa và tóm tắt dữ liệu được sử dụng để người dùng có thể hiểu được bộ dữ liệu của mình.

- *Bước 7: Trình bày thông tin.*
  Dữ liệu sẽ được diễn giải lại dưới các báo cáo, hoặc các báo cáo dạng bảng, sau đó gửi cho bên bộ phân xử lý thông tin này.



Hình 1.1 Quy trình khai phá dữ liệu

1.1.3 Phương pháp khai phá dữ liệu

Phân lớp (Classification): Phương pháp sử dụng để dự báo dữ liệu thông qua bộ dữ liệu huấn luyện, phân loại đối tượng. Tôi sẽ sử dụng phương pháp khai phá dữ liệu này trong bài để dự báo số liệu.

Hồi quy (Regression): Mục đích chính của phương pháp hồi quy này là dùng để khám phá và ánh xạ dữ liệu.

Phân cụm (Clustering): Phương pháp phân cụm giúp việc mô tả dữ liệu trở nên dễ dàng hơn bằng các xác định tập hợp hữu hạn các cụm với nhau.

Tổng hợp (Summarization): Phương pháp này cho phép người làm tìm kiếm một mô tả nhỏ gọn.

Mô hình ràng buộc (Dependency modeling): Người làm sẽ tìm được mô hình cục bộ mô tả các phụ thuộc dựa vào phương pháp mô hình ràng buộc.

Dò tìm biến đổi và độ lệch (Change and Deviation Dectection): Mục đích của phương pháp này là để tìm ra những thay đổi quan trọng.

1.1.4 Ứng dụng của khai phá dữ liệu trong đời sống

Khai phá dữ liệu được ứng dụng rất nhiều trong đời sống xã hội, tiêu biểu ở một số những lĩnh vực như sau: Phân tích thị trường – chứng khoán, phát hiện gian lận, quản trị rủi ro doanh nghiệp, bán lẻ, trí tuệ nhân tạo, thương mại điện tử, phòng chống tội phạm và rất nhiều các lĩnh vực khác.

1.2 Mô hình phân lớp dữ liệu

1.2.1 Khái niệm về phân lớp dữ liệu

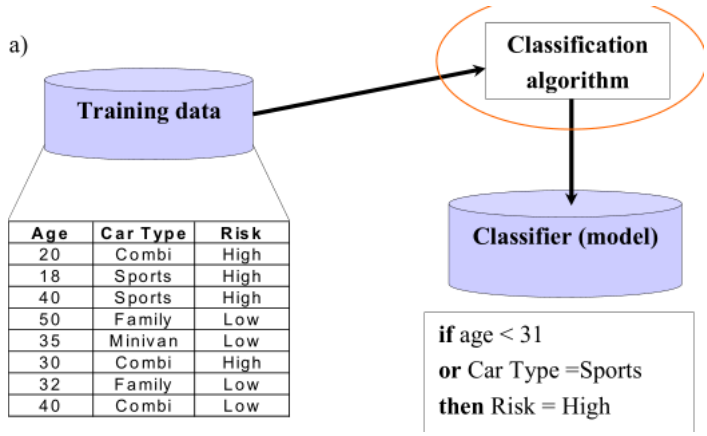
Phân lớp dữ liệu chính là một trong những hướng nghiên cứu chính của khai phá dữ liệu. Phân lớp dữ liệu là quá trình phân các đối tượng vào một hay nhiều lớp đã cho trước nhờ vào một mô hình phân lớp. Mô hình này được xây dựng dựa trên một tập dữ liệu đã được gán nhãn trước đó. Quá trình gán nhãn cho đối tượng dữ liệu chính là quá trình phân lớp dữ liệu. Phân lớp và dự đoán là một trong hai dạng của phân tích dữ liệu nhằm rút ra một mô hình mô tả các lớp dữ liệu quan trọng hoặc dự đoán xu hướng của dữ liệu trong tương lai.

1.2.2 Quy trình phân lớp dữ liệu

Quy trình phân lớp dữ liệu gồm hai bước như sau: Bước thứ nhất là xây dựng mô hình phân lớp (Learning). Bước thứ hai là ước lượng độ chính xác của mô hình cùng với đó là phân lớp dữ liệu mới (Classification).

Xây dựng mô hình phân lớp (Learning)

Ở bước xây dựng mô hình phân lớp, một mô hình được xây dựng để mô tả tập hợp các dữ liệu. Ban đầu, bộ dữ liệu có cấu trúc, nó được mô tả bằng các thuộc tính và được tạo ra từ các bộ giá trị có thuộc tính đó. Một bộ giá trị sẽ được coi là một phần tử của dữ liệu, ngoài ra còn thể thể là các mẫu, đối tượng,... Trong tập dữ liệu này, mỗi phần tử dữ liệu thuộc về một lớp định trước, lớp ở đây có nghĩa là các giá trị của một thuộc tính được chọn làm các thuộc tính gán nhãn hay còn gọi là các thuộc tính phân lớp. Sau đó, sử dụng các quy tắc phân lớp dưới dạng if–then, cây quyết định (Decision Tree), hồi quy Logistic (Logistic Regression), mạng neural (Neural Network),... Ở bước xây dựng mô hình phân lớp có thể được mô tả lại ở hình 1.2 dưới đây:



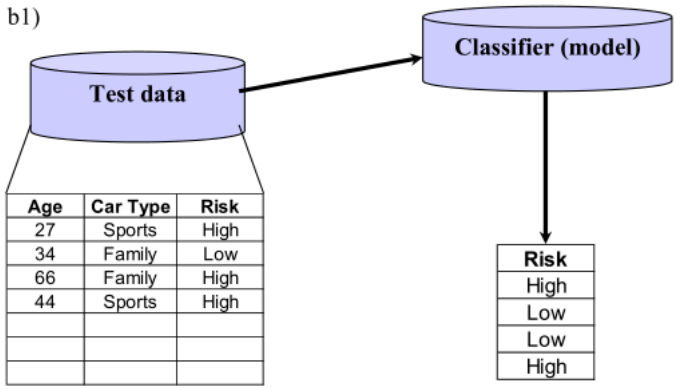
Age	Car Type	Risk
20	Combi	High
18	Sports	High
40	Sports	High
50	Family	Low
35	Minivan	Low
30	Combi	High
32	Family	Low
40	Combi	Low

Hình 1.2 Quá trình phân lớp dữ liệu – Xây dựng mô hình phân lớp

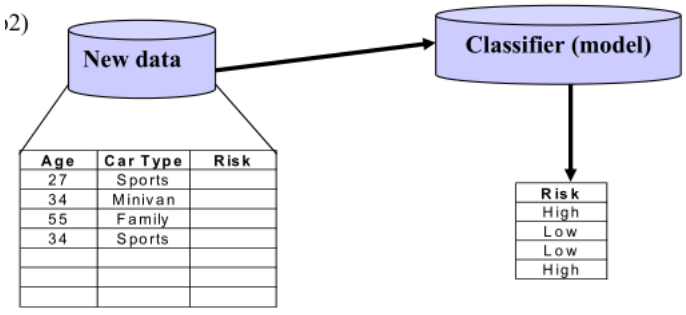
Ước lượng độ chính xác của mô hình và phân lớp dữ liệu mới (Classification)

Ở bước này, chúng ta sẽ dùng mô hình đã xây dựng ở bước bước để phân lớp dữ liệu mới. Đầu tiên, độ chính xác mang tính chất dự đoán của mô hình phân lớp vừa tạo ra được ước lượng. Độ chính xác của mô hình trên tập dữ liệu kiểm tra là tỉ lệ phần trăm của các mẫu trong tập dữ liệu kiểm tra được mô

hình phân lớp đúng (so với thực tế). Nếu độ chính xác của mô hình được ước lượng dựa trên tập dữ liệu đã được huấn luyện thì kết quả dự báo thu được rất khả quan. Chúng ta cần phải có một bộ dữ liệu dự báo độc lập với bộ dữ liệu đã được huấn luyện. Nếu độ chính xác của mô hình là có thể chấp nhận thì mô hình được sử dụng để phân lớp những dữ liệu trong tương lai hoặc dữ liệu mà giá trị thuộc tính phân lớp là chưa biết. Ở bước ước lượng độ chính xác của mô hình và phân lớp dữ liệu mới được mô tả qua hình 1.3 và hình 1.4 như sau:



Hình 1.3 Quá trình phân lớp dữ liệu – Ước lượng độ chính xác của mô hình



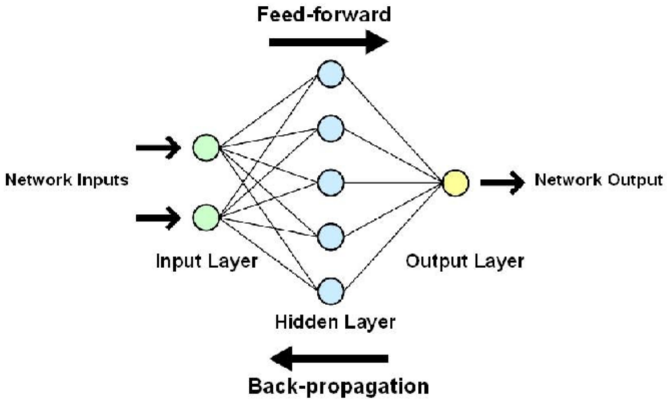
Hình 1.4 Quá trình phân lớp dữ liệu – Phân lớp dữ liệu mới

1.2.3 Artificial Neural Network

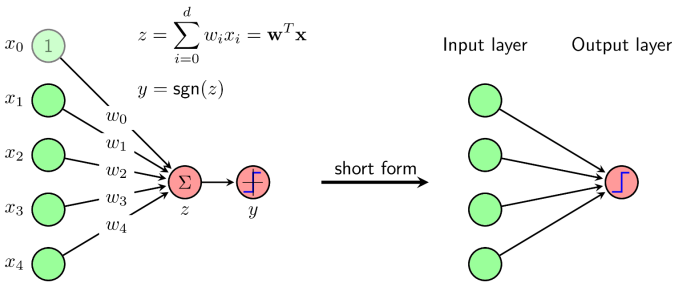
1. Khái niệm

Artificial Neural Network (Mạng nơron) là một mô hình được xây dựng dựa trên cách hoạt động bộ não con người. Gồm 3 thành phần chính: input layer và output layer chỉ gồm 1 layer, hidden layer có thể có một hay nhiều layer tùy vào bài toán cụ thể. ANN hoạt động theo hướng mô tả lại cách hoạt động của hệ thần kinh với các nơ-ron được kết nối với nhau.

Trong ANN, trừ input layer thì tất cả các node thuộc các layer khác đều full-connected với các node thuộc layer trước nó. Mỗi node thuộc hidden layer nhận ma trận đầu vào từ layer trước và kết hợp với trọng số để ra được kết quả.



Hình 1.6 Artificial Neural Network (1)



Hình 1.5 Artificial Neural Network (2)

Weights và Biases

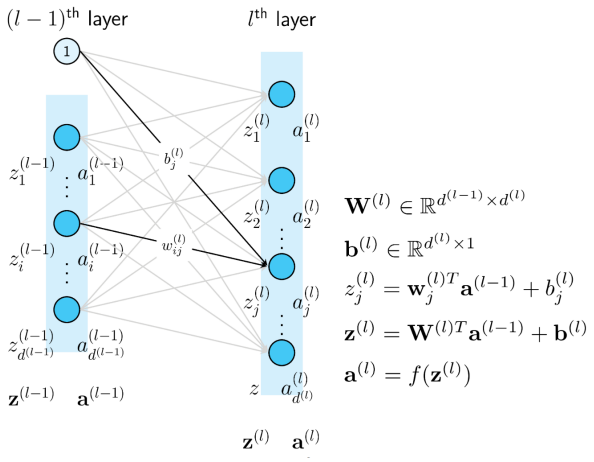
• Ta gọi tham số w trong Perceptron 01 layer là trọng số.

• Phần tử $w_{ij}^{(l)}$ là tham số trong tổ hợp kết nối từ node thứ i của layer thứ $(l1)$ tới node j của layer thứ (l).

• Coi input là layer thứ 0, các trọng số $w_{ij}^{(l)}$ tạo thành ma trận $\mathbf{W^{(l)}} \in \mathrm{R}^{d^{l-1} \cdot d^l}$

• Đây là ma trận trọng số $\mathbf{W^{(l)}}$ kết nối giữa layer thứ $(l-1)$ và layer thứ (l).

• Có L ma trận trọng số cho một MLP có L layers: $\mathbf{W^{(l)}}$, $l = 1, 2, ..., L$.



Hình 1.7 Artificial Neural Network (3)

Activation function

Một số hàm activation:

$$sigmoid(z) = \frac{1}{[1 + exp(-z)]}$$

$$tanh(z) = \frac{[1 + exp(z)] - [1 + exp(-z)]}{[1 + exp(z)] + [1 + exp(-z)]}$$

$$RELU : f(z) = \max(0, z)$$

• Nhược điểm của $sigmoid(z)$ và $tanh(z)$: Với $z \gg 1$ thì$f'(z) \approx 0$.

• ReLU (Rectified Linear Unit): Với $z > 0, f'(z) = 1$ và $z < 0, f'(z) = 0$.

2. Thuật toán Backpropagation

Phương pháp phổ biến nhất để tối ưu MLP vẫn là Gradient Descent (GD). Để áp dụng GD, chúng ta cần tính được gradient của hàm mất mát theo từng ma trận trọng số $\mathbf{W^{(l)}}$và vector bias $\mathbf{b^l}$. Trước hết, chúng ta cần tính *predicted output* $\widehat{y}$ với một input $\mathbf{x}$:

$$\begin{aligned}
\mathbf{a}^{(0)} &= \mathbf{x} \\
z_i^{(l)} &= \mathbf{w}_i^{(l)T}\mathbf{a}^{(l-1)} + b_i^{(l)} \\
\mathbf{z}^{(l)} &= \mathbf{W}^{(l)T}\mathbf{a}^{(l-1)} + \mathbf{b}^{(l)}, \;\; l = 1, 2, \ldots, L \\
\mathbf{a}^{(l)} &= f(\mathbf{z}^{(l)}), \;\; l = 1, 2, \ldots, L \\
\hat{\mathbf{y}} &= \mathbf{a}^{(L)}
\end{aligned}$$

Bước này được gọi là *feedforward* vì cách tính toán được thực hiện từ đầu đến cuối của network. MLP cũng được gọi

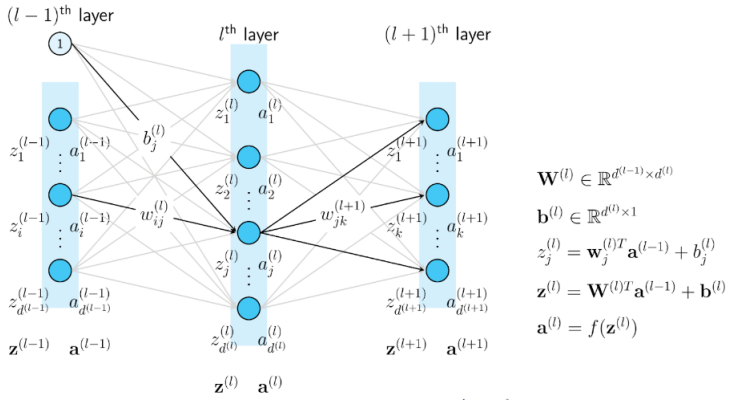
Giả sử $\mathbf{J}(\mathbf{W}, \mathbf{b}, \mathbf{X}, \mathbf{Y})$ là một hàm mất mát của bài toán, trong đó $\mathbf{W}, \mathbf{b}$ là tập hợp tất cả các ma trận trọng số giữa các layers và biases của mỗi layer. $\mathbf{X}, \mathbf{Y}$ là cặp dữ liệu huấn luyện với mỗi cột tương ứng với một điểm dữ liệu. Để có thể áp dụng các gradient-based methods (mà Gradient Descent là một ví dụ), chúng ta cần tính được:

$$\frac{\partial J}{\partial \mathbf{W}^{(l)}}; \frac{\partial J}{\partial \mathbf{b}^{(l)}}, \;\; l = 1, 2, \ldots, L$$

Tại mỗi bước lặp GD, ta cập nhật lại các trọng số $\mathbf{W^{(l)}}$ và bias $\mathbf{b^l}$ theo công thức:

$$w_{ij}^{(l)} = w_{ij}^{(l)} - \alpha \frac{\partial \boldsymbol{J}(\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{X}, \boldsymbol{Y})}{\partial w_{ij}^{(l)}} = w_{ij}^{(l)} - \alpha \frac{\partial \boldsymbol{J}(\boldsymbol{W}, \boldsymbol{b})}{\partial w_{ij}^{(l)}}$$

$$b_i^{(l)} = b_i^{(l)} - \alpha \frac{\partial \boldsymbol{J}(\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{X}, \boldsymbol{Y})}{\partial b_i^{(l)}} = b_i^{(l)} - \alpha \frac{\partial \boldsymbol{J}(\boldsymbol{W}, \boldsymbol{b})}{\partial b_i^{(l)}}$$



Hình 1.8 Artificial Neural Network (4)

Đạo hàm theo từng hệ số tội, b w

1. Bước feedforward: Với 1 giá trị đầu vào $\mathbf{a}$, tính giá trị đầu ra của network, trong quá trình tính toán, lưu lại các *activation* a^l tại mỗi layer.

2. Với mỗi unit j ở output layer, tính

$$e_j^{(L)} = \frac{\partial J}{\partial z_j^{(L)}}$$

3. Từ đó suy ra:

$$\frac{\partial J}{\partial w_{ij}^{(L)}} = a_i^{(L-1)} e_j^{(L)}$$
$$\frac{\partial J}{\partial b_j^{(L)}} = e_j^{(L)}$$

4. Với $l = L-1, L-2, ..., 1$, tính:

$$e_j^{(l)} = \left(\mathbf{w}_{j:}^{(l+1)} \mathbf{e}^{(l+1)} \right) f'(z_j^{(l)})$$

5. Cập nhật đạo hàm cho từng hệ số:

$$\frac{\partial J}{\partial w_{ij}^{(l)}} = a_i^{(l-1)} e_j^{(l)}$$
$$\frac{\partial J}{\partial b_j^{(l)}} = e_j^{(l)}$$

1.2.4 Bayesian Belief Network

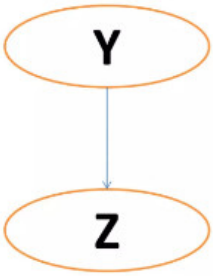
1. Khái niệm

Mạng niềm tin Bayesian được định nghĩa bởi hai phần: Đồ thị có hướng không tuần hoàn (DAG - Directed Acyclic Graph) và một tập các bảng xác suất có điều kiện (CPT - Conditional Probability Table). Mạng gồm các node thể hiện các biến và các link thể hiện sự phụ thuộc.

Mạng niềm tin Bayesian là đồ thị với các thuộc tính dưới đây:

- Nodes: Tập các biến ngẫu nhiên.
- Directed Links: liên kết từ node X đến node Y thể hiện X có ảnh hưởng trực tiếp đến Y.
- Mỗi node có một bảng xác suất có điều kiện (CPT) định lượng hệ quả của node cha lên node đó.
- Đồ thị không có chu trình có hướng.

Khi ta vẽ kết nổi của Y và Z như ở hình dưới thì Y ngay lập tức trở thành cha hay tiền nhân của Z còn Z là hậu duệ, con cháu của Y.



Hình 1.9

Mỗi biến độc lập có điều kiện với các biến không phải hậu duệ của nó trong đồ thị với biến cha cho trước.

Ý nghĩa:

- Để đại diện cho các mối quan hệ xác suất giữa các lớp khác nhau.
- Để tránh sự phụ thuộc giữa giá trị của các thuộc tính bằng phân phối xác suất có điều kiện kết hợp.
- Trong phân loại Naive Bayes, các thuộc tính độc lập có điều kiện.

2. Incremental Network Construction

- 1. Chọn tập các biến liên quan Xi mô tả domain.
- 2. Chọn thứ tự cho các biến.
- 3. Khi vẫn còn biến
  - Chọn một biến X và thêm một node cho nó.
  - Đặt node cha của X - parent(X) thành một số tập tối thiểu các node hiện có sao cho thuộc tính độc lập có điều kiện được thỏa mãn.
  - Định nghĩa CPT cho X

Điều kiện độc lập:

$$\mathrm{P}_{(X_1,X_2,.....X_n)} = P_{(X_n/X_{n-1},X_{n-2},X_1)}P_{(X_{n-1},X_{n-2},...,X_1)}$$

$$= \mathrm{P}_{(X_n/X_{n-1},X_{n-2},X_1)}P_{(X_{n-1}/X_{n-2},...,X_1)}...P_{(X_2/X_1)}P_{(X_2/X_1)}P_{(X_1)}$$

$$= \prod_{i=1}^{n} P_(X_i/Parents(P_{X_i}))$$

Suy ra :

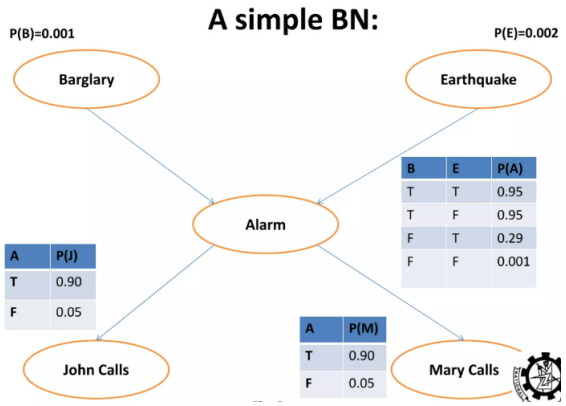
$$\mathrm{P}_{(X_n/X_{n-1},X_{n-2},X_1)} = P_{(X_n)/Parents(P_{X_n})}$$

Ví dụ:

Báo động chống trộm - Burglar Alarm tại nhà Khá đáng tin cậy trong việc phát hiện một vụ trộm - Burglary Ứng phó vào thời điểm động đất - Earthquakes

Hai người hàng xóm nghe thấy báo động - Alarm, gọi cảnh sát John luôn gọi khi nghe thấy báo động - Alarm, nhưng đôi khi nhầm lẫn giữa chuông điện thoại báo thức và các cuộc gọi. Mary thích mở nhạc âm lượng lớn và đôi khi bỏ lỡ báo động - Alarm cùng lúc.

Cho xác suất xảy ra vụ trộm - Burglary là 0.001. Xác suất xảy ra Động đất là 0.002. Xác suất John gọi điện khi nghe thấy tiếng báo động - Alarm là 0.9. Xác suất John không gọi điện khi nghe thấy tiếng báo động là 0.05. Xác suất Mary gọi điện khi nghe thấy tiếng báo động - Alarm là 0.9. Xác suất Mary không gọi điện khi nghe thấy tiếng báo động là 0.05.



Vậy làm cách nào để tìm $P_{(B/J)}$?

$$
\begin{aligned}
&\mathrm{P}_{(JB)} = P_{(JAB)} + P_{(JA'B)} \\
&= P_{(J/AB)}P_{(AB)} + P_{(J/A'B)}P_{(A'B)} \\
&= P_{(J/A)}P_{(AB)} + P_{(J/A')}P_{(A'B)} \\
&= (0,9) * P_{(AB)} + (0.05)P(A'B)P_{(AB)} \qquad (2)
\end{aligned}
$$

$$
\begin{aligned}
&\mathrm{P}_{(AB)} = P_{(ABE)} + P_{(ABE')} \\
&= P_{(A/BE')}P_{(BE')} + P_{(A/BE')}P_{(BE')} \\
&= P_{(A/B)}P_{(B)}P_{(E)} + P_{(A/B)}P_{(B)}P_{(E')} \\
&= (0.95)(0.001)(0.002) + (0.95)(0.001)(0.998) \\
&= 0.00095
\end{aligned}
$$

$$
\begin{aligned}
&\mathrm{P}_{(A'B)} = P_{(A'BE)} + P_{(A'BE')} \\
&= P_{(A'/BE)}P_{(BE)} + P_{(A'/BE')}P_{(BE')} \\
&= P_{(A'/B)}P_{(B)}P_{(E)} + P_{(A'/B)}P_{(B)}P_{(E')} \\
&= (0.05)(0.001)(0.002) + (0.05)(0.001)(0.998) \\
&= 0.00005
\end{aligned}
$$

Thay thế các giá trị vào trong phương trình (2) ta được:

$\mathrm{P}_{(JB)} = (0.9) * (0.00095) + (0.05) * (0.00005)$

Suy ra

$\mathbf{P}_{(JB)} = 0,00086$

Tìm $P_{(J)}$ như thế nào?

$\mathrm{P}_{(J)} = P_{(JA)} + P_{(JA')}$
$= P_{(J/A)}P_{(A)} + P_{(J/A')}P_{(A')}$
$= (0,9)P_{(A)} + (0.05)P(A')$ (3)

$\mathrm{P}_{(A)} = P_{(ABE)} + P_{(AB'E)} + P_{(ABE')} + P_{(AB'E')}$
$= P_{(A/BE)}P_{(BE)} + P_{(A/BE')}P_{(BE')} + P_{(A/B'E)}P_{(B'E)} + P_{(A/B'E')}P_{(B'E')}$
$= (0.95)P_{(B)}P_{(E)} + (0.95)P_{(B)}P_{(E')} + (0.29)P_{(B')}P_{(E)} + (0.001)P_{(B')}P_{(E')}$
$= (0.95)(0.001)(0.002)+(0.95)(0.001)(0.998)+(0.29)(0.999)(0.002)+(0.001)(0.998)(0.999)$
$= 0.0025$
$P_{(A)} = 0.0025 \rightarrow P_{(A')} = 0.9975$

Thay thế các giá trị vào phương trình (3) ta được:
$\mathrm{P}_{(J)} = (0.9)P_{(A)} + P_{(0.05)}P_{(A')}$
$= (0.9)(0.0025) + (0.05)(0.9975)$
$= 0.052125$
$\rightarrow P_{(J)} = 0.052125$

Thay $P_{(JB)}$ và $P_{(J)}$ vào phương trình (1), khi đó:
$\mathrm{P}_{(B/J)} = P_{(JB)}/P_{(J)}$
$= (0.00086)/(0.052125)$
$\rightarrow P_{(B/J)} = 0.016$

Ưu điểm:

- Có thể dễ dàng xử lý các tập dữ liệu không đầy đủ.
- Tìm hiểu được về các mối quan hệ nhân quả.
- Dễ dàng tạo điều kiện cho việc sử dụng tiên nghiệm.

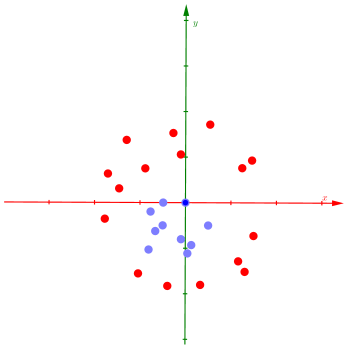
Nhược điểm:

- Phức tạp hơn khi phải xây dựng đồ thị.

1.2.5 SVM (Support Vector Machine)

1. Khái niệm

Trường hợp dữ liệu tách được tuyến tính và gần như tách được tuyến tính, ta có Hard Margin và Soft Margin SVM. Xét trường hợp dữ liệu thực sự không tách được tuyến tính như trong hình không sử dụng được Hard và Soft Margin SVM.



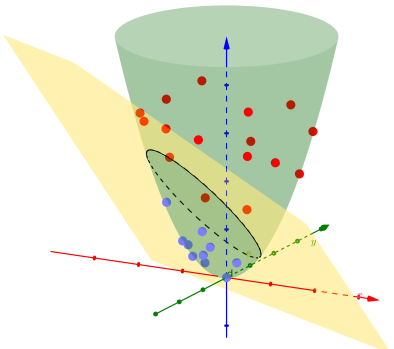
Hình 1.10

Trick: Nếu bổ sung chiều thứ ba là hàm số của hai chiều đã có

$$z = x^2 + y^2$$

Dữ liệu mới được phân bố trong không gian 3D: Những điểm xa tâm sẽ có giá trị z lớn hơn.

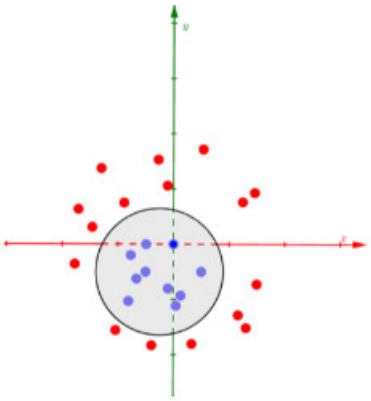
→ Các lớp dữ liệu tách được tuyến tính trong chiều thứ ba.



Hình 1.11

Các điểm dữ liệu 3D sẽ được phân bố trên một parabolic trong không gian 3D và đã trở nên phân biệt tuyến tính.

Thực tế dữ liệu gốc (2D) là tách được nhưng đường biên là phi tuyến.



Hình 1.12

Kernel Support Vector Machine

Tổng quát

- Kernel SVM $\iff$ Tìm hàm $\phi(x)$ biến đổi dữ liệu X từ không gian đặc trưng ban đầu thành dữ liệu trong không gian mới.
- Trong ví dụ, $\phi(\cdot)$ bổ sung một chiều dữ liệu mới (hay một đặc trưng mới), là hàm số của các đặc trưng đã biết.

Mục đích của $\phi(\cdot)$

- $\phi(\cdot)$ cần thỏa mãn: Trong không gian mới, dữ liệu của các lớp là tách được tuyến tính hoặc gần như tách được tuyến tính.
- Khi đó, có thể dùng các bộ phân lớp tuyến tính thông thường như PLA, Logistic Regression, hay Hard/Soft Margin SVM.

So sánh tương đối

- Hàm Kernel $\phi(\cdot)$ tương đồng với hàm Activation trong nhóm phương pháp dạng Neural Networks.
- Cả hai hàm đều có mục đích chuyển đổi giữa bài toán phân loại phi tuyến và bài toán phân loại tuyến tính.

  Điểm khác biệt: Nhiệm vụ của activation function là phá vỡ tính tuyến tính của mô hình. Trong khi đó, hàm $\phi(\cdot)$ biến dữ liệu không phân biệt tuyến tính thành phân biệt tuyến tính.

$\phi(\cdot)$ tạo ra dữ liệu mới có số chiều cao hơn số chiều của dữ liệu ban đầu

- Nếu tính toán trực tiếp cho từng điểm dữ liệu, sẽ dẫn tới các vấn đề về bộ nhớ và hiệu năng tính toán.
- Cách tiếp cận: sử dụng các kernel functions mô tả quan hệ giữa hai điểm dữ liệu bất kỳ trong không gian mới.
- Cụ thể: Thiết lập lại công thức của các bài toán tối ưu trong không gian mới sau khi đã sử dụng Kernel function.
- Kỹ thuật này được xây dựng dựa trên quan sát về bài toán đối ngẫu của SVM.

2. Thuật toán tối ưu

Bài toán đối ngẫu trong Soft Margin SVM cho dữ liệu hầu như tách được tuyến tính (Nearly linearly separable)

$$\begin{aligned} \lambda = \arg\max_{\lambda} & \sum_{n=1}^{N} \lambda_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \lambda_n \lambda_m y_n y_m \mathbf{x}_n^T \mathbf{x}_m \\ \text{subject to:} \quad & \sum_{n=1}^{N} \lambda_n y_n = 0 \\ & 0 \leq \lambda_n \leq C, \ \forall n = 1, 2, \ldots, N \end{aligned} \tag{1}$$

Trong đó:

N: Số cặp điểm dữ liệu trong tập training.
x_n: Feature vector của dữ liệu thứ n trong tập training.
y_n: Nhãn của dữ liệu thứ n, bằng 1 hoặc -1.
λ_n: Nhân tử Lagrange ứng với điểm dữ liệu thứ n.
$C > 0$: Hằng số cân đối độ lớn của margin và các điểm phải chấp nhận nằm trong vùng không an toàn. Khi C = ∞ (thực tế chọn số rất lớn), Soft Margin SVM trở thành Hard Margin SVM.

Sau khi giải được λ từ bài toán (1), với một điểm dữ liệu mới x, ta xác định nhãn y ứng với x bởi biểu thức:

$$\sum_{m \in \mathcal{S}} \lambda_m y_m \mathbf{x}_m^T \mathbf{x} + \frac{1}{N_{\mathcal{M}}} \sum_{n \in \mathcal{M}} \left(y_n - \sum_{m \in \mathcal{S}} \lambda_m y_m \mathbf{x}_m^T \mathbf{x}_n \right) \qquad (2)$$

Trong đó:

M = { n : $0 < \lambda_n <$ C }: Tập hợp những điểm nằm trên margin.
S = { n : $0 < \lambda_n$ }: Tập hợp các điểm support.
N_m: Số phần tử của M.

- Thực tế, các phân lớp dữ liệu có thể có đường biên phi tuyến, tức là không hầu như tách được tuyến tính (non - nearly linearly separable).
- Trong trường hợp trên, nghiệm của (1) không phản ánh đúng phân lớp dữ liệu.
- Từ ý tưởng của ví dụ trên, ta tìm $\phi(\cdot)$: x $|\rightarrow \phi(x)$ - Điểm dữ liệu trong không gian mới. Trong không gian này, các lớp dữ liệu (sau biến đổi) là hầu như tách được tuyến tính.

Bài toán đối ngẫu (1) trong không gian mới (sau biến đổi bởi ϕ) trở thành:

$$\lambda = \underset{\lambda}{argmax} \sum_{n=1}^{N} \lambda_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \lambda_n \lambda_m y_n y_m \Phi(x_n)\, \Phi(x_m) \qquad (3)$$

Subject to : $\sum_{n=1}^{N} \lambda_n y_n = 0; \qquad 0 \le \lambda_n \le C\ , \forall\, n = 1,2, \dots N$

Và nhãn y của mẫu dữ liệu mới x sẽ được xác định bởi

$$y \coloneqq sign\,\{w^T \Phi(x) + b\}$$

$$= \left\{ \sum_{m \in S} \lambda_m y_m\, \Phi(x_m)^T + \frac{1}{N_M} \sum_{n \in M} \left(y_n - \sum_{m \in S} \lambda_m y_m\, \Phi(x_m) \right) \right\} \qquad (4)$$

- $M = \{n: 0 < \lambda_n < C\}$: là tập hợp những điểm nằm trên margin.
- $S = \{n: 0 < \lambda_n\}$: là tập hợp các điểm support.
- N_M : là số phần tử của N

Nhận xét

- Số chiều của $\Phi(x)$ thường là rất lớn và nếu tính các $\Phi(x)$ dẫn tới cần tính tiếp tích vô hướng $\Phi(x_m)^T\, \Phi(x_n)$; $\Phi(x_m)^T\, \Phi(x)$
- Trong bài toán (3) và biểu thức (4), chúng ta không cần tính trực tiếp $\Phi(x)$cho mọi điểm dữ liệu
- Ta chỉ thực sự cần tính $\Phi(x)^T\, \Phi(z)$ cho mọi cặp dữ liệu xz bất kì

Kernel trick: Những phương pháp dựa trên kỹ thuật này, tức thay vì trực tiếp tính tọa độ của một điểm trong không gian mới, ta đi tính tích vô hướng giữa hai điểm trong không gian mới.

Định nghãi hàm Kernel : $k(x, z) = \Phi(x)^T\, \Phi(z)$ lúc đó bài toán trở thành

$$\lambda = \underset{\lambda}{argmax} \sum_{n=1}^{N} \lambda_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \lambda_n \lambda_m y_n y_m \Phi(x_n)\, \Phi(x_m) \qquad (5)$$

Subject to : $\sum_{n=1}^{N} \lambda_n y_n = 0; \qquad 0 \le \lambda_n \le C\ , \forall\, n = 1,2, \dots N$

Lúc đó với mọi c ∈ $\mathbb{R}^N$ ta có

$$\mathrm{C}^{\mathrm{T}} K\, c = \sum_{n=1}^{\mathrm{N}} \sum_{m=1}^{\mathrm{N}} k(x_m, x_n) c_n c_m \geq 0$$

Vậy bản chất điều kiện Mercer là để đảm bảo ánh xạ K là đối xứng nửa xác định dương.

Quay lại bài toán (5) với hàm kernel $k(x, z)$ đối xứng, thỏa mãn (7), đặt $C_n = y_n \lambda_n$ ta thu được với mọi $\lambda \in \mathbb{R}^N$

$$\lambda^{\mathrm{T}} K \lambda = \sum_{n=1}^{\mathrm{N}} \sum_{m=1}^{\mathrm{N}} \lambda_n \lambda_m y_n y_m = \sum_{n=1}^{\mathrm{N}} \sum_{m=1}^{\mathrm{N}} C_n\, C_m \geq 0 \qquad (8)$$

Ở đây K là một ma trận đối xứng với các phần tử $K_{nm} = y_n y_m k(x_n x_m)$. Từ (8) suy ra ma trận K đối xứng nửa xác định dương.

Vậy, bài toán tối ưu (5) có ràng buộc lồi & hàm mục tiêu lồi *(quadratic form)*.

Do đó nó có thể được giải tương tự như trong phương pháp SoftMargin SVM.

Bài toán (5) có dạng quadratic nên sẽ có nghiệm toàn cục, và có thể giải bằng một số thư viện, ví dụ CVXOPT. Sau khi tìm được các λ_n, với mỗi mẫu dữ liệu x mới ta dự đoán nhãn (phân lớp) của nó theo (6).

Nhắc lại việc sử dụng CVXOPT solver trong việc giải bài toán quadratic programming. Bài toán dạng:

$$\min_x \left\{ \frac{1}{2} x^T P x + q x \right\}$$

Subject to : $Gx \leq \mathrm{h}$

$Ax = \mathrm{b}$

So sánh với (5) ta thấy các toán tử trong bài toán QP được xác định bởi

- $P = K = (k_{nm})$: $k_{nm} = y_n y_m k(x_n x_m)$ là ma trận đã được nêu ở trên
- $q = -1(ma\ trận\ đơn\ vị)$
- $h = (0, \dots, 0, C, \dots, C)^T$ – vecto 2N phần tử và

$$G = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Sau khi giải được λ từ (5), ta xác định nhãn của mẫu dữ liệu mới x theo (6).

Hàm Sigmoid cũng được sử dụng làm kernel:

Công thức tổng quát của kernel:

$$k(x, z) = \tanh(\gamma x^T z + r)$$

Khi sử dụng thư viện sklearn, lựa chọn sigmoid thông qua kernel = 'sigmoid'.

Chương 2

Phân tích xử lý bộ dữ liệu

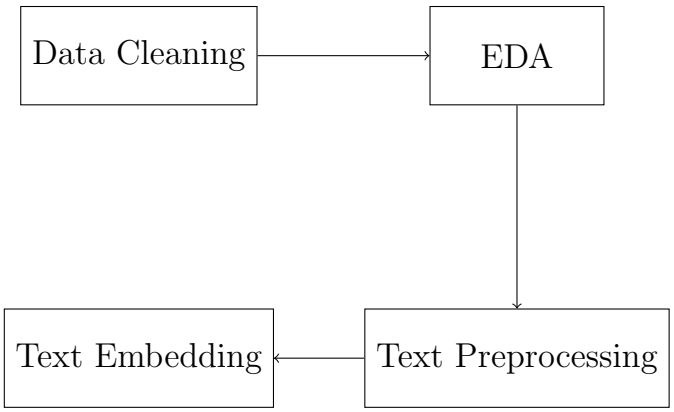
2.1 Bộ dữ liệu SMS Spam Collection

- SMS Spam Collection là bộ dữ liệu được sử dụng rộng rãi trong lĩnh vực xử lý ngôn ngữ tự nhiên và phân loại tin nhắn rác. Bộ dữ liệu này được thu thập tại Anh và chứa hơn 5572 tin nhắn SMS bằng tiếng Anh, trong đó có 4825 tin nhắn thường (ham) và 747 tin nhắn rác (spam).
- Bộ dữ liệu gồm các tệp văn bản, chứa thông tin về một tin nhắn, bao gồm cả nội dung của tin nhắn, loại tin nhắn (thông thường hoặc tin rác) và một số thông tin khác như thời gian gửi và số điện thoại của người gửi.
- Được sử dụng để huấn luyện các mô hình phân loại tin nhắn là tin nhắn rác hoặc tin nhắn thường.
- Bộ dữ liệu SMS Spam Collection chỉ chứa các tin nhắn tiếng Anh và không phải là một bộ dữ liệu đại diện cho mọi loại tin nhắn rác. Do đó khi sử dụng bộ dữ liệu này, cần xem xét kỹ càng để đảm bảo tính khả dụng của nó đối với ứng dụng cụ thể.
- *Bộ dữ liệu này bao gồm 2 cột trong đó:*
  - Cột đầu tiên: Ứng với 2 nhãn: ham hoặc spam.
  - Cột còn lại: Chứa các tin nhắn tiếng anh.

2.2 Tiền xử lý dữ liệu

Gồm 4 giai đoạn chính:

- Data Cleaning: Làm sạch dữ liệu
- EDA (Exploratory Data Analysis): Phân tích khai phá dữ liệu
- Text Preprocessing: Tiền xử lý văn bản
- Text Embedding: Nhúng văn bản



Hình 2.1 Quá trình tiền xử lý dữ liệu

2.2.1 Data Cleaning

Data Cleaning là quá trình làm sạch và chuẩn hóa dữ liệu nhằm đảm bảo tính chính xác, đầy đủ của dữ liệu. Trong quá trình này, ta sẽ xác định và loại bỏ các giá trị dữ liệu lỗi, thiếu sót, trùng lặp hoặc không hợp lệ.

Khi dữ liệu đã được làm sạch, nó sẽ trở nên đáng tin cậy hơn và được sử dụng để phân tích, đưa ra các quyết định có cơ sở.

Trong project này, quy trình làm sạch dữ liệu trải qua các công đoạn sau:

- Kiểm tra thông tin bộ dữ liệu. Sau đó loại bỏ đi các trường trống (nếu có) và format lại tên các trường.
- Mã hóa nhãn thành các giá trị nằm trong (0, n-1); với n là số nhãn, bộ dữ liệu có 2 nhãn "ham: 0", "spam: 1".
- Kiểm tra các giá trị trống (null) và các giá trị trùng lặp (duplicate).

Bộ dữ liệu này có 403 giá trị trùng.
→ Sử dụng Drop duplicate để loại bỏ các giá trị trùng.

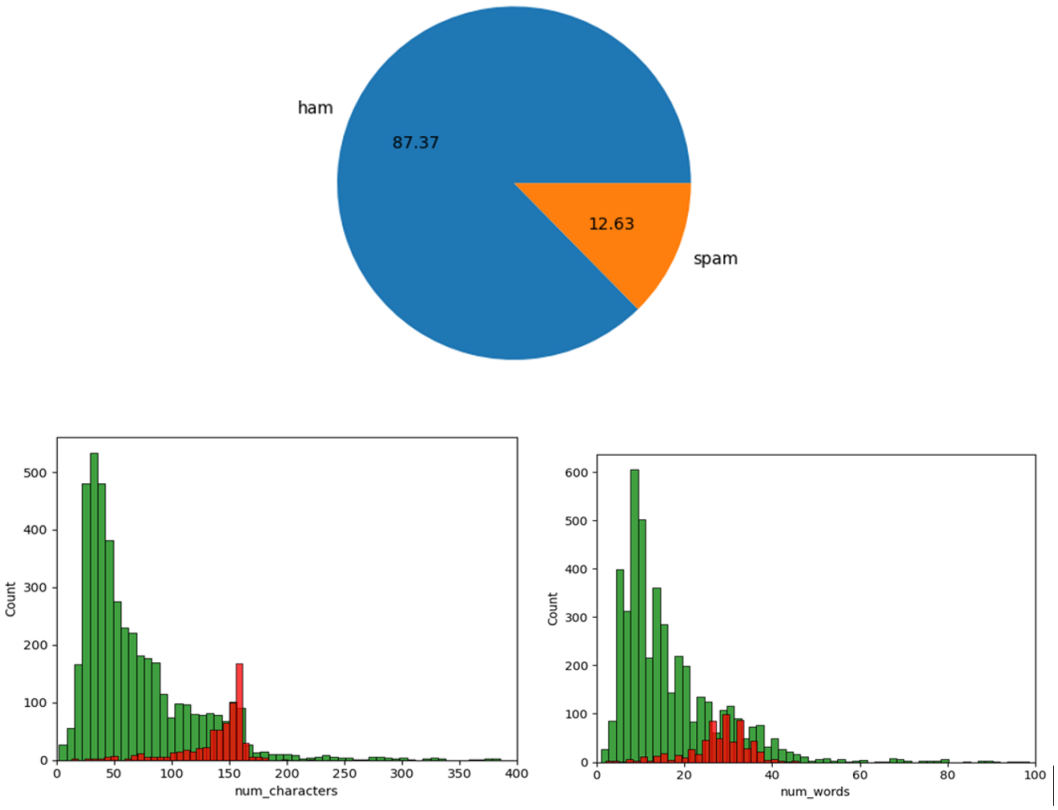
2.2.2 Exploratory Data Analysis

Với EDA, người phân tích sẽ khám phá và phân tích dữ liệu để hiểu và tìm ra mối quan hệ, xu hướng và đặc điểm của các biến. Từ đó có thể hiểu dữ liệu một cách sâu sắc hơn và đưa ra những phát hiện, giả định ban đầu cho các bước tiếp theo của quy trình phân tích dữ liệu.

Kiểm tra số lượng dữ liệu thuộc về 2 nhãn "ham" và "spam":

- Nhãn 0 (ham) có 4516 giá trị
- Nhãn 1 (spam) có 653 giá trị.

→ Bộ dữ liệu không cân bằng (Do bản chất dữ liệu trong thực tế là tin nhắn thường nhiều hơn tin nhắn rác).



Hình 2.2 Output của quá trình phân tích khai phá dữ liệu

Sử dụng thư viện 'nltk' (natural language toolkit) phân tích:

- Số ký tự, số từ, trong tin nhắn rác nhiều hơn
- Tần suất xuất hiện của tin nhắn thường nhiều hơn tin nhắn rác

2.2.3 Text Preprocessing

Thực hiện theo các bước sau:
1. Tokenization: Tách một cụm từ, câu, đoạn văn thành các đơn vị nhỏ - tokens.
2. Loại bỏ các ký tự đặc biệt và dấu câu.
3. Chuyển đổi chữ hoa thành chữ thường Ví dụ, "Hello" và "hello" sẽ được coi là cùng một từ trong quá trình phân tích văn bản.
4. Loại bỏ các stopword để tập trung vào những từ mang thông tin quan trọng.
5. Chuẩn hóa các từ đồng nghĩa bằng cách đưa về các từ gốc (stemming, lemmatizer).
6. Kiểm tra các từ thường xuyên xuất hiện trong văn bản.

2.2.4 Chuẩn hóa các từ về dạng gốc

• Stemming: Quá trình chuyển đổi các từ trong văn bản về dạng từ gốc bằng cách loại bỏ các hậu tố.

Ví dụ: Từ "running" sẽ được chuyển thành từ "run" thông qua việc loại bỏ hậu tố "ing".

Stemming dựa theo nguyên tắc:

$$\text{Generous} \Longrightarrow \textit{Gener}$$
$$\text{Sings} \Longrightarrow \textit{Sing}$$
$$\text{Changing} \Longrightarrow \textit{Chang}$$

• Lemmatizer: Quá trình chuyển đổi các từ trong văn bản về dạng từ điển gốc (lemma) bằng cách sử dụng thông tin về ngữ cảnh và từ điển.

Ví dụ: "better" sẽ được chuyển đổi về từ "good".

Lemmatizer dựa theo từ điển:

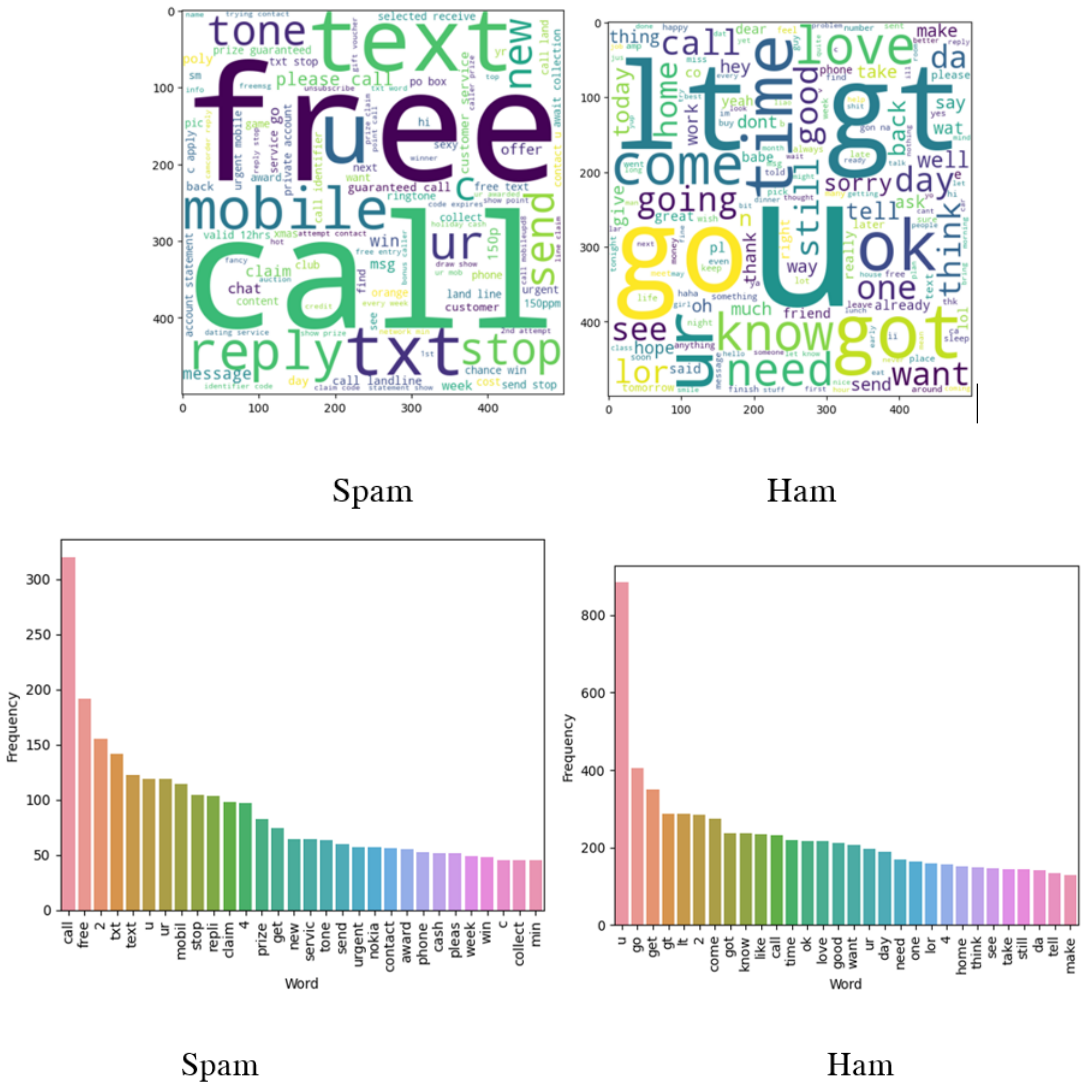
$$\text{Better} \Longrightarrow \textit{Good}$$
$$\text{Sings} \Longrightarrow \textit{Sing}$$
$$\text{Changing} \Longrightarrow \textit{Change}$$

2.2.5 Porter's stemmer

Một trong những phương pháp chuyển từ về dạng gốc phổ biến nhất được đề xuất vào năm 1980, dựa trên quy tắc các hậu tố được tạo thành từ sự kết hợp của các hậu tố nhỏ hơn và đơn giản hơn. Chỉ áp dụng được trong tiếng Anh, đầu ra của thuật toán không nhất thiết phải là một từ có nghĩa.

Một số quy tắc của Porter's stemmer:

$$\text{SSES} \Longrightarrow SS$$
$$\text{ISE} \Longrightarrow I$$



Hình 2.3 Output của quá trình tiền xử lý văn bản

2.2.6 Text embedding

• Text embedding là quá trình biểu diễn một từ, cụm từ hoặc văn bản thành vector số trong không gian đa chiều dựa trên mối quan hệ ngữ nghĩa giữa chúng.
• Các vector số này được tạo ra từ các thuật toán biểu diễn từ, ví dụ như các thuật toán: Word2Vec, GloVe, FastText, TF-IDF.
• Các vector số được xây dựng để có khoảng cách và hình dạng tương tự giữa các từ mang nghĩa tương đồng. Nhờ vậy, các từ được biểu diễn bằng các vector số này có thể được sử dụng để phân tích ngữ nghĩa của câu hoặc văn bản.

Thuật toán TF-IDF:
1. TF-IDF là viết tắt của Term Frequency-Inverse Document Frequency.
2. Phương pháp đánh giá độ quan trọng của một từ dựa trên tần suất xuất hiện trong văn bản và tần suất xuất hiện trong bộ dữ liệu.
3. Term Frequency (Tần suất xuất hiện của từ): Chỉ số này cho biết tần suất xuất hiện của một từ trong một tài liệu văn bản cụ thể.
4. Inverse Document Frequency (Nghịch đảo tần suất tài liệu): Chỉ số này cho biết tần suất xuất hiện của một từ trong cả bộ dữ liệu.

$$W_{ij} = tf_{ij} * \log \frac{N}{d_{ij}}$$

Trong đó:

i: Từ cần tính giá trị TF-IDF
j: Tài liệu cần tính giá trị TF-IDF
N: Tổng số tài liệu trong bộ dữ liệu
$tf_{i,j}$: Tần suất xuất hiện từ i trong tài liệu j
df_i: Tổng số tài liệu chứa từ i

Ví dụ:
Câu 1: "This movie is very scary and long"
Câu 2: "This movie is not scary and is slow"
Câu 3: "This movie is spooky and good"
Xét câu 2, có các từ: "This", "movie", "is", "not", "scary", "and", "is", "slow"
TF(This) = $\frac{1}{8}$
IDF(This) = $\log(\frac{3}{3}) = 0$
TF-IDF(This) = $\frac{1}{8}.0 = 0$
→ Từ "This" có độ quan trọng bằng 0 trong bộ dữ liệu trên.

Term	Câu 1	Câu 2	Câu 3	IDF	TF-IDF (Câu 1)	TF-IDF (Câu 2)	TF-IDF (Câu 3)
This	1	1	1	0.00	0.000	0.000	0.000
movie	1	1	1	0.00	0.000	0.000	0.000
is	1	2	1	0.00	0.000	0.000	0.000
very	1	0	0	0.48	0.068	0.000	0.000
scary	1	1	0	0.18	0.025	0.022	0.000
and	1	1	1	0.00	0.000	0.000	0.000
long	1	0	0	0.48	0.068	0.000	0.000
not	0	1	0	0.48	0.000	0.060	0.000
slow	0	1	0	0.48	0.000	0.060	0.000
spooky	0	0	1	0.48	0.000	0.000	0.080
good	0	0	1	0.48	0.000	0.000	0.080

Hình 2.4

Chương 3

Xây dựng và thực thi chương trình

3.1 Tổng quan

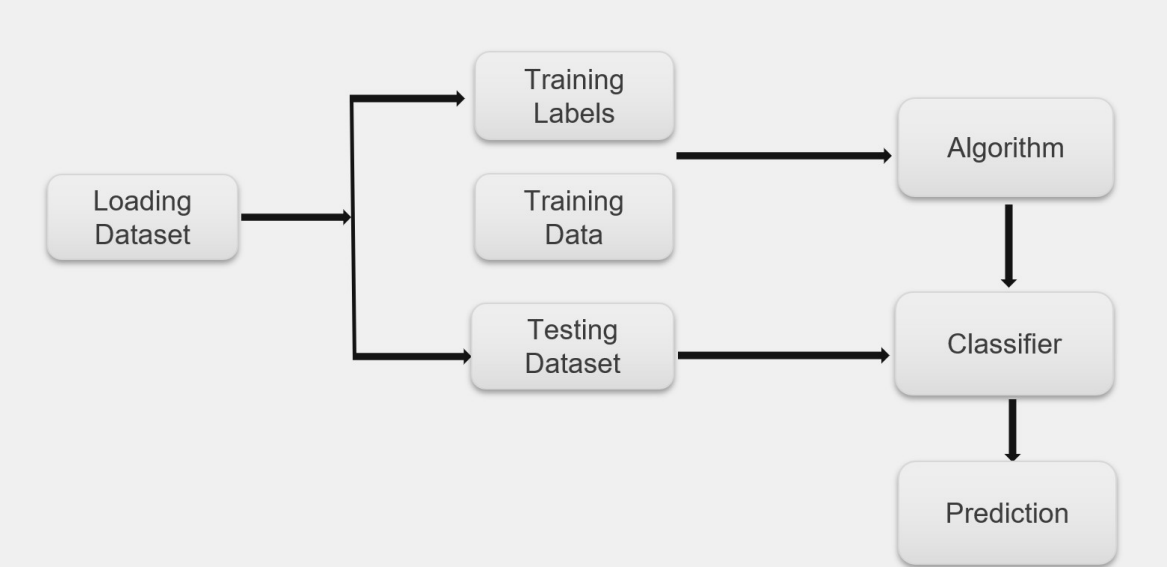
Trong chương tình này chúng em sử dụng những thư viện sau:

numpy, pandas, matplotlib, seaborn, sklearn, nltk.

Bộ dữ liệu sau khi tiền xử lý gồm: 5169 tin nhắn (4516 giá trị ham, 653 giá trị spam). Trong đó:

- Bộ dữ liệu huấn luyện: 80% của bộ dữ liệu sau khi tiền xử lý.
- Bộ dữ liệu kiểm định: 20% của bộ dữ liệu sau khi tiền xử lý.

Phương pháp trích xuất đặc trưng: TF-IDF (Số đặc trưng: 3000)
Đầu vào mô hình: Dữ liệu dưới dạng vectơ số
Đầu ra mô hình: Không spam (0) hoặc spam (1)



Hình 2.5

Chỉ số đánh giá: Accuracy, Precision, Recall, F1

$$Precision = \frac{TP}{(TP + FP)}$$

$$Recall = \frac{TP}{(TP + FN)}$$

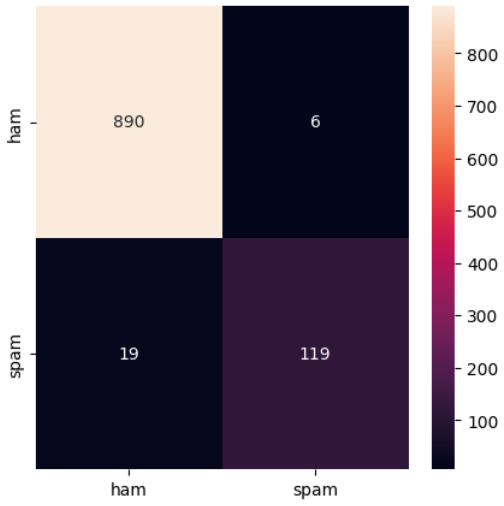
$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)}$$

$$F1 = \frac{2.(Precision.Recall)}{(Precision + Recall)}$$

3.2 Mô hình Artificial Neural Network

Kết quả chạy trên bộ dữ liệu kiểm định:

```
Execution time: --- 0.05915403366088867 seconds ---
Accuracy 0.9729206963249516
Precision 0.9661016949152542
Recall 0.8260869565217391
F1 0.890625
```

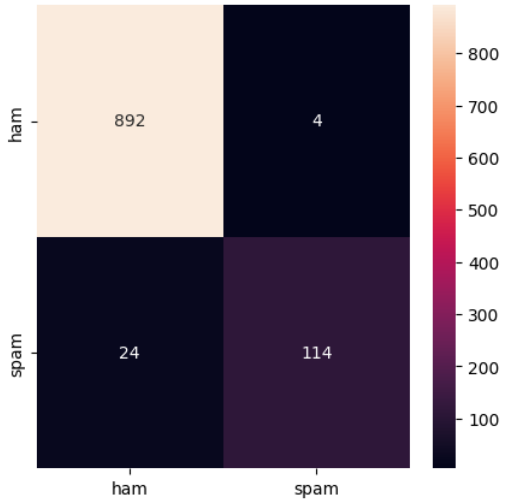


Hình 3.1

3.3 Mô hình Multinomial Naive Bayes

Kết quả chạy trên bộ dữ liệu kiểm định:

```
Execution time: --- 0.05915403366088867 seconds ---
Accuracy 0.9729206963249516
Precision 0.9661016949152542
Recall 0.8260869565217391
F1 0.890625
```

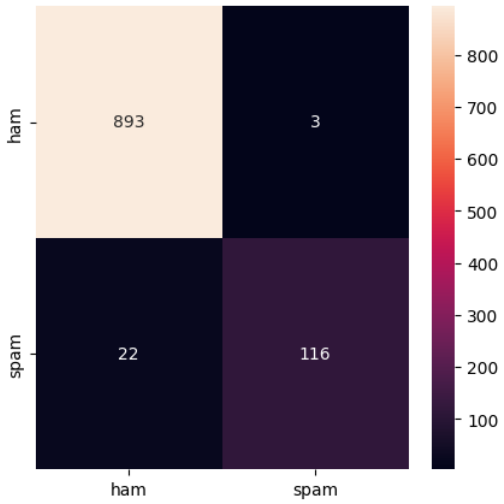


Hình 3.2

3.4 Mô hình Support Vector Machine

Kết quả chạy trên bộ dữ liệu kiểm định:

```
Execution time: --- 6.117401838302612 seconds ---
Accuracy 0.9758220502901354
Precision 0.9747899159663865
Recall 0.8405797101449275
F1 0.9027237354085604
```

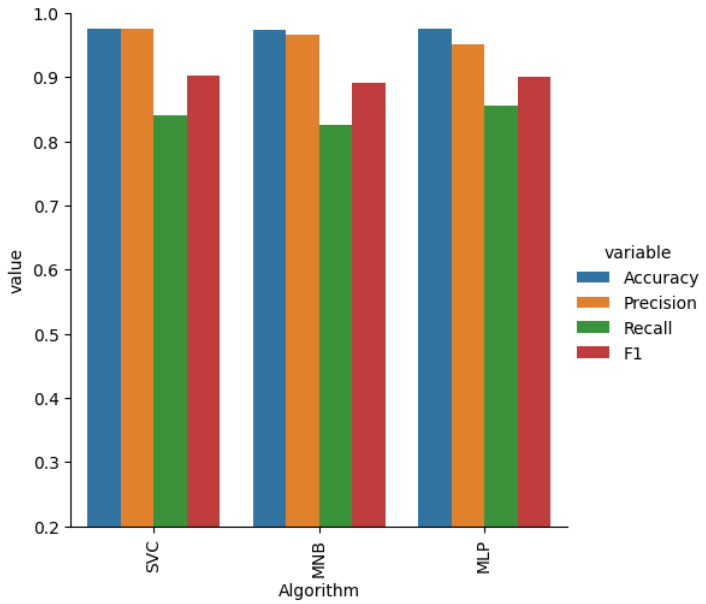


Hình 3.3

3.5 So sánh độ chính xác và thời gian chạy giữa 3 mô hình

1. Độ chính xác trên 4 thang đo

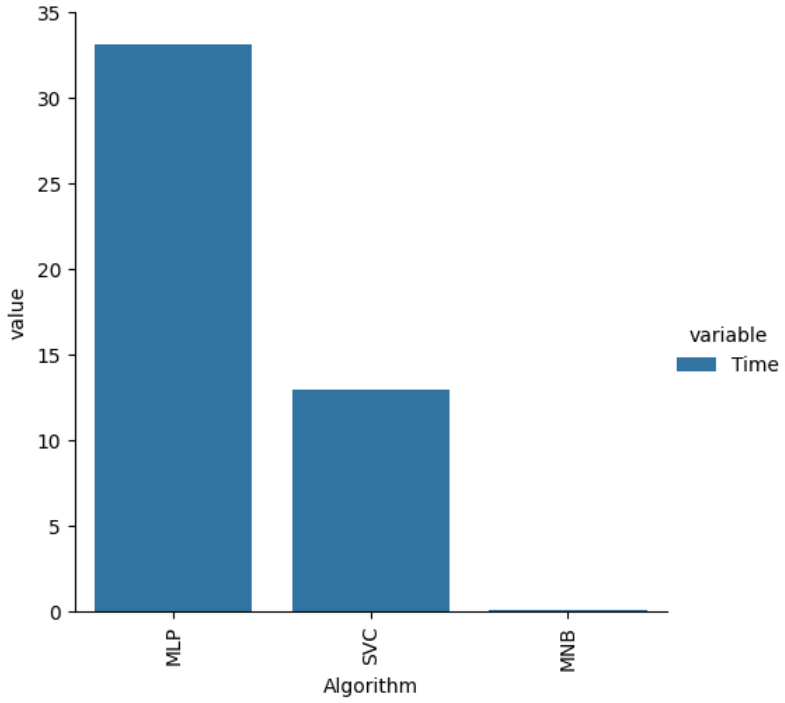
	Algorithm	Accuracy	Precision	Recall	F1
0	SVC	0.975822	0.974790	0.840580	0.902724
2	MNB	0.972921	0.966102	0.826087	0.890625
1	MLP	0.974855	0.951613	0.855072	0.900763



Hình 3.4

2. Thời gian thực thi của 3 mô hình

	Algorithm	variable	value
0	MLP	Time	33.107310
1	SVC	Time	12.959101
2	MNB	Time	0.036999



Hình 3.5

3.6 Nhận xét

Mô hình có độ hiệu quả tốt nhất trên bộ dữ liệu kiểm định xếp theo thứ tự từ cao xuống thấp lần lượt là:
1. Multi Layer Perceptron
2. Support Vector Machine
3. Multinomial Naive Bayes

Mô hình chạy nhanh nhất trên bộ dữ liệu kiểm định xếp theo thứ tự từ cao xuống thấp lần lượt là:
1. Multinomial Naive Bayes
2. Support Vector Machine
3. Multi Layer Perceptron

Nhận thấy độ chính xác của mô hình Multinomial Naive Bayes kém hơn không đáng kể so với Multi Layer Perceptron, trong khi đó thời gian xử lý nhanh hơn gấp khoảng 800 lần nên mô hình Multinomial Naive Bayes là phù hợp nhất trong 3 mô hình cho bộ dữ liệu Spam SMS Collection.

Chương 4

Kết luận và hướng phát triển

Phát hiện tin nhắn rác đã trở thành một đề tài nghiên cứu vô cùng quan trọng trong an toàn thông tin. Sự phát triển của trí tuệ nhân tạo trong thời gian gần đây góp phần ngăn chặn, phát hiện tin nhắn rác với độ chính xác cao.

Trong đó các mô hình chúng em sử dụng đã phát huy nhiều ưu điểm cho vấn đề này. Trong đề tài này, chúng em đã tiến hành tìm hiểu những lý thuyết về khai phá dữ liệu để nghiên cứu mô hình từ đó xây dựng mô hình ứng dụng thực tế nhằm áp dụng vào công việc hiện tại.

Mô hình của bọn em là sự kết hợp của thuật toán trích xuất đặc trưng TF-IDF và thuật toán phân loại Multinomial NB.

• Sau khi được huấn luyện trên bộ dữ liệu Spam SMS Collection, mô hình đã đưa ra kết quả khá tốt khi phát hiện và phân loại thành công tin nhắn rác.

• Tuy nhiên, vẫn còn một số trường hợp mà mô hình không phân loại chính xác được. Vì vậy trong tương lai nhóm em sẽ có thêm những cải tiến cho mô hình: Huấn luyện trên đa dạng các bộ dữ liệu về SMS, kết hợp với việc tối ưu các tham số và thuật toán giúp tăng tốc độ và hiệu suất của mô hình...

• Hạn chế trong bài nghiên cứu này là chúng em mới khai thác trên Spam SMS Tiếng Anh, chưa áp dụng với Spam SMS Tiếng Việt. Trong tương lai, chúng em sẽ phát triển cho các ngôn ngữ khác.

Nhóm 11 muốn gửi cảm ơn sâu sắc tới thầy Lê Hoàng Sơn và thầy Phạm Huy Thông vì đã tận tình quan tâm và hướng dẫn chúng em hoàn thành báo cáo này. Chúng em xin chân thành cảm ơn!

Tài liệu tham khảo

[1] Miszalska, I., Zabierowski, W., Napieralski, A. (2007, February). "Selected Methods of Spam Filtering in Email." In CAD Systems in Microelectronics, 2007. CADSM'07. 9th International Conference-The Experience of Designing and Applications of (pp. 507-513). IEEE.

[2] Scholar, M. (2010). "Supervised learning approach for spam classification analysis using data mining tools." organization, 2(08), 2760-2766.

[3] Youn, S., McLeod, D. (2007). "A comparative study for email classification." In Advances and Innovations in Systems, Computing Sciences and Software Engineering (pp. 387-391). Springer Netherlands.

[4] Xiao-li, C., Pei-yu, L., Zhen-fang, Z., Ye, Q. (2009, August). "A method of spam filtering based on weighted support vector machines." In IT in Medicine Education, 2009. ITIME'09. IEEE International Symposium on (Vol. 1, pp. 947-950). IEEE.

[5] Sculley, D., Wachman, G. M. (2007, July). "Relaxed online SVMs for spam filtering." In Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval (pp. 415-422).

[6] Chan, T. Y., Ji, J., Zhao, Q. "Learning to Detect Spam: Naive-Euclidean Approach." International Journal of Signal Processing, 1.

[7] Puniškis, D., Laurutis, R., Dirmeikis, R. (2006). "An artificial neural nets for spam e–mail recognition." Elektronika ir Elektrotechnika (Electronics and Electrical Engineering), 5(69), 73-76.

[8] Drucker, H., Wu, D., Vapnik, V. N. (1999). "Support vector machines for spam categorization." Neural Networks, IEEE Transactions on, 10(5), 1048-1054.

[9] Provost, J. (1999). "Naıve-Bayes vs. Rule-Learning in Classification of Email." University of Texas at Austin.

[10] Medlock, B. (2006, July). "An Adaptive, Semi-Structured Language Model Approach to Spam Filtering on a New Corpus." In CEAS.

[11] Kiran, S. R. (2009). "Spam or not spam–That is the question."

[12] Sharma, S., Arora, A. (2013). "Adaptive Approach for Spam Detection." International Journal of Computer Science Issues (IJCSI), 10(4).

[13] Theodoridis S. and Koutroumbas K., Pattern Recognition, USA, Academic Press, 2009.

[14] Rizk Y., Mitri N., Awad M., "A Local Mixture Based SVM for an Efficient Supervised Binary Classification", International Joint Conference on Neural Networks (IJCNN), Dallas, TX, August 4–9, 2013.