HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Huỳnh Vũ Trường Giang

PHÁT HIỆN LỖI TRONG HỆ THỐNG MẠNG VÀ TRUYỀN THÔNG

Chuyên ngành: Hệ Thống thông tin

Mã số: 8.48.01.04

TÓM TẮT LUẬN VĂN THẠC SĨ

Tp. HCM - NĂM 2021

Luận văn được hoàn thành tại:

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Người hướng dẫn khoa học: **PGS. TS. Trần Mạnh Hà**

Phản biện 1: ……..

Phản biện 2: ……..

Luận văn sẽ được bảo vệ trước Hội đồng chấm luận văn thạc sĩ tại Học viện Công nghệ Bưu chính Viễn thông

Vào lúc: ..... giờ ..... ngày ..... tháng ..... năm .....

Có thể tìm hiểu luận văn tại:

- Thư viện của Học viện Công nghệ Bưu chính Viễn thông

MỞ ĐẦU

Ngày nay, với sự phát triển của hệ thống mạng và truyền thông cả về sự đa dạng, độ phức tạp và độ ổn định thì việc phát hiện lỗi trong mạng truyền thông và hệ thống phân tán thường yêu cầu sự tham gia của các công cụ hỗ trợ và chuyên môn của người vận hành hệ thống. Hệ thống giám sát đưa ra các sự kiện lỗi sau đó được chuyển tiếp cho người vận hành hệ thống để phân tích và tạo báo cáo lỗi. Việc xây dựng các chức năng phát hiện lỗi là một thách thức vì rất khó để có cách tiếp cận hiệu quả thay thế kiến thức và cơ chế suy luận của người vận hành hệ thống, đặc biệt là một số vấn đề liên quan đến tính khả dụng, khả năng chịu lỗi và khả năng dự đoán hiệu suất là rất khó phát hiện trên mạng truyền thông diện rộng và hệ thống phân tán với độ phức tạp, khả năng mở rộng và tầm quan trọng cao.

Hiện tại không có cách thực tế nào để phân tích lỗi của một thành phần trong hệ thống mạng một cách tự động. Nó được để lại như một nhiệm vụ yêu cầu người vận hành thực hiện thủ công bằng cách sử dụng vô số công cụ để thu thập thông tin về hoạt động hiện tại của các thiết bị trên hệ thống. Các câu hỏi yêu cầu câu trả lời trong luận văn này là:

- Liệu có một công cụ tự động có thể hỗ trợ thực hiện quá trình trên?

- Chúng ta có thể xây dựng một mô hình có khả năng thu thập tất cả thông tin lỗi này không, hãy hiểu rõ về nó,

và do đó tiết kiệm thời gian và tài nguyên cho người vận hành?

Mục đích của luận văn là xây dựng một mô hình có thể thực hiện tự động đánh giá mức độ nghiêm trọng của lỗi. Trước đây, khó có được thông tin các lỗi đã xảy ra khi thu thập bằng cách thủ công. Luận văn muốn khai thác thông tin có sẵn tại các Bug Tracking System và sử dụng nó một cách hiệu quả nhất có thể để tiết lộ nguyên nhân của lỗi và đánh giá mức độ nghiêm trọng của lỗi. Luận văn này nhằm mục đích vạch ra con đường hướng tới một cách tiếp cận tự chủ hơn để quản lý lỗi bằng cách phát triển một mô hình dựa trên việc phân lớp và dự đoán theo thuật toán Rừng ngẫu nhiên và phương pháp tf x idf.

Xuất phát từ những lý do trên, học viên chọn thực hiện đề tài luận văn tốt nghiệp chương trình đào tạo thạc sĩ có tên "**Phát hiện lỗi trong hệ thống mạng và truyền thông**".

Mục tiêu của luận văn là đưa các cảnh báo mức độ nghiêm trọng của lỗi một cách tự động thay vì thực hiện thăm dò thủ công. Nhằm mục đích đưa ra các cảnh báo này một cách kịp thời, đáng tin cậy.

Nội dung của luận văn được trình bày trong ba chương nội dung chính như sau:

Chương 1: Nghiên cứu tổng quan

Đưa ra lĩnh vực nghiên cứu cũng như mang lại cho người đọc kiến thức về các khái niệm được sử dụng trong luận văn.

Chương 2: Tìm hiểu cách phân loại lỗi mạng

Nghiên cứu mô hình, thuộc tính của lỗi mạng và phương pháp khai phá nội dung của lỗi mạng.

Chương 3: Xây dựng mô hình phân loại lỗi mạng

Mô tả công việc được thực hiện để trả lời các câu hỏi nghiên cứu.

Chương 4: Phân tích và đánh giá kết quả thực hiện

Chương 5: Kết luận

CHƯƠNG 1. TỔNG QUAN VỀ PHÂN LỚP DỮ LIỆU VÀ HỌC MÁY

1.1. Giới thiệu bài toán phân lớp dữ liệu và các vấn đề liên quan

1.1.1. Khái niệm về phân lớp dữ liệu và bài toán phân lớp dữ liệu

Phân lớp (classification) dữ liệu là một tiến trình xử lý nhằm xếp các mẫu dữ liệu hay các đối tượng vào một trong các lớp đã được định nghĩa trước. Các mẫu dữ liệu hay các đối tượng được xếp vào các lớp dựa trên giá trị của các thuộc tính (attributes) của mẫu dữ liệu hay đối tượng. Quá trình phân lớp dữ liệu kết thúc khi tất cả các dữ liệu đã được xếp vào các lớp tương ứng. Khi đó, mỗi lớp dữ liệu được đặc trưng bởi tập các thuộc tính của các đối tượng chứa trong lớp đó.

Quy trình giải quyết bài toán phân lớp dữ liệu

(1) Giai đoạn huấn luyện

(2) Giai đoạn kiểm chứng

1.1.2. Các độ đo đánh giá mô hình phân lớp dữ liệu

(1) Độ đo Precision (Mức chính xác)

- Định nghĩa: Precision = TP / (TP + FP).

- Ý nghĩa: Giá trị Precision càng cao thể hiện khả năng càng cao để một kết quả phân lớp dữ liệu được đưa ra bởi bộ phân lớp là chính xác.

(2) Độ đo Recall (Độ bao phủ, độ nhạy hoặc độ triệu hồi)

- Định nghĩa: Recall = TP / (TP + FN).

- Ý nghĩa: Giá trị Recall càng cao thể hiện khả năng kết quả đúng trong số các kết quả đưa ra của bộ phân lớp càng cao.

(3) Độ đo Accuracy (Độ chính xác)

- Định nghĩa: Accuracy = (TP + TN) / (TP + TN + FP + FN) * 100%.

- Ý nghĩa: Accuracy phản ánh độ chính xác chung của bộ phân lớp dữ liệu..

(4) Độ đo F-Measure

- Định nghĩa: F-Measure = 2.(Precision.Recall) / (Precision + Recall).

- Ý nghĩa: F-Measure là độ đo nhằm đánh giá độ chính xác thông qua quá trình kiểm chứng dựa trên sự xem xét đến hai độ đo là Precision và Recall. Giá trị F-Measure càng cao phản ánh độ chính xác càng cao của bộ phân lớp dữ liệu. Có thể coi độ đo F-Measure là trung bình điều hoà của hai độ đo Precision và Recall.

(5) Độ đo Specitivity (Độ đặc hiệu)

- Định nghĩa: Specitivity = TN/(TN+FP).

- Ý nghĩa: Độ đo Specitivity đánh giá khả năng một dữ liệu là phần tử âm được bộ phân lớp cho ra kết quả chính xác.

1.2. Tổng quan về học máy

1.2.1. Khái niệm về học máy

Học máy là một lĩnh vực của trí tuệ nhân tạo liên quan đến việc nghiên cứu và xây dựng các kĩ thuật cho phép các hệ thống "học" tự động từ dữ liệu để giải quyết những vấn đề cụ thể .

1.2.2 Phân loại các loại học máy

- Học có giám sát

- Học không giám sát
- Học bán giám sát

1.3. Thuật toán Cây quyết định

1.3.1. Giới thiệu phương pháp

Cây quyết định là một mô hình cấu trúc cây giống như một lưu đồ mà trong đó mỗi nút bên trong cây diễn tả cho việc kiểm tra một thuộc tính, mỗi nhánh trên cây sẽ đại diện cho một kết quả của quá trình kiểm tra và các nút lá sẽ đại diện cho các lớp hoặc phân phối lớp. Nút trên cùng sẽ là nút gốc. Quá trình xây dựng cây quyết định được thực hiện bằng việc phân tách các dữ liệu trong một nút, chia chúng thành các nút con. Quá trình tương tự được áp dụng cho từng các nút con một cách đệ quy cho đến khi không còn nút con nào có thể được tách ra nữa. Các nút không thể được chia nhỏ hơn nữa sẽ được phát triển thành các nút lá.

Quá trình xây dựng một cây quyết định thường được thực hiện như sau:

(1) Bắt đầu từ nút gốc nơi biểu diễn tất cả các mẫu của tập dữ liệu.

(2) Nếu tất cả các mẫu thuộc về cùng một lớp, nút đang xét sẽ trở thành nút lá và được gán nhãn chính bằng lớp đó.

(3) Ngược lại, dùng độ đo thuộc tính nào đó để chọn thuộc tính sẽ phân tách các mẫu tốt nhất vào các lớp tương ứng.

(4) Một nhánh được tạo ra cho từng giá trị của thuộc tính được chọn.

(5) Lặp lại quá trình trên để tạo cây quyết định.

(6) Tiến trình kết thúc chỉ khi bất kỳ điều kiện nào sau đây là đúng:

- Tất cả các mẫu của một nút cho trước đều thuộc về cùng một lớp.
- Không còn thuộc tính nào mà mẫu có thể dựa vào để phân hoạch xa hơn.
- Không còn mẫu nào cho nhánh.

Tuy nhiên, nếu chúng ta không lựa chọn được thuộc tính nào để phân loại hợp lý tại mỗi nút, cây quyết định sau khi xây dựng có thể rất phức tạp. Vì thế người ta thường sử dụng hai cách sau để xây dựng cây quyết định phù hợp:

- Dừng việc phát triển cây sớm hơn bình thường trước khi phân lớp hoàn toàn tập dữ liệu huấn luyện.

- Sử dụng một số kỹ thuật "cắt", "tỉa" cây phù hợp.

1.3.2. Thuật toán Rừng ngẫu nhiên

Rừng ngẫu nhiên là một thuật toán học có giám sát. Như bạn có thể thấy từ tên của nó, nó tạo ra một khu rừng một cách ngẫu nhiên. "Khu rừng" mà ta tạo ra là một tập hợp các cây quyết định. Ý tưởng chính của phương pháp là sự kết hợp của các mô hình học tập làm tăng kết quả chung.

1.4. Bug Tracking System

Theo dõi lỗi là một quá trình được sử dụng bởi các nhân viên đảm bảo chất lượng và lập trình viên để theo dõi các vấn đề phần mềm và phần cứng. Một hệ thống theo dõi lỗi thường đưa ra để lưu trữ thông tin về lỗi đã thông báo.

Hệ thống theo dõi lỗi (BTS) là một hệ thống kiểm tra sự cố đặc biệt được sử dụng để theo dõi các lỗi phần mềm. Các trường (thuộc tính) được sử dụng để theo dõi trạng thái của vấn đề trong khi mô tả bằng văn bản được sử dụng để mô tả vấn đề. Các BTS nói chung nhằm mục đích nâng cao chất lượng của các sản phẩm phần mềm.

Có nhiều hệ thống BTS khác nhau, mỗi hệ thống sẽ tập trung vào một hoặc nhiều thuộc tính đặc trưng của dữ liệu mà hệ thống đó hướng đến. Một trong những tính năng quan trọng nhất của BTS là khả năng cho phép truy xuất dữ liệu. Tất cả BTS tối thiểu đều phải có giao diện Web thông qua HTML, nhưng việc hỗ trợ truy xuất tự động sẽ thuận tiện hơn nhiều. Thông tin cung cấp từ BTS thường dưới dạng text, trong khi một số BTS cung cấp thông tin dưới dạng đồ thị thông qua một số dạng ký tự đặc biệt. Một số BTS không cung cấp thông tin với mô tả cụ thể mà dưới dạng data model, đòi hỏi người dùng phải chuyển đổi sang dạng có thể sử dụng.

Việc theo dõi mối quan hệ phụ thuộc giữa các báo cáo lỗi rất quan trọng, đôi khi một lỗi liên quan đến lỗi khác không thể giải quyết nếu mối liên hệ với lỗi đó chưa được giải quyết hoặc tác động. Một số hệ thống BTS cho phép tìm kiếm theo các từ khóa, trong khi một số khác phải sử dụng bộ lọc có sẵn để tìm kiếm dữ liệu lỗi. Hệ thống cho phép tìm kiếm theo từ khóa thì thuận tiện hơn cho cho một hệ thống tự động.

Các cách truy xuất dữ liệu từ BTS:

- Dùng API.
- Truy xuất thông qua công cụ để lấy nội dung từ HTML. Truy xuất từ HTML có một vấn đề cần giải quyết đó là có thể với cùng một cấu trúc nhưng dữ liệu có thể được thể hiện với các giao diện khác nhau. Tuy nhiên tất cả báo cáo lỗi từ một nguồn BTS sẽ có cấu trúc giống nhau, nếu một thuật toán truy xuất được 1 báo cáo lỗi thì có thể truy xuất tất cả các báo cáo lỗi còn lại.

1.5. Thư viện Scikit-learn

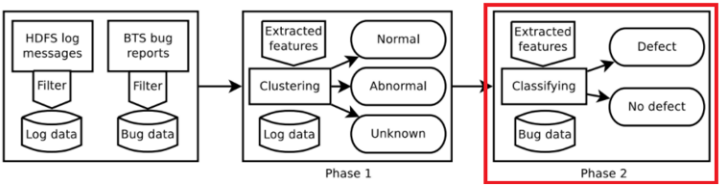
Là một thư viện mạnh mẽ có thể mang các thuật toán học máy (machine learning) vào trong một hệ thống thích hợp nhất. Thư viện này tích hợp rất nhiều thuật toán hiện đại và cố điển hỗ trợ việc học và tiến hành đưa ra các giải pháp hữu ích cho bài toán học máy một cách đơn giản.

Scikit-learn (Sklearn) là thư viện mạnh mẽ nhất dành cho các thuật toán học máy được viết trên ngôn ngữ Python.

Chương 2 – PHƯƠNG PHÁP PHÂN LOẠI LỖI MẠNG

2.1. Mô hình Two-Phase Defect Detection

Quá trình phân loại lỗi mạng thông qua mô hình Two-Phase Defect Detection được thể hiện qua mô hình như hình bên dưới.



Hình 2.1: Mô hình Two-Phase Defect Detection

Trong mô hình trên, dữ liệu log sẽ được thu thập từ các thiết bị trên một hệ thống (ở đây là HDFS logs) [6], sau đó dữ liệu log sẽ được phân loại dựa vào phương pháp gom cụm để chia các dữ liệu log thành 3 loại chính:

- Bình thường
- Bất bình thường
- Không xác định

Lỗi được thu thập từ các BTS thông qua công cụ thu thập dữ liệu từ giao diện HTML. Các lỗi sau đó được lưu

thành một Bug Database với một định dạng chuẩn hóa duy nhất. Tập dữ liệu lỗi sẽ được sử dụng để huấn luyện máy học thông qua thuật toán Rừng ngẫu nhiên và kỹ thuật tf x idf để phân loại thành các dạng lỗi nhỏ, lỗi thông thường và lỗi đặc biệt nghiêm trọng.

Các dữ liệu log file bất bình thường sau khi được gom cụm tại Phase 1 sẽ được đưa vào mô hình phân loại lỗi như một đầu vào, từ đó phân loại lỗi trên có ảnh hưởng đến hệ thống hay không dưới dạng 3 lớp như sau:

- Lỗi thông thường
- Lỗi nhỏ
- Lỗi nghiêm trọng

Mô hình phân loại lỗi trong luận văn chính là Phase 2 của mô hình **Two-Phase Defect Detection.**

2.2. Mô hình dữ liệu lỗi

Để tích hợp các dữ liệu lỗi từ nhiều nguồn BTS khác nhau ta phải đưa các dữ liệu lỗi trên về cùng một mô hình thống nhất cho các lỗi. Các lỗi trên các nền tảng BTS khác

nhau cũng có nhiều trường thuộc tính dữ liệu tương tự và có thể chia làm hai nhóm chính sau đây:

Trường dữ liệu quản trị thường được biểu diễn dưới dạng ngữ nghĩa rất chính xác như: ID, mức độ nghiêm trọng, báo cáo, tóm tắt.

Các mô tả chi tiết về lỗi hoặc các thảo luận về lỗi thường được thể hiện dưới dạng tệp đính kèm hoặc văn bản tự do.

Một số thuộc tính đặc trưng của lỗi như:

+ ID: để xác định duy nhất một phiên bản lỗi.

+ Mức độ nghiêm trọng: hầu hết các BTS đều có phân loại mức độ nghiêm trọng của lỗi tuy nhiên nó là khác nhau giữa các BTS, việc cần làm là đồng bộ các giá trị từ các BTS khác nhau về cùng một loại cho thuộc tính trên.

+ Thuộc tính trạng thái của một lỗi: là lỗi mới hay lỗi đã được giải quyết, xác minh và đóng.

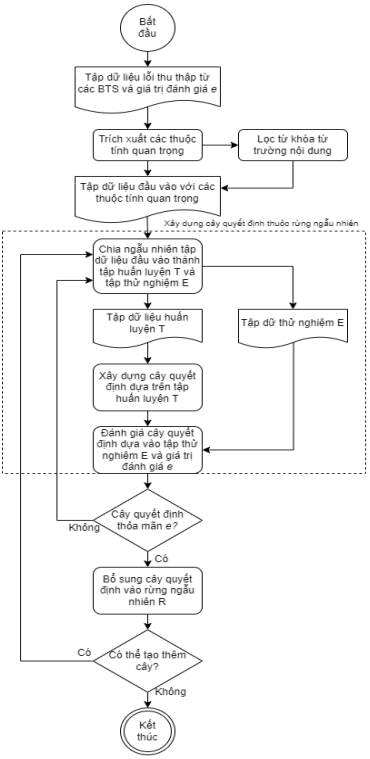
+ Các mô tả dạng văn bản được mô hình hóa dưới dạng tệp đính kèm, mỗi tệp đính kèm thuộc về chính xác một lỗi nào đó. Phần này có thể có các thuộc tính như loại phức tạp, kiểu sự cố, phân vùng lỗi…

2.3. Sử dụng phương pháp tf x idf để lọc nội dung quan trọng từ nội dung mô tả lỗi

TF-IDF (Term Frequency – Inverse Document Frequency) là 1 kĩ thuật sử dụng trong khai phá dữ liệu văn bản. Trọng số này được sử dụng để đánh giá tầm quan trọng của một từ trong một văn bản. Giá trị cao thể hiện độ quan trọng cao và nó phụ thuộc vào số lần từ xuất hiện trong văn bản nhưng bù lại bởi tần suất của từ đó trong tập dữ liệu.

Chương 3 - XÂY DỰNG MÔ HÌNH PHÂN LOẠI LỖI MẠNG

Quá trình để xây dựng rừng ngẫu nhiên cho tập dữ liệu lỗi mạng được biểu diễn qua lưu đồ giải thuật như sau.



Hình 3.1: Lưu đồ giải thuật xây dựng rừng ngẫu nhiên

3.1. Tập dữ liệu lỗi thu thập từ các BTS

Đối với đề xuất hiện tại của luận văn, dữ liệu lỗi được lấy từ nguồn là các báo cáo lỗi trên các hệ thống Bugtracking System. Đầu vào ở dạng thô bao gồm các thông tin của một lỗi trên một hệ thống BTS như thời gian báo cáo, người báo cáo, nội dung lỗi và các trường thông tin khác. Từ dữ liệu Bugs trên, các file bugs nhận được là dữ liệu thô được trích xuất từ giao diện HTML, ta cần phân loại, định nghĩa lại và đồng bộ các trường, các trạng thái, nội dung đồng nhất để hệ thống xử lý. Một vài số liệu thống kê về thông tin lỗi như sau:

- Số lượng bug files: 5
- Tổng dung lượng: 0.77 GB
- Số lượng bug reports: 483000
- Số lượng gentoo bug report: 150000
- Số lượng redhat bug report: 83000
- Số lượng mozilla bug report: 250000

Dữ liệu lỗi sau khi thi thập từ các hệ thống được lưu dưới dạng .csv như ảnh minh họa.

Dữ liệu lỗi sau khi thu thập từ các BTS cần thực hiện các bước tiền xử lý để loại bỏ các mẫu nhiễu trong tập dữ liệu như các dòng trống, các dòng không có giá trị.

3.2. Trích xuất thuộc tính quan trọng của lỗi

Các lỗi từ tập dữ liệu lỗi được trích xuất để lấy các thuộc tính quan trọng với quá trình phân loại lỗi, các lỗi được trích xuất cụ thể như sau:

	State	Content	Component	Software	Platform	Relation	Severity
0	fixed	event-dialog needs to work with the actual cal...	Internal Compone	Calendar	win	296893.0	feature
1	fixed	Wildcards in install locations list not recogn...	Add-ons Manager	Toolkit	win	259006.0	feature
2	fixed	Convert xmlextras tests to use the glue	DOM: Mozilla Ext	Core	all	305949.0	feature
3	fixed	Unable to view deleted comments [as developer]	Developer Pages	addons.mozilla.o	all	NaN	feature
4	fixed	Freeze upon connex to many URLs, & enterin...	General	Firefox	mac	319747.0	critical
...	...	...	...	...	...	...	...
95	fixed	E-Mail Synchronisation with Palm	Palm Sync	MailNews Core Gr	all	36836.0	feature
96	fixed	File - Save As - Template menu item does nothi...	Mail Window Fron	Thunderbird	all	331876.0	feature
97	fixed	In <prefs.js>, "Warning: reference ...	ChatZilla	Other Applicatio	all	NaN	feature
98	fixed	Message search dialog title is "Find in T...	Mail Window Fron	Thunderbird	all	328795.0	feature
99	open	Updates to the content does not update the scr...	Layout: View Ren	Core	win	NaN	normal

Hình 3.3: Dữ liệu lỗi sau khi Import

Thuộc tính từ khóa với giá trị là từ có giá trị tf x idf cao nhất được bổ sung vào tập dữ liệu để làm đầu vào cho thuật toán Rừng ngẫu nhiên.

3.3. Xây dựng rừng ngẫu nhiên

3.3.1. Chuẩn hóa dữ liệu sang dạng số

Dữ liệu được chuẩn hóa sang kiểu dữ liệu dạng số để phù hợp với mô hình thuật toán Rừng ngẫu nhiên. Gọi n_c là số phần tử riêng biệt của thuộc tính c trong tập huấn luyện. Mỗi giá trị của thuộc tính c được gán nhãn từ 0 đến $n_c - 1$.

Ví dụ như trường thuộc tính trạng thái với hai trạng thái là mở và đóng được chuyển về dạng số dưới dạng giá trị 0 (ứng với trạng thái đóng) và 1 (ứng với trạng thái mở).

Một lỗi với các thuộc tính quan trọng sau khi được trích xuất và chuẩn hóa được chuyển về dạng véc tơ để làm đầu vào xây dựng Rừng ngẫu nhiên với đầu vào và đầu ra như sau:

Đầu vào: tập dataset Y là tập dữ liệu lỗi với các thuộc tính trạng thái, thành phần xảy ra lỗi, phần mềm, nền tảng, mối liên hệ, từ khóa, mức độ ưu tiên và giá trị ngưỡng đánh giá σ (giá trị trong luận văn sử dụng là giá trị F1 score và điều kiện là $\sigma > 0.8$).

Đầu ra: Rừng ngẫu nhiên với tập hợp các cây quyết định tối ưu R.

3.3.2. Lấy mẫu dữ liệu cho việc xây dựng cây quyết định

Chia tập dataset Y ngẫu nhiên thành 2 phần: tập dữ liệu để kiểm tra E (20%) và tập dữ liệu để huấn luyện T (80%). Sử dụng T để xây dựng Rừng ngẫu nhiên. Dữ liệu huấn luyện T với các thuộc tính trạng thái, thành phần xảy ra lỗi, phần mềm, nền tảng, mối liên hệ, từ khóa để xây dựng mô hình phân lớp. Tổng cộng có 6 thuộc tính dùng cho việc phân lớp dữ liệu. Giả sử có 1000 mẫu dữ liệu lỗi, 800 mẫu sẽ được dùng để xây dựng một cây quyết định, 200 mẫu lỗi còn lại dùng để đánh giá mô hình cây quyết định xây dựng được có thỏa mãn giá trị đánh giá σ hay không. Nếu thỏa mãn cây vừa xây dựng sẽ được thêm vào tập cây của rừng ngẫu nhiên.

3.3.3 Xây dựng cây quyết định

Quá trình tạo cây quyết định với từng tập dữ liệu sau quá trình lấy mẫu ngẫu nhiên ở bước trước được thực hiện như sau: Quá trình xây dựng cây bắt đầu bằng việc khởi động nút gốc thành nút nhị phân. Ban đầu, tất cả các mẫu dữ liệu là nằm trong nút gốc. CART triển khai một tổ hợp các thuật toán chuyên sâu tìm kiếm cách phân chia tốt nhất ở tất cả các điểm phân chia có thể có cho mỗi biến. Các phương pháp mà CART sử dụng để xây dựng cây

quyết định được gọi là phân vùng đệ quy nhị phân. Thông qua chỉ số Gini như một quy tắc tách.

CART dừng quá trình phân tách khi:

- Chỉ có một mẫu trong mỗi nút.

- Tất cả các mẫu trong mỗi nút con có phân loại giống hệt nhau của các biến phân lớp, tức là không thể tách.

- Cây đạt đủ độ sâu theo cài đặt (max_depth=10).

Một số ví dụ về cây quyết định được xây dựng với giá trị max_depth=3 được thể hiện qua các hình minh họa như sau:

1.3.4. Xây dựng rừng ngẫu nhiên

Cây quyết định sau khi xây dựng xong nếu thỏa giá trị đánh giá ban đầu sẽ được cập nhật vào rừng ngẫu nhiên R. Việc đánh giá dựa trên tập thử nghiệm E và giá trị đánh giá σ. Độ chính xác của cây quyết định, cụ thể là giá trị F1 Score của cây sẽ được tính dựa vào kết quả phân loại tập thử nghiệm E, nếu F1 Score > 0.8 thì cây sẽ được đưa vào rừng ngẫu nhiên và ngược lại.

Toàn bộ quá trình lấy mẫu ngẫu nhiên dữ liệu và xây dựng cây quyết định được thực hiện lặp lại cho đến khi không thể tạo thêm cây mới hoặc rừng ngẫu nhiên đạt đủ số lượng cây theo cài đặt ban đầu. Sau khi kết thúc ra sẽ thu được một mô hình rừng ngẫu nhiên gồm nhiều cây quyết định tối ưu được xây dựng trên tập mẫu được lấy ngẫu nhiên từ tập dataset ban đầu.

Chương 4 – PHÂN TÍCH VÀ ĐÁNH GIÁ

4.1. Phân tích độ chính xác của mô hình

Để đánh giá độ chính xác của mô hình đã xây dựng ta dựa vào kết quả phân lớp tập thử nghiệm là tập dữ liệu được trích xuất từ tập dataset ban đầu với 1000 mẫu lỗi được xây dựng thủ công, trường thuộc tích từ khóa được chọn lọc từ nội dung lỗi để lấy từ liên quan đến lỗi đó nhất.

Giá trị Recall Score đạt 89.50% chứng tỏ tỉ lệ lỗi ảnh hưởng đến hệ thống được phân loại thành không ảnh hưởng đến hệ thống vẫn còn cao, điều này có thể gây ra việc bỏ qua các lỗi đặc biệt nghiêm trọng của hệ thống.

Giá trị Precision Score đạt 85.65%, chứng tỏ tỉ lệ lỗi không ảnh hưởng đến hệ thống bị phân loại thành có ảnh hưởng đến hệ thống cao hơn trường hợp ngược lại, điều này có nghĩa cảnh báo giả cũng sẽ xuất hiện nhiều hơn với 30 cảnh báo giả được dự báo trên 1000 mẫu lỗi được xử lý chiến tỉ lệ 3%.

Giá trị F1 Score đạt 87.53%, giá trị trên với mô hình phát hiện lỗi sử dụng rừng ngẫu nhiên cho kết quả ở mức chấp nhận được.

Để đánh giá mức độ ảnh hưởng của các tham số quan trọng trong quá trình xây dựng rừng ngẫu nhiên là max_depth (độ sâu của cây quyết định) và n_ estimators (số lượng cây quyết định trong rừng ngẫu nhiên), các thử nghiệm được thực hiện 10 lần và các kết quả thu được dựa trên giá trị trung bình của các lần chạy như bảng kết quả đánh giá sau.

Bảng 4.2: Giá trị F1 Score với hai tham số quan trọng của rừng ngẫu nhiên

max_depth\n_estimators	20	30	50	100	200
5	87.38	87.38	87.39	87.34	87.39
10	87.4	87.41	87.39	87.39	87.39
15	87.18	87.37	87.37	87.39	87.38
20	86.78	86.65	86.77	86.81	86.87
25	85.81	85.75	85.9	85.97	86.03

Dựa vào các chỉ số đánh giá độ chính xác của mô hình trên có thể đưa ra các nhận định sau:

- Mô hình cho độ chính xác cao nhất với giá trị max_depth = 10, giá trị n_estimators = 30 với các giá trị F1 Score = 87.41%.

Chương 5 - KẾT LUẬN

5.1. Kết quả đạt được

5.1.1. Về mặt lý thuyết

Khai thác được mô hình dữ liệu lỗi để xây dựng mô hình phát hiện lỗi mạng.

Ứng dụng Trí tuệ nhân tạo (AI), Machine Learning, các thuật toán học máy và phương pháp khai phá dữ liệu văn bản vào việc phát hiện lỗi mạng.

Khai thác được các thuật toán phân lớp dữ liệu, cụ thể là mô hình cây quyết định và rừng ngẫu nhiên. Nắm bắt được quá trình xây dựng một cây quyết định dựa trên giá trị gini index hay entropy và quá trình xây dựng một rừng ngẫu nhiên dựa trên các cây quyết định.

Ứng dụng thư viện scikit-learn trên nền tảng python vào việc nghiên cứu các vấn đề học máy, sử dụng được các tham số để tối ưu mô hình rừng ngẫu nhiên xây dựng được.

5.1.2. Về mặt thực tiễn

Luận văn đã đưa ra được giải pháp phân loại lỗi và cảnh báo sớm sự cố có thể ảnh hưởng đến một hệ thống

mạng dựa vào việc phân tích lỗi từ các hệ thống báo cáo lỗi hiện có. Việc này sẽ là tiền đề để xây dựng một công cụ phát hiện lỗi mạng ảnh hưởng đến một hệ thống mạng và truyền thông trong tương lai, thay thế người vận hành với vai trò là một chuyên gia nhận định lỗi và đưa ra dự báo về mức độ nghiêm trọng của một lỗi.

Mô hình trên có thể hỗ trợ người dùng trên các hệ thống BTS, khi người dùng báo cáo một lỗi mới lên hệ thống sẽ có thể nhận định sớm từ ban đầu mức độ ảnh hưởng của lỗi trên đến hệ thống của họ.

5.2. Hạn chế

Kết quả phân loại lỗi đạt được chỉ ở mức tốt chứ chưa thật sự cao. Kết quả đạt được chưa bao quát được hết các trường hợp. Dữ liệu mẫu cần được training và mở rộng môi trường áp dụng.

Các trường hợp lỗi nghiêm trọng bị phân loại sai thành không nghiêm trọng còn nhiều, gây nhầm lẫn cho người sử dụng nếu áp dụng với thực tế.

Mô hình rừng ngẫu nhiên trong luận văn còn ở mức cơ bản, chưa phân tích sâu vào các tham số để phù hợp với mô hình dữ liệu lỗi.

Việc phân tích nội dung lỗi bằng phương pháp tf x idf còn nhiều thiếu sót, chưa đủ để trích xuất được nội dung quan trọng với việc phân loại lỗi từ nội dung mô tả.

5.3. Hướng phát triển

Áp dụng kết hợp phương pháp phân tích ngữ nghĩa LSI/LSA kết hợp với phương pháp tf x idf trong việc trích xuất từ khóa từ nội dung lỗi để mang lại kết quả có giá trị hơn đối với việc phân loại lỗi.

Tiến hành áp dụng cho cho hệ thống mạng lưới mạng di động của Viễn thông Tây Ninh. Cảnh báo sớm các lỗi nghiêm trọng có thể xảy ra dựa trên cơ sở dữ liệu là các lỗi và mức độ ảnh hưởng của chúng trong quá khứ. Trích xuất các nội dung ghi chú của các lỗi để tìm ra đặc trưng của một lỗi nghiêm trọng trên hệ thống mạng di động thuộc Viễn thông Tây Ninh góp phần cảnh báo sớm các sự cố ảnh hưởng nghiêm trọng đến hệ thống.