**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**---------------------------------------**



**NGUYỄN THỊ HIỀN**

**ĐÁNH GIÁ HIỆU NĂNG MỘT SỐ THUẬT TOÁN**

**PHÂN LỚP CHO PHÁT HIỆN XÂM NHẬP**

**LUẬN VĂN THẠC SĨ KỸ THUẬT**

HÀ NỘI – 2017

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**---------------------------------------**



**NGUYỄN THỊ HIỀN**

**ĐÁNH GIÁ HIỆU NĂNG MỘT SỐ THUẬT TOÁN**

**PHÂN LỚP CHO PHÁT HIỆN XÂM NHẬP**

|  |  |  |
| --- | --- | --- |
| **CHUYÊN NGÀNH** | **:** | **KHOA HỌC MÁY TÍNH** |
| **MÃ SỐ** | **:** | **60.48.01.01** |

**LUẬN VĂN THẠC SĨ KỸ THUẬT**

NGƯỜI HƯỚNG DẪN KHOA HỌC

**TS. PHẠM HOÀNG DUY**

HÀ NỘI - 2017

LỜI CAM ĐOAN

Luận văn này là thành quả của quá trình học tập nghiên cứu của em cùng sự giúp đỡ, khuyến khích của các quý thầy cô sau 2 năm em theo học chương trình đào tạo Thạc sĩ, chuyên ngành Khoa học máy tính của trường Học viện Công nghệ Bưu chính Viễn thông.

Em cam đoan đây là công trình nghiên cứu của riêng em. Nội dung của luận văn có tham khảo và sử dụng một số thông tin, tài liệu từ các nguồn sách, tạp chí được liệt kê trong danh mục các tài liệu tham khảo và được trích dẫn hợp pháp.

|  |  |
| --- | --- |
|  | Tác giả  *(Ký và ghi rõ họ tên)*  **Nguyễn Thị Hiền** |

LỜI CÁM ƠN

Em xin gửi lời cảm ơn và tri ân tới các thầy cô giáo, cán bộ của Học viện Công nghệ Bưu chính Viễn thông đã giúp đỡ, tạo điều kiện tốt cho em trong quá trình học tập và nghiên cứu chương trình Thạc sĩ.

Em xin gửi lời cảm ơn sâu sắc tới **TS. Phạm Hoàng Duy** đã tận tình hướng dẫn, giúp đỡ và động viên em để hoàn thành tốt nhất Luận văn “ĐÁNH GIÁ HIỆU NĂNG MỘT SỐ THUẬT TOÁN PHÂN LỚP CHO PHÁT HIỆN XÂM NHẬP“.

Do vốn kiến thức lý luận và kinh nghiệm thực tiễn còn ít nên luận văn không tránh khỏi những thiếu sót nhất định. Em xin trân trọng tiếp thu các ý kiến của các thầy, cô để luận văn được hoàn thiện

Trân trọng cám ơn.

|  |  |
| --- | --- |
|  | Tác giả  *(Ký và ghi rõ họ tên)*  **Nguyễn Thị Hiền** |

MỤC LỤC

[MỞ ĐẦU 1](#_Toc486505408)

[Chương 1 - TỔNG QUAN VỀ PHÁT HIỆN XÂM NHẬP 3](#_Toc486505409)

[1.1. Xâm nhập mạng là gì. 3](#_Toc486505410)

[1.2. Một số kiểu tấn công phổ biến. 3](#_Toc486505411)

[*1.2.1.* *Tấn công từ chối dịch vụ* 3](#_Toc486505412)

[*1.2.2.* *Tấn công thăm dò* 4](#_Toc486505413)

[*1.2.3.* *Tấn công chiếm quyền root* 4](#_Toc486505414)

[*1.2.4.* *Tấn công điều khiển từ xa* 5](#_Toc486505415)

[1.3. Một số giải pháp ngăn chặn xâm nhập truyền thống. 5](#_Toc486505416)

[*1.3.1.* *Tường lửa.* 5](#_Toc486505417)

[*1.3.2.* *Mã hóa dữ liệu.* 6](#_Toc486505418)

[*1.3.3.* *Xác thực.* 6](#_Toc486505419)

[*1.3.4.* *Quyền truy cập* 6](#_Toc486505420)

[1.4. Hệ thống phát hiện xâm nhập (instrucsion detection system). 7](#_Toc486505421)

[1.5. Phân loại hệ thống phát hiện xâm nhập mạng. 8](#_Toc486505422)

[Chương 2 - MỘT SỐ THUẬT TOÁN PHÂN LỚP CHO PHÁT HIỆN XÂM NHẬP 13](#_Toc486505423)

[2.1. Học cây quyết định. 13](#_Toc486505424)

[*2.1.1.* *Xây dựng cây quyết định.* 14](#_Toc486505425)

[*2.1.2.* *Xây dựng cây quyết định dựa vào Entropy.* 15](#_Toc486505426)

[2.2. Máy vectơ hỗ trợ (support vector machine-SVM). 16](#_Toc486505427)

[*2.2.1.* *Ý tưởng thuật toán.* 16](#_Toc486505428)

[*2.2.2.* *Cơ sở lý thuyết.* 17](#_Toc486505429)

[*2.2.3.* *Một số ứng dụng tiêu biểu của SVM.* 20](#_Toc486505430)

[2.3. Mạng neural nhân tạo (Artificial Neural Networks)-ANN. 20](#_Toc486505431)

[*2.3.1.* *Kiến trúc của một mạng neural nhân tạo.* 20](#_Toc486505432)

[*2.3.2.* *Quá trình học của ANN* 24](#_Toc486505433)

[*2.3.3.* *Ứng dụng của mạng neural nhân tạo* 25](#_Toc486505434)

[Chương 3 – THỰC NGHIỆM VÀ ĐÁNH GIÁ 26](#_Toc486505435)

[3.1. Một số độ đo đánh giá 26](#_Toc486505436)

[*3.1.1.* *Các độ đo đánh giá* 26](#_Toc486505437)

[*3.1.2.* *Cách đánh giá* 27](#_Toc486505438)

[3.2. Giới thiệu bộ dữ liệu NSL- KDD 29](#_Toc486505439)

[*3.2.1.* *Giới thiệu.* 29](#_Toc486505440)

[*3.2.2.* *Mô tả tập dữ liệu NSL KDD.* 29](#_Toc486505441)

[*3.2.3.* *Các tấn công trong tập dữ liệu NSL- KDD.* 35](#_Toc486505442)

[3.3. Công cụ thử nghiệm và chuẩn dữ liệu đầu vào 36](#_Toc486505443)

[*3.3.1.* *Giới thiệu công cụ Weka.* 36](#_Toc486505444)

[*3.3.2.* *Chuẩn dữ liệu đầu vào.* 38](#_Toc486505445)

[3.4. Thực nghiệm và đánh giá 39](#_Toc486505446)

[*3.4.1.* *Thực nghiệm.* 39](#_Toc486505447)

[*3.4.2.* *Kết quả* 41](#_Toc486505448)

[*3.4.3.* *Đánh giá.* 54](#_Toc486505450)

[KẾT LUẬN 56](#_Toc486505451)

[DANH MỤC TÀI LIỆU THAM KHẢO 57](#_Toc486505452)

DANH MỤC TỪ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| **Từ viết tắt** | **Tiếng anh** | **Tiếng việt** |
| SVM | Support Vector Machine | Máy vector hỗ trợ |
| ANN | Artificial Neural Network | Mạng neural nhân tạo |
| U2R | User to root | Tấn công chiếm quyền root |
| R2L | Remote to Local | Tấn công điều khiển từ xa |
| DoS | Denial of Service | Tấn công từ chối dịch vụ |
| HIDS | Host Intrusion Detection System | Hệ thống phát hiện xâm nhập cho máy trạm |
| IDS | Intrusion Detection System | Hệ thống phát hiện xâm nhập |
| KDD | Knowledge Discovery and Data Mining | Phát hiện tri thức và khai phá dữ liệu |
| NIDS | Network Intrusion Detection System | Hệ thống phát hiện xâm nhập mạng |
| Weka | Waikato Environment for Knowledge Acquisition | Học máy |

DANH MỤC CÁC BẢNG BIỂU

[Bảng 3. 1: Danh sách và mô tả tập file NSL- KDD 29](#_Toc486493006)

[Bảng 3. 2: Thông tin chi tiết các bản ghi trong các tập dữ liệu 30](#_Toc486493007)

[Bảng 3. 3: Mô tả các thuộc tính mỗi bản ghi 31](#_Toc486493008)

[Bảng 3. 4: Các tấn công trong tập dữ liệu NSL-KDD 35](#_Toc486493009)

[Bảng 3. 5: Tóm tắt thao tác thự hiện thử nghiệm 40](#_Toc486493010)

[Bảng 3. 6: Tóm tắt kết quả các thuật toán trên tập train 51](#_Toc486493011)

[Bảng 3. 7: Tóm tắt kết quả các thuật toán trên tập kiểm thử 52](#_Toc486493012)

DANH MỤC CÁC HÌNH VẼ

[Hình 1.1: Vị trí của hệ thống phát hiện xâm nhập (IDS) [5] 8](#_Toc486492058)

[Hình 1.2: Hệ thống phát hiện xâm nhập NIDS [5] 10](#_Toc486492059)

[Hình 1.3: Hệ thống phát hiện xâm nhập HIDS [5] 10](#_Toc486492060)

[Hình 2.1: Mô tả chung về cây quyết định 13](#_Toc486492285)

[Hình 2.2: Mô tả phương pháp SVM 17](#_Toc486492286)

[Hình 2.3: Kiến trúc chung của một mạng neural nhân tạo 21](#_Toc486492287)

[Hình 2.4: Mô hình mạng neural truyền thẳng [4] 22](#_Toc486492288)

[Hình 2.5: Mô hình mạng hồi quy[4] 23](#_Toc486492289)

[Hình 2.6: Hoạt động của 1 ANN 25](#_Toc486492290)

[Hình 3.1: Đánh giá cross- validation k fold 28](#_Toc486492726)

[Hình 3.2: Phân bố các bản ghi [1] 31](#_Toc486492727)

[Hình 3.3: Giao diện màn hình chính của Weka. 36](#_Toc486492728)

[Hình 3.4: Hình Môi trường explorer khi nạp một tập dữ liệu 37](#_Toc486492729)

[Hình 3.5: Kiểm thử mô hình 41](#_Toc486492730)

[Hình 3.6: Phân lớp với Randomforest 41](#_Toc486492731)

[Hình 3.7: Phân lớp với j48 43](#_Toc486492732)

[Hình 3.8: Một phần cây quyết định J48 44](#_Toc486492733)

[Hình 3.9: Kết quả kiểm thử phân lớp với radomforest 47](#_Toc486492734)

[Hình 3.10: Phân lớp với MLP 48](#_Toc486492735)

[Hình 3.11: Kết quả kiểm thử phân lớp với SMO puk 49](#_Toc486492736)

[Hình 3.12: Kết quả kiểm thử phân lớp với SMO-polykenel 50](#_Toc486492737)

[Hình 3.13: Mạng neural nhân tạo 50](#_Toc486492738)

[Hình 3.14: Độ chính xác 53](#_Toc486492739)

[Hình 3.15: PRECISION 54](#_Toc486492740)

MỞ ĐẦU

Ngày nay, công nghệ thông tin đã và đang được ứng dụng sâu rộng trong mọi lĩnh vực kinh tế - xã hội … Đây là công cụ không thể thiếu cho việc phát triển thương mại, vận hành hệ thống trong các tổ chức, doanh nghiệp và phục vụ nhu cầu công việc, học tập, giải trí cá nhân. Mạng máy tính và Internet đóng vai trò hết sức quan trọng và quyết định sự tồn tại của các tổ chức, doanh nghiệp. Chúng ta có thể thấy mạng là cơ sở hạ tầng cho tất cả các dịch vụ, ứng dụng, đang chi phối hầu hết các lĩnh vực của cuộc sống từ kinh tế, giải trí đến giáo dục và đào tạo… Vì vậy vấn đề an toàn thông tin khi sử dụng mạng là vấn đề lớn cần được quan tâm và nghiên cứu cho cả các tổ chức và cá nhân. Tuy nhiên bên cạnh những tiến bộ và lợi ích mang lại, thì có khá nhiều rủi ro xuất phát từ các cuộc tấn công mạng trên môi trường internet. Do đó nhiều hệ thống khác nhau đã được thiết kế và xây dựng ngăn chặn các cuộc tấn công này để bảo đảm tính an toàn của hệ thống máy tính như: Hệ thống phát hiện xâm nhập (Intrusion Detection Systym- IDS). Mục tiêu IDS là để giúp các hệ thống mạng có khả năng phản ứng tích cực với các cuộc tấn công. Việc phân tích phát hiện và cảnh báo sớm các cuộc tấn công, giảm cảnh báo sai và giảm chi phí tính toán có vai trò quan trọng để nâng cao hiệu quả phát hiện của IDS. Trong thời gian qua nhiều hệ thống phát hiện xâm nhập được phát triển. Các nhà nghiên cứu đã quan tâm đánh giá hiệu quả của nó nhưng việc đánh giá này là khá khó khăn do thứ nhất việc thu dữ liệu mạng chất lượng cao để thực hiện việc đánh giá. Thứ hai, ngay cả khi dữ liệu thực tế là có sẵn, ghi nhãn các kết nối mạng như bình thường hoặc xâm nhập đòi hỏi rất lớn nhiều thời gian. Thứ ba, sự thay đổi liên tục của mạng, lưu lượng truy cập,biến thể của các cuộc tấn công

Phát hiện xâm nhập dựa trên bất thường dựa trên giả thiết các hành vi tấn công, xâm nhập thường có quan hệ chặt chẽ với các hành vi bất thường. Phát hiện xâm nhập dựa trên bất thường có nhiều triển vọng do nó có khả năng phát hiện các tấn công, xâm nhập mới. Việc phát hiện bất thường có thể được giải quyết thông qua bài toán phân loại các hành vi sử dụng hay truy cập tài nguyên của hệ thống để xác định các hành vi truy nhập tiềm tàng.

Đề tài “**Đánh giá hiệu năng của một số thuật toán phân lớp cho phát hiện xâm nhập”** nhằm giới thiệu khả năng và mức độ chính xác các thuật toán cơ bản dùng cho việc phát hiện các hành vi bất thường trong việc sử dụng và truy nhập mạng.

Luận văn gồm có các nội dung sau

**Chương 1: Tổng quan về phát hiện xâm nhập**

Chương này tìm hiểu chung về vấn đề phát hiện xâm nhập và hệ thống phát hiện xâm nhập dựa trên mạng.

**Chương 2: Một số thuật toán phân lớp cho phát hiện xâm nhập**

Giới thiệu các thuật toán phân loại cơ bản cho phát hiện xâm nhập mạng bao gồm: Cây quyết định, mạng nơ-ron nhân tạo, véc-tơ học máy SVM.

* Thuật toán SVM
* Thuật toán mạng nơ- ron nhân tạo
* Thuật toán cây quyết định

**Chương 3: Thực nghiệm và đánh giá**

Nội dung phân tích hiệu năng của các thuật toán cơ bản trên tập dữ liệu NSL KDD.

Chương 1 - TỔNG QUAN VỀ PHÁT HIỆN XÂM NHẬP

An toàn dữ liệu luôn là mối quan tâm hàng đầu với những người sử dụng máy tính và mạng máy tính, những năm gần đây vấn đề an ninh mạng đã trở nên cấp thiết và tác động lớn tới hiệu quả hoạt động của các mạng máy tính hiện đại. Chương này trình bày tổng quan về xâm nhập và phát hiện xâm phạm.

* 1. Xâm nhập mạng là gì.

“Tấn công hay còn gọi là xâm nhập mạng là những hoạt động có chủ đích, lợi dụng các tổn thương của hệ thống thông tin nhằm phá vỡ tính sẵn sàng, tính toàn vẹn và tính bảo mật của hệ thống”.

Kẻ xâm nhập trái phép có thể là kẻ đột nhập từ bên ngoài hệ thống máy tính, hệ thống mạng hoặc cũng có thể là một người dùng hợp pháp trong hệ thống máy tính, hệ thống mạng đó.

* 1. Một số kiểu tấn công phổ biến.

Có rất nhiều kiểu xâm nhập mạng khác nhau và thường được phân thành các loại chính: Tấn công từ chối dịch vụ, kiểu thăm dò, tấn công chiếm quyền “*root*”, tấn công điều khiển từ xa. Phần dưới đây sẽ trình bày khái quát về các kiểu tấn công này.

1. ***Tấn công từ chối dịch vụ***

Tấn công từ chối dịch vụ (*Denial of Service*) hay viết tắt DoS là kiểu tấn công làm cho một hệ thống nào đó bị quá tải không thể cung cấp dịch vụ, làm gián đoạn hoạt động của hệ thống hoặc hệ thống phải ngưng hoạt động [10].

Tùy theo phương thức thực hiện mà nó được biết dưới nhiều tên gọi khác nhau. Bắt đầu là lợi dụng sự yếu kém của giao thức TCP để thực hiện tấn công từ chối dịch vụ DoS. Ngoài ra còn có kiểu tấn công từ chối dịch vụ phân tán DDoS (*Distributed Denial of Service*) và mới nhất là tấn công từ chối dịch vụ theo phương pháp phản xạ DRDoS (*Distributed Reflection of Service*).

Tấn công từ chối dịch vụ cổ điển DoS sử dụng các hình thức: bom thư, đăng nhập liên tiếp, làm ngập SYN, tấn công Smurf, trong đó thủ phạm sinh ra nhiều giao tiếp ICMP (Internet control message protocol) tới địa chỉ Broadcast của các mạng với địa chỉ nguồn là mục tiêu cần tấn công, tấn công gây lụt UDP…

Tấn công dịch vụ phân tán DDoS xuất hiện vào năm 1999, so với tấn công DoS cổ điển, sức mạnh của DDoS cao hơn gấp nhiều lần. Hầu hết các cuộc tấn công DDoS nhằm vào việc chiếm dụng băng thông gây nghẽn mạch hệ thống dẫn đến hệ thống ngưng hoạt động. Để thực hiện thì kẻ tấn công tìm cách chiếm dụng và điều khiển nhiều máy tính/mạng máy tính trung gian, từ nhiều nơi để đồng loạt gửi ào ạt các gói tin với số lượng rất lớn nhằm chiếm dụng tài nguyên và làm nghẽn đường truyền của một mục tiêu xác định nào đó.

1. ***Tấn công thăm dò***

Đối với kiểu tấn công thăm dò (*Probe*), tin tặc quét mạng hoặc máy tính để tìm ra điểm yếu dễ tấn công mà thông qua đó tin tặc có thể thu thập thông tin trái phép về tài nguyên, các lỗ hổng hoặc dịch vụ của hệ thống. Các kiểu tấn công thăm dò như: Sniffing, Ping Sweep, Ports Scanning,…

Ví dụ: Sniffer là một hình thức nghe lén trên hệ thống mạng dựa trên những đặc điểm của cơ chế TCP/IP. Sniffer ban đầu là một kỹ thuật bảo mật, được phát triển nhằm giúp các nhà quản trị mạng khai thác mạng hiệu quả hơn và có thể kiểm tra các dữ liệu ra vào mạng cũng như các dữ liệu trong mạng. Sau này, kẻ tấn công dùng phương pháp này để lấy cắp mật khẩu hay các thông tin nhạy cảm khác. Biến thể của sniffer là các chương trình nghe lén bất hợp pháp như: Công cụ nghe lén yahoo, ăn cắp mật khẩu của thư điện tử,…

1. ***Tấn công chiếm quyền root***

Kiểu tấn công chiếm quyền root (*User to Root*) viết tắt là U2R, tin tặc với quyền của một người dùng hợp pháp cố gắng để đạt được quyền truy nhập cao nhất (đặc quyền của addmin) vào hệ thống một cách bất hợp pháp [10]. Cách thức phổ biến của kiểu tấn công này là gây tràn bộ đệm.

Kiểu tấn công này ít gặp hơn so với hai kiểu tấn công DoS và probe. Tuy nhiên, đây cũng là loại tấn công rất nguy hiểm do kẻ tấn công chiếm được quyền cao nhất và chúng có thể kiểm soát toàn bộ hệ thống.

1. ***Tấn công điều khiển từ xa***

Là kiểu tấn công điều khiển từ một máy tính từ xa có tên tiếng anh là “*remote to local*” hay R2L. Đầu tiên kẻ tấn công không có tài khoản trong hệ thống nhưng chúng lại cố gắng gửi các gói tin đến một máy tính trong một hệ thống thông qua mạng. Sau đó, chúng khai thác các lỗ hổng bảo mật để truy cập trái phép nhưng với tư cách của một người dùng cục bộ. Cách tấn công phổ biến của loại này là đoán mật khẩu thông qua phương pháp từ điển brute-force, FTP Write...

* 1. Một số giải pháp ngăn chặn xâm nhập truyền thống.

Một số các biện pháp ngăn chặn xâm nhập được sử dụng khá phổ biến như: tường lửa (*firewall*), mã hóa, xác thực, quyền truy cập…

* + 1. ***Tường lửa.***

Tường lửa có thể là phần mềm hay phần cứng được thiết kế để ngăn chặn những truy cập trái phép và cho phép các truy cập hợp pháp được lưu thông dựa trên một tập luật và các tiêu chuẩn khác. Ngoài ra tường lửa còn có thể mã hóa, giải mã hay ủy quyền cho các giao dịch giữa các miền bảo mật khác nhau. Một số loại tường lửa có chức năng chuyên dụng như: Tường lửa lọc gói, cổng ứng dụng, cổng mức mạch…

Một số nhược điểm của tường lửa:

* Không chống được các tấn công từ bên trong mạng. Tường lửa luôn giả sử rằng kẻ xấu chỉ ở bên ngoài mạng, còn mọi trạm bên trong mạng là đánh tin cậy.
* Không chống được sự lây nhiễm và phá hoại của các chương trình virus hay mã độc (malware).
* Không chống được các tấn công kiểu bỏ qua. Tường lửa cho rằng mọi kết nối giữa bên trong và ngoài mạng đều cần đi qua tường lửa, nhưng thực tế vẫn có một số kết nối không qua tường lửa do bản thân các tổ chức tạo nên.
  + 1. ***Mã hóa dữ liệu.***

Mã hóa dữ liệu là phương pháp bảo mật thông tin dữ liệu bằng cách biến đổi thông tin/ dữ liệu từ dạng thông thường sang dạng bị mã hóa không thể đọc được, nhưng có thể đưa về dạng ban đầu được nhờ một hình thức giải mã tương ứng. Một hệ thống mã hóa gồm thông tin gôc, thông tin đã mã hóa và khóa. Tác dụng của nó là ngăn chặn việc nghe trộm và chỉnh sửa dữ liệu trên đường truyền, bên thứ ba có thể lấy được các gói tin đã mã hóa, nhưng không thể đọc được nội dung thông điệp từ các gói tin này. Ngăn chặn việc giả mạo thông tin và đảm bảo tính không thể phủ nhận của an toàn mạng. Tương ứng với mỗi cách mã hóa là một cách giải mã nhất định, bao gồm thuật toán và bộ khóa mã là bí mật.

* + 1. ***Xác thực.***

Xác thực (*authentication*) là quá trình xác định xem người dùng đang truy cập vào hệ thống có phải là người dùng hợp lệ hay không, có một số cơ chế xác thực như: tài khoản và mật khẩu truy cập, dấu vân tay, giọng nói...

* + 1. ***Quyền truy cập***

Quyền truy cập là mức bảo vệ trong cùng. Sau khi xác thực thành công, người dùng có thể truy cập vào hệ thống. Tùy vào từng người dùng cụ thể mà quản trị sẽ phân quyền truy cập, sử dụng hệ thống khác nhau.

Mặc dù, bản thân mỗi hệ thống máy tính đều có những cơ chế bảo mật riêng  
nhằm chống lại và ngăn chặn những xâm nhập trái phép, nhưng những giải pháp bảo mật như firewall, mã hóa dữ liệu,… chưa đủ mạnh để có thể phát hiện, cảnh báo, ngăn chặn được những cuộc tấn công mới, ngày càng tinh vi hơn. Để nâng cao mức độ bảo vệ bên cạnh những cơ chế bảo mật trên, cần xây dựng thêm hệ thống giám sát chuyên biệt, có khả năng kiểm soát nội dung toàn diện hơn, phát hiện và đửa cảnh báo sớm về các nguy cơ tấn công. Hệ thống phát hiện xâm nhập intruction detected system (IDS) là một hệ thống gần đây được đông đảo những người liên quan đến bảo mật khá quan tâm.

* 1. Hệ thống phát hiện xâm nhập (instrucsion detection system).

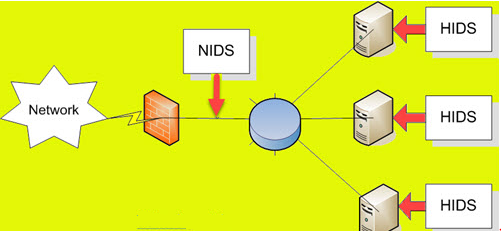
Phát hiện xâm nhập mạng là quá trình theo dõi các sự kiện xảy ra trong một hệ thống máy tính hoặc mạng máy tính và phân tích chúng để tìm ra các dấu hiệu sự cố có thể xảy ra, đó là các hành vi hoặc các mối đe dọa sắp xảy ra, vi phạm các chính sách bảo mật, các chính sách sử dụng được chấp nhận hoặc dựa trên tiêu chuẩn bảo mật.

Hệ thống phát hiện xâm nhập mạng (*Intrusion Detetion System*) là một hệ thống (có thể là thiết bị phần cứng hay phần mềm) nhằm giám sát lưu lượng mạng theo dõi, thu thập thông tin để phát hiện xâm nhập mạng và đưa ra cảnh báo.

Ưu điểm của IDS là:

* Cung cấp một cách nhìn toàn diện về toàn bộ lưu lượng mạng
* Giúp kiểm tra các sự cố đối với hệ thống
* Kết hợp với các hệ thống giám sát, tường lửa, diệt vi rút tạo thành hệ thống bảo mật tốt hơn.[5]

IDS sẽ được cài đặt thường trực trên sever hoặc máy tính cần theo dõi để phát hiện và cảnh báo khi có những bất thường liên quan đến hệ thống như: bẻ khóa, crack, truy cập vi phạm, đánh chặn lấy cắp IP, spyware, phần mềm gián điệp, các kết nối trái phép ra bên ngoài,nghe gói tin mạng, lạm dụng xác thực (thường xảy ra với các tấn công bên trong), thay đổi tài nguyên trái phép (bao gồm việc thay đổi, xóa, can thiệt vào thuộc tính thông, thay đổi cấu hình mạng, máy chủ, truyền tải dữ liệu trái phép), khai thác lỗ hổng hệ thống để truy cập thông tin, tấn công từ chối dịch vụ (gửi gói tin để làm lụt dữ liệu nhằm gây tắc nghẽn và hạn chế dịch vụ, tràn bộ đệm và làm tổn hại hệ thống từ xa).



Hình 1.1: Vị trí của hệ thống phát hiện xâm nhập (IDS) [5]

Kiến trúc và thành phần của IDS:

* Bộ phận thu thập phân tích gói tin: Bộ phận này có nhiệm vụ lấy tất cả các gói tin đi đến mạng.
* Bộ phận phát hiện gói tin: Trong bộ phận phát hiện gói tin có một thành phần quan trọng đó là bộ cảm biến. Vai trò của bộ cảm biến là dùng để lọc thông tin và loại bỏ dữ liệu không tương thích đạt được từ các sự kiện liên quan đến hệ thống bảo vệ, vì vậy có thể phát hiện được các hành động ghi ngờ
* Bộ phận xử lý (Phản ứng): Khi có dấu hiệu tấn công hoặc xâm nhập,thành phần phát hiện tấn công sẽ gửi tín hiệu đến thành phần phản ứng. Lúc đó thành phần phản ứng sẽ kích hoạt tường lửa có chức năng ngăn chặn cuộc tấn công hay cảnh báo tới người quản trị
  1. Phân loại hệ thống phát hiện xâm nhập..

Có các cách phân loại hệ thống IDS: Dựa trên vùng thu thập dữ liệu giám sát hoặc dựa trên phương pháp phân tích.

1. ***Dựa trên vùng thu thập dữ liệu giám sát .***

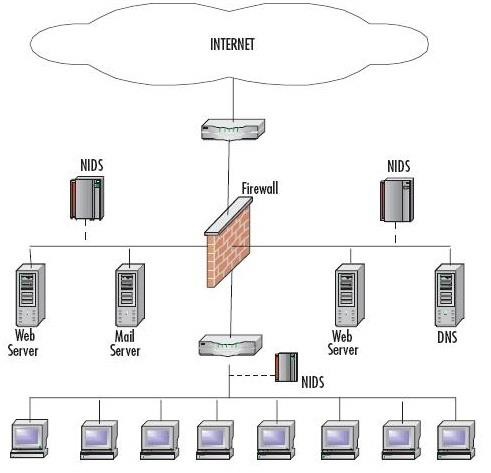
Phân theo cách này thì IDS được chia thành hai dạng hệ thống phát hiện ở mức mạng NIDS (Network -based IDS) và hệ thống phát hiện xâm nhập ở mức máy trạm chủ HIDS (Host – based IDS)

* *Hệ thống phát hiện xâm nhập ở mức mạng (Network – based IDS)*

NIDS là một hệ thống độc lập, xác định các truy cập trái phép bằng cách kiểm tra các luồng thông tin trên mạng và giám sát nhiều máy trạm. NIDS truy cập vào luồng thông tin trên mạng bằng cách kết nối vào các Hub, Switch được cấu hình Port mirroring hoặc Network tap để bắt các gói tin, phân tích nội dung các gói tin và từ đó sinh ra các cảnh báo hoặc phát hiện tấn công.

Port mirroring được sử dụng trong một Switch mạng để gửi một bản sao của tất cả các gói tin trên mạng khi nó đi qua cổng của Switch tới một thiết bị giám sát mạng trên cổng khác của Switch đó. Nó thường được dùng trong các thiết bị mạng cần giám sát luồng trên mạng, ví dụ như hệ thống IDS, Port mirroring trên Switch của Cisco System thường được gọi là Switched Port Analyzer (SPAN) hoặc 3Com là Roving Analysis Port (RAP).

Network tap là thiết bị phần cứng dùng để truy cập vào luồng dữ liệu đi ngang qua một mạng máy tính. Trong nhiều trường hợp, nó được xem như là một thành phần thứ ba để giám sát luồng dữ liệu trao đổi giữa hai điểm mạng, điểm A và điểm B. Một network tap có ít nhất ba cổng kết nối, một cổng A, một cổng B, và một cổng giám sát. Để đặt Network tap giữa điểm A và điểm B, cáp mạng giữa hai điểm A, B được thay thế bằng một cặp dây, một dây đấu cổng A và dây kia đấu cổng B. Network tap cho qua tất cả các dữ liệu giữa A và B vì thế giao tiếp giữa A và B vẫn diễn ra bình thường, tuy nhiên dữ liệu trao đổi đã bị Network tap sao chép và đưa vào thiết bị giám sát thông qua cổng giám sát.

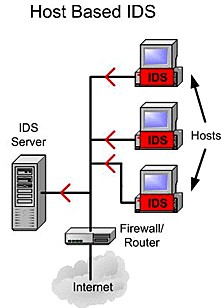


Hình 1.2: Hệ thống phát hiện xâm nhập NIDS [5]

Nhược điểm của hệ thống NIDS là giới hạn băng thông và có thể xảy ra hiện tượng tắc nghẽn cổ chai khi lưu lượng mạng hoạt động ở mức cao.

* *Hệ thống phát hiện xâm nhập ở mức máy trạm chủ (Host – based IDS)*

HIDS thường là một phần mềm chạy trên các trạm làm việc để giám sát tất cả các hoạt động trên máy trạm. Hệ thống này phân tích thông tin thu được trong nội bộ hệ thống, vì vậy nó cung cấp một cơ chế phân tích toàn diện các hoạt động và phát hiện một cách chính xác các thành phần tấn công.



Hình 1.3: Hệ thống phát hiện xâm nhập HIDS [5]

Nhược điểm của HIDS là việc thu thập dữ liệu xảy ra trên mỗi trạm máy chủ và ghi vào log do đó có thể làm giảm hiệu năng mạng.

1. ***Theo cách phân loại theo phương pháp phát hiện.***

IDS được chia làm các dạng

Phát hiện dấu hiệu dựa vào đặc trưng (của cuộc xâm nhập) hay dựa vào dấu hiệu (*Signature-based IDS*)

Phát hiện dựa vào bất thường (*Anomaly-based IDS*).

Đặc tả, còn gọi là phân tích giao thức có trạng thái*(stateful protocol analysis)*

* *Hệ thống phát hiện xâm nhập dựa vào đặc trưng (Signature-based IDS)*

Hệ thống IDS loại này dựa vào các dấu hiệu của các cuộc xâm nhập. Những dấu hiệu đó có thể là thông tin về các kết nối nguy hiểm đã biết trước. Hệ thống sẽ lưu và mô hình hóa các dấu hiệu của các cuộc xâm nhập đã biết và bằng việc so sánh thông tin của các gói tin đến với các dấu hiệu này để phát hiện ra các hoạt động đáng ngờ và đưa ra cảnh báo cho hệ thống.

Ưu điểm của hệ thống này là rất hiệu quả trong việc phát hiện tấn công đã biết với tỷ lệ cảnh báo sai thấp. Tuy nhiên, nhược điểm chính của hệ thống là chỉ phát hiện được những cuộc tấn công đã biết, không phát hiện được cuộc tấn công mới, và biến thể, do đó phải thường xuyên cập nhật các đặc trưng (dấu hiệu) về các cuộc tấn công mới và bộ nhớ, thời gian phát hiện tăng khi cơ sở dữ liệu lớn.

* *IDS phát hiện theo đặc tả*

IDS loại này nhận biết và theo dõi các trạng thái các giao thức (tương ứng giữa cặp yêu cầu/ đáp ứng). Việc đặc tả phụ thuộc vào nhà cung cấp giao thức.

* *Hệ thống phát hiện xâm nhập dựa vào bất thường (Anomaly-based IDS)*

Ý tưởng của cách tiếp cận này xuất phát từ giả thiết “*Dấu hiệu của các cuộc tấn* *công khác biệt với dấu hiệu của những trạng thái mạng được coi là bình thường*”. Khi đó việc phát hiện sẽ diễn được tiến hành qua hai giai đoạn: Giai đoạn huấn luyện (pha huấn luyện) và gia đoạn phát hiện (pha phát hiện). Tại pha huấn luyện sẽ xây dựng một hồ sơ về các hoạt động bình thường (thông số chuẩn). Sau đó tại pha phát hiện sẽ tiến hành so khớp các quan sát (gói tin) với hồ sơ từ đó xác định dấu hiệu bất thường.

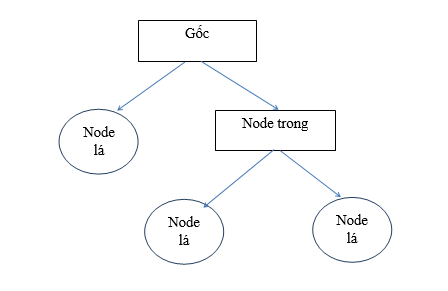
Ưu điểm của hệ thống này là hiệu quả trong việc phát hiện các mối nguy hiểm không được biết trước. Phát hiện xâm nhập dựa trên bất thường có nhiều triển vọng do nó có khả năng phát hiện các tấn công, xâm nhập mới. Những năm gần đây, hướng tiếp cận này đang thu hút rất nhiều sự quan tâm của các nhà nghiên cứu. Do đó, luận văn cũng tập trung vào hướng tiếp cận này.. Việc phát hiện bất thường có thể được giải quyết thông qua bài toán phân loại các hành vi sử dụng hay truy cập tài nguyên của hệ thống để xác định các hành vi truy nhập tiềm tàng. Chương sau sẽ giới thiệu một số thuật toán phân lớp ứng dụng cho phát hiện xâm nhập.

Chương 2 - MỘT SỐ THUẬT TOÁN PHÂN LỚP CHO PHÁT HIỆN XÂM NHẬP

1. Học cây quyết định.

Học cây quyết định là một trong phương pháp học máy tiêu biểu có nhiều ứng dụng trong phân loại và dự đoán. Học cây quyết định có nhiều ưu điểm như đơn giản, dễ lập trình và cho phép biểu diễn hàm phân loại dưới dạng dễ hiểu, dễ giải thích cho con người.

Cây quyết định: là biểu đồ quyết định phát triển có cấu trúc dạng cây



Hình 2.1: Mô tả chung về cây quyết định

Trong đó

* Gốc: Nút trên cùng của cây.
* Nút trong: Biểu diễn một kiểm tra trên một thuộc tính đơn (hình chữ nhật)
* Nhánh: Biểu diễn các kết quả của kiểm tra trên node trong (mũi tên)
* Node lá: Biểu diễn lớp (hình tròn) hay là nhãn phân loại

Cây quyết định dùng để xác định nhãn phân loại cho một dữ liệu nào đó, giá trị các thuộc tính mẫu dược đưa vào kiểm tra trên cây quyết định. Mỗi mẫu tương ứng với một đường đi từ gốc tới node lá. Tại mỗi nút, thuộc tính tương ứng với nút được kiểm tra, tùy theo giá trị của thuộc tính đó mà ví dụ được chuyển xuống nhánh tương ứng bên dưới. Quá trình này lặp lại cho đến khi ví dụ tới được nút lá và được nhận nhãn phân loại là nhãn của nút lá tương ứng.

1. ***Xây dựng cây quyết định.***

Trước khi sử dụng cây quyết định,ta cần xây dựng cây quyết đinh(hay học cây quyết định) từ dữ liệu huấn luyện. Có nhiều thuật toán xây dựng cây quyết định, đa số các thuật toán này dựa trên nguyên tắc chung là xây dựng cây theo kiểu tìm kiếm tham lam từ cây đơn giản tới cây phức tạp hơn (từ trên xuống).

Quá trình xây dựng một cây quyết định cụ thể bắt đầu bằng một nút rỗng bao gồm toàn bộ các đối tượng huấn luyện và thực hiện như sau:

* Nếu tại nút hiện thời, tất cả các đối tượng huấn luyện đều thuộc vào một lớp nào đó thì nút này là nút lá có tên là nhãn lớp chung của các đối tượng.
* Ngược lại với trường hợp trên, sử dụng một số độ đo, chọn thuộc tính điều kiện phân chia tốt nhất tập mẫu huấn luyện có tại nút.
* Tạo một lượng nút con của nút hiện thời bằng số các giá trị khác nhau của thuộc tính con được lựa chọn. Gán cho mỗi nhánh từ nút cha đến nút con mộ giá trị của thuộc tính rồi phân chia các đối tượng huấn luyện vào các nút con tương ứng
* Tại nút con K được gọi là thuần nhất, trở thành lá nếu tất cả các đối tượng mẫu tại đó đều thuộc cùng vào một lớp.
* Lập lại các bước 1-3 đối với mỗi nút chưa phải là thuần nhất.

Cây quyết định được tạo thành bằng cách lần lượt chia (theo phương pháp đệ quy) một tập dữ liệu thành các tập dữ liệu con, mỗi tập con được tạo thành từ các phần tử của cùng một lớp. Các nút (không phải là nút lá) là các điểm phân nhánh của cây. Việc phân nhánh tại các nút có thể dựa trên việc kiểm tra một hay nhiều thuộc tính để xác định việc phân chia dữ liệu.

Chúng ta mong muốn chọn thuộc tính sao cho việc phân chia lớp tập mẫu là tốt nhất. Thuộc tính lựa chọn là thuộc tính cho phép tạo ra nững tập con có độ đồng nhất cao nhất (trường hợp lý tưởng thuộc tính lựa chọn là thuộc tính cho phép chia tập dữ liệu thành các tập con cùng nhãn, do vậy chỉ cần một phép kiểm tra thuộc tính khi phân loại). Yêu cầu đặt ra là cần có một tiêu chí đo độ đồng nhất của tập dữ liệu và mức tăng độ đồng nhất đó khi sử một thuộc tính nào đó. Một cách đo độ đồng nhất đó là sử dụng entropy.

1. ***Xây dựng cây quyết định dựa vào Entropy.***

Tiêu chí để đánh giá tìm điểm chia là rất quan trọng, chúng được xem là một tiêu chuẩn “heuristic” để phân chia dữ liệu. Ý tưởng chính trong việc đưa ra các tiêu chí trên là làm sao cho các tập con được phân chia càng trở nên “trong suốt” (tất cả các bộ thuộc cùng một nhãn) tốt nhất có thể. Thuât toán dùng độ đo lường thông tin thu thêm (informationGain- IG) để xác định điểm chia. Độ đo này dựa trên cơ sở lý thuyết thông tin của nhà toán học Claude shannon, độ đo này được xác như sau:

Xét bảng quyết định DT = (U, C ∪{d}), số giá trị (nhãn lớp) có thể của d là k. Khi đó Entropy của tập các đối tượng trong DT được định nghĩa bởi:

 (2.1)

Trong đó  là tỉ lệ các đối tượng trong DT mang nhãn lớp i.

Ý nghĩa của đại lượng Entropy trong lĩnh vực lý thuyết công nghệ thông tin:  
Entropy của tập U chỉ ra số lượng bít cần thiết để mã hóa lớp của một phần tử được lấy ra ngẫu nhiên từ tập U. Lượng thông tin thu thêm (Information Gain- IG) là lượng Entropy còn lại khi tập các đối tượng trong DT được phân hoạch theo một  
thuộc tính điều kiện c nào đó. IG xác định theo công thức:

 (2.2)

Trong đó là tập các giá trị của thuộc tính c, là tập các đối tượng trong DT có giá trị thuộc tính c bằng v. Giá trị IG(U, c) được sử dụng làm độ đo lựa chọn thuộc tính phân chia dữ liệu tại mỗi nút trong thuật toán xây dựng cây quyết định. Thuộc tính được chọn là thuộc tính cho lượng thông tin thu thêm lớn nhất. Ý nghĩa của đại lượng IG trong lĩnh vực lý thuyết công nghệ thông tin: IG của tập S chỉ ra số lượng bít giảm đối với việc mã hóa lớp của một phần tửc được lấy ra ngẫu nhiên từ tập U.

Một số thuật toán xây dựng cây quyết định như: ID3, J48, CART (Classification and Regression Tree).

Thuật toán ID3, và Cart cho hiệu quả phân lớp cao đối với các trường dữ liệu số (quantitative value) trong khi đó các thuật toán C4.5 (C5.0) có hiệu quả đối với các dữ liệu ordinal, binary, nomial.

Điểm mạnh của cây quyết đinh: sinh ra các quy tắc hiểu được chuyển đổi được sang SQL, dễ dàng tính toán trong khi phân lớp, xử lý với thuộc tính liên tục và rời rạc, thể hiện rõ ràng những thuộc tính tốt nhất.

Điểm yếu của cây quyết định: Dễ xảy ra lỗi khi có nhiều phân lớp (do chỉ thao tác với các lớp có giá trị nhị phân, tốn thời gian và bộ nhớ do phải đi qua nhiều node để đến với node lá cuối cùng).

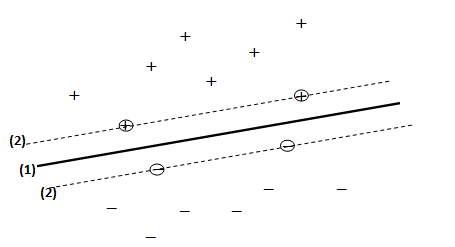
1. Máy vectơ hỗ trợ (support vector machine-SVM).

Máy vectơ hỗ trợ (SVM) là một giải thuật máy học dựa trên lý thuyết học thống kê do Vapnik (1998) đề xuất. Bài toán cơ bản của SVM là bài toán phân lớp loại 2 lớp: Cho trước *n* điểm trong không gian *d* chiều (mỗi điểm thuộc vào một lớp ký hiệu là +1 hoặc -1, mục đích của giải thuật SVM là tìm một siêu phẳng (hyperplane) phân hoạch tối ưu cho phép chia các điểm này thành hai phần sao cho các điểm cùng một lớp nằm về một phía với siêu phẳng này

1. *Ý tưởng thuật toán.*

Ý tưởng cơ bản của SVM là ánh xạ (tuyến tính hoặc phi tuyến) dữ liệu vào không gian các vector đặc trưng (space of feature vectors) mà ở đó một siêu phẳng tối ưu được tìm ra để tách dữ liệu thuộc hai lớp khác nhau.

Mục đích của phương pháp SVM là tìm được khoảng cách biên lớn nhất:



Hình 2.2: Mô tả phương pháp SVM

1. *Cơ sở lý thuyết.*

Giả sử dữ liệu trong không gian x được ánh xạ đến không gian y với y= (x). Và giả sử có một siêu phẳng phân tách tạo bởi phương trình

wT+b=0 (2.3)

Trong đó:

* x: vector đầu vào (mẫu)
* w: vector trọng số(được điều chỉnh)
* b: Giá trị lệch (khoảng cách từ gốc tọa độ đến siêu phẳng)

Khoảng cách giữa điểm dữ liệu gần nhất đến siêu phẳng được gọi là lề (Margin of separation) và được kí hiệu là.Mục tiêu là tìm được  lớn nhất. Siêu phẳng phân tách sao cho margin lớn nhất gọi là siêu phẳng tối ưu.

Giả sử di là đáp án mong muốn,và giá trị +1,và-1 biểu thị hai lớp phân tách được

Ta có

, 

,  (2.4)

Xi là mẫu ví dụ thứ i, Do đó siêu phẳng phân tách tối ưu sẽ là

 (2.5)

Tại siêu phẳng tối ưu là

 (2.6)

 (2.7)

Trong đó, xp là khoảng cách của x tới siêu phẳng, r là khoảng cách mong muốn, r dương nếu x được đặt ở lớp dương (+1) và âm nếu x được đăt trên mặt âm (-1) của siêu phẳng

Giá trị của r được tính bằng công thức

 (2.8)

Theo như trên, (w0, b0) là được cho bởi

,  (2.9)

, 

Các điểm dữ liệu (xi,di) thỏa mãn một trong các phương trình trên (với dấu bằng) được gọi là vector hỗ trợ. Tập các điểm này gọi là *Support vector Machine.* Trong hệ thống này, sẽ được tính bởi

 (2.10)

Trong hệ thống này, hàm chi phí được định nghĩa là: 

Huấn luyện sao cho hàm này là nhỏ nhất, sử dụng toán tử Lagrage , một hàm mục tiêu có thể được xây dựng:

;  (2.11)

Với các ràng buộc:

 (2.12)

, i=1,2,…,N (2.13)

là một giá trị tối ưu, được thay thế vào  và kết quả là:

 (2.14)

Các giá trị tối ưu có thể được tính theo công thức:

  (2.15)

Với vector đặc trưng , với véc tơ đầu vào xi, trong mẫu ví dụ thứ i,Ta có biểu thức sau:

 (2.16)

Vì thế hàm nhân có thể được định nghĩa:

 (2.17)

Theo định lý Mercer chỉ ra rằng, hàm K(x,x’) là hàm kenel là đối xứng liên tục giá trị thuộc khoảng a ≤x≤b

Tương tự cũng đúng với mọi x0, do đó hàm K(x,x’) có thể được biểu diễn:

 (2.18)

Với  là một hệ số cho i, hàm  được gọi hàm đặc trưng, và dược gọi là giá trị đặc trưng.

Theo tài liệu của Haykin chỉ ra rằng:

* Với ≠1, hình ảnh thứ i  tạo ra trong không gian đặc trưng bởi véc tơ đầu vào x, với số chiều lớn hơn
* Theo lý thuyêt, số chiều của không gian đặc trưng (tức là số lượng giá trị đặc trưng/ hàm đặc trưng)có thể là vô hạn

Do đó, hàm mục tiêu với ràng buộc tối ưu có thể thu được:

 (2.19)

Với điều kiện:



0 ≤αi≤C, với i=1,2…,N (2.20)

Trong đó C là tham số xác định chi phí lỗi, trọng số được tính bằng:

 (2.21)

1. *Một số ứng dụng tiêu biểu của SVM.*

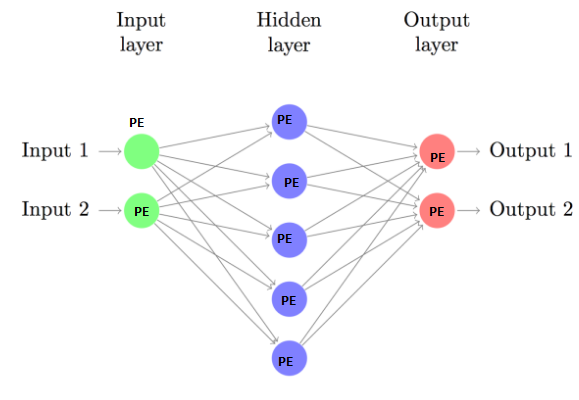
Máy véc-tơ hỗ trợ được sử dụng trong nhiều tình huống phân loại khác nhau như phân loại các văn bản, tài liệu Web, nhận dạng hình ảnh hay phân loại các chức năng các protein trong ứng dụng sinh học. So với các phương pháp dựa trên luật hay cây quyết định, các kết quả dựa trên SVM khó hiểu hơn và khó giải thích. Tuy vậy SVM cải thiện được hiệu năng phân loại khá tốt so với thuật toán cây quyết định truyền thống.

1. Mạng neural nhân tạo (Artificial Neural Networks)-ANN.

Là mô hình xử lý thông tin mô phỏng hoạt động của hệ thống thần kinh sinh vật, bao gồm số lượng lớn các neural được gắn kết để xử lý thông tin. ANN giống như bộ não con người, được học bởi kinh nghiệm (qua huấn luyện), có khả năng lưu giữ những kinh nghiệm hiểu biết (tri thức) và sử dụng những tri thức đó trong việc dự đoán các dữ liệu chưa biết.

1. ***Kiến trúc của một mạng neural nhân tạo.***

Kiễn trúc chung của một mạng neural nhân tạo thường: thường gồm 3 phần đó là:Input layer (lớp đầu vào), Hide layer (lớp ẩn) và output layer(lớp đầu ra).



Hình 2.3: Kiến trúc chung của một mạng neural nhân tạo

Trong đó các processing elements (PE) của ANN gọi là neural,mỗi neural nhận các dữ liệu vào và xử lý chung cho ra một kết quả output duy nhất. Output này có thể là input cho neural khác. Có 3 kiểu nút đó là iput node, hide node và output node tương ứng ở 3 lớp của mạng neural.

Lớp nhập đóng vai trò nhập dữ liệu và phân tích sơ bộ có thể coi như một mạng neural một lớp riêng, đầu ra của lớp chính là đầu vào của lớp ẩn.

Lớp Hide layer gồm các neural nhận dữ liệu từ input từ các neural ở các lớp (layer) trước và chuyển đổi input này cho các lớp xử lý tiếp theo. Có thể có nhiều lớp ẩn trong 1 mạng.

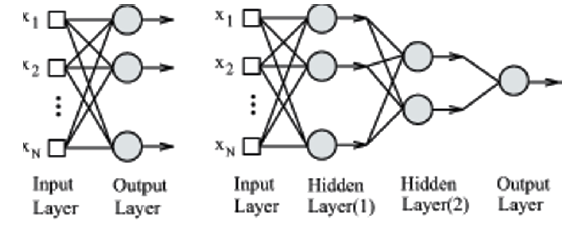
Số node ở hidden layer càng nhiều, nghĩa là số chiều của feature càng lớn thì chi phí tính toán sẽ tăng lên, để chọn được số nút ở hide layer phù hợp phụ thuộc vào bài toán.

Lớp xuất chính là lớp đầu ra của mạng neuron cung cấp kết quả bài toán được xét tới. Các kết quả này được thông qua quá trình xử lý lớp ẩn.

Có một số loại mạng neural như

1. ***Mạng truyền thẳng (feedforward networks)***

Mạng neural truyền thẳng bảo gồm tập hợp các nút kết nối với nhau trong một lớp (single layer) hay nhiều lớp (multi layer). Trong kiến trúc này không có kết nối phản hồi, do đó hình ảnh mạng neural truyền thẳng được mô tả như đồ thị vô hướng.



1. Mạng neural một lớp (b) Mạng neural da lớp [4]

Hình 2.4: Mô hình mạng neural truyền thẳng [4]

Hình (a) là mô hình mạng neural truyền thẳng một lớp, đây là mô hình đơn giản nhất, chỉ một lớp đầu vào và một lớp đầu ra.

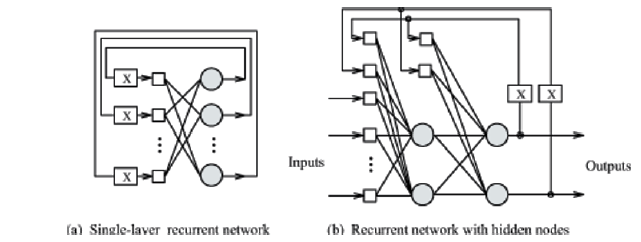
Hình (b) là mạng neural đa lớp (multilayer perceptron, MLP).Mạng neural nhiều lớp có thể giải quyết các bài toán phi tuyến nhờ vào các lớp ẩn. Các lớp ẩn này xen giữa các input bên ngoài và output của mạng. Càng nhiều lớp ẩn thì khả năng mở rộng thông tin càng cao và xử lý tốt mạng có nhiều input và output. Ngoài ra còn có mạng hồi quy và mạng Neural dạng lưới.

1. ***Mạng hồi quy (Recurrent networks)***

Các mạng hồi quy là mạng truyền thẳng với các luồng phản hồi. Có hai loại mạng liên tục và mạng rời rạc.

Hình a là mạng hồi quy không có lớp ẩn.

Hình b là mạng hồi quy có lớp ẩn.

Hình 2.5: Mô hình mạng hồi quy[4]

1. ***Kết nối các neural***

Các nút trong mạng được liên kết với nhau.

+ Connection weights (trọng số liên kết): Đây là thành phần rất quan trọng của một ANN, nó thể hiện mức độ quan trọng của dữ liệu (độ mạnh) đối với quá trình xử lý thông tin(quá trình chuyển đổi dữ liệu từ layer này sang layer khác.Quá trình học(learning processing) của ANN thực ra là quá trình điều chỉnh các trọng số (weight) của input data để có được kết quả mong muốn.

+ Summation function (hàm tổng: Tính tổng trọng số đối với tất cả các input được đưa vào mỗi neural (phần tử xử lý PE). Hàm tổng của một neural đối với input được tính theo công thức sau:



+ Activation function –hàm kích hoạt(hay còn gọi là hàm chuyển đổi) là hàm nhận vector đầu vào, sau đó biến đổi để trả về vector đầu ra. Có nhiều hàm activation function như:

Hàm Binary Threshold:

Y**T=**

Hàm Piecewise Linear:

YT =

Hàm Sigmoid:

Y**T**= 1/(1 + e**-αY**)

trong đó α là hệ số hàm Sigmoid α ϵ [0,1] []

Việc lựa chọn hàm kích hoạt có tác động lớn đến kết quả của ANN. Hàm phi tuyến được sử dụng phổ biến trong ANN là sigmoid (logical activation) function.

1. ***Quá trình học của ANN***

ANN được huấn luyện hay được học theo 2 kỹ thuật cơ bản là học có giám sát và học không giám sát.

- Học có giám sát: quá trình huấn luyện được lặp lại cho đến khi kết quả (output) của ANN đạt được giá trị mong muốn đã biết. Điển hình cho kỹ thuật này là mạng neural lan truyền ngược (back-propagation).

- Học không giám sát: không sử dụng tri thức bên ngoài trong quá trình học, nên còn gọi là tự tổ chức (Self - Organizing). Mạng neural điển hình được huấn luyện theo kiểu không giám sát là SOM.

Quá trình học có giám sát của ANN được mô tả như ở Hình 3, gồm các bước:

1. Tính giá trị output *Y*.
2. So sánh *Y* với giá trị mong muốn *Z*.
3. Nếu chưa đạt giá trị mong muốn (*delta* = *Z* - *Y* lớn) thì chỉnh trọng số (weights) và tính lại output cho đến khi *delta* = 0 hoặc nhỏ đến mức chấp nhận được.



*delta* = *Zj* - *Yj*

*delta*

*Zj*

*x1*

*w*1*j*

*x2*

*w*2*j*

PE: Neural

*j* = *wijxi*

∫

Hàm chuyển giao

*Yj*

*xi wij*

Hình 2.6: Hoạt động của neural

xi: các tín hiệu input

wkp: trọng số của từng input

F(.): hàm kích hoạt

yk: kết xuất của neural

delta: thông số ảnh hưởng đến ngưỡng ra của output

1. ***Ứng dụng của mạng neural nhân tạo***

Mạng neural được ứng dụng trong các bài toán dự đoán và phân loại như: dự đoán chứng khoán, dự báo các vấn đề về chi phí, chất lượng kết hoạch trong các dự án, phân loại ảnh (thị giác máy tính), nhận dạng văn bản (xử lý ngôn ngữ tự nhiên)....

Chương 3 – THỰC NGHIỆM VÀ ĐÁNH GIÁ

Chương này phân tích và đánh giá hiệu năng của các thuật toán cơ bản trên cơ sở dữ liệu thử nghiệm thu thập được, trình bày việc lập mô hình thử nghiệm các thuật toán trên bộ dữ liệu mô phỏng và thu thập dữ liệu dùng cho việc phân tích đánh giá.

1. Một số độ đo đánh giá
2. ***Các độ đo đánh giá***

Xét bài toán phân loại hai lớp, trong đó mỗi ví dụ có thể nhận nhãn *dương (nhiễm)* hoặc *âm (không nhiễm- bình thường)*.

Với mỗi trường hợp ví dụ mà mô hình dự đoán nhãn, có bốn khả năng xẩy ra như liệt kê trên bảng sau,

|  |  |  |  |
| --- | --- | --- | --- |
|  | Nhãn dự đoán | | |
| Nhãn thật | Dương | Âm | Tổng số |
| Dương | tp: dương đúng | fn: âm sai | P |
| Âm | fp: dương sai | tn: âm đúng | N |
| Tổng số | p' | n' | N |

Trong đó

* Nhãn thật: Là nhãn của ví dụ
* Nhãn dự đoán: Là nhãn do chương trình dự đoán ra.
* tp: là nhãn dương trong mẫu được chương trình phân loại dương
* fp: là nhãn dương trong mẫu, chương trình phân loại gán nhãn âm.
* tn: là nhãn âm, chương trình phân loại âm
* fn: nhãn âm được chương trình phân loại gán dương.

Thì ta có định nghĩa một số độ đo sau:

* **Tỷ lệ lỗi**: error = (fp + fn)/N.
* **Độ chinhd xác accuracy**: accuracy = (tp + tn)/N = 1 – error.
* **Tỉ lệ dương đúng**: tp-rate = tp/p.
* **Tỉ lệ dương sai**: fp-rate = fp/p.
* **Độ chính xác precision**: precision = tp/p.
* **Độ thu hồi Recall**= tp/p=tp-rate.
* **Độ đo F: F-measure** =2\*p\*r(precision+recall).

Các độ đô nói trên đều có gía trị nằm trong khoảng [0,1].

Các chỉ số này sẽ được dùng để đánh giá kết quả thử nghiệm.

1. ***Cách đánh giá***

Có nhiều kỹ thuật đánh giá độ chính xác dự báo như: đánh giá chéo K-fold, Holdout, Re-substitution và Leaveone-out [8].

Đánh giá mô hình bằng kiểm tra chéo (cross-validation).

**Kiểm tra chéo với tập kiểm tra tách riêng** (hold-out cross validation), còn gọi là kiểm tra chéo đơn giản.

Cách thực hiện:

* Chia ngẫu nhiên tập dữ liệu S huấn luyện ban đầu thành 2 tập: Shl và Skt
* (Skt= S- Shl). Thông thường Shl= 70% của S.
* Huấn luyện mô hình cần đánh giá hi trên tập Shl,.
* Đánh giá độ chính xác của mô hình hi trên tập kiểm tra Skt.
* Chọn mô hình có độ chính xác cao nhất trên tập kiểm tra để sử dụng (Nếu mục đích là lựa chọn mô hình).

Do mô hình được đánh giá trên tập dữ liệu kiểm tra *Skt*, là dữ liệu chưa được dùng khi huấn luyện nên độ chính xác của mô hình trong trường hợp dữ liệu mới nó chung đươc ước lượng chính xác hơn. Thông thường, tập kiểm tra có kích thước bằng 1/4 đến 1/3 tập dữ liệu ban đầu (30%).

Trong trường hợp lựa chọn mô hình, mô hình tốt nhất được chọn sau đó được luyện lại trên toàn bộ tập dữ liệu ban đầu. Việc huấn luyện lại mô hình trên tập dữ liệu lớn hơn như vậy cho phép tăng độ chính xác.

Một nhược điểm của phương pháp sử dụng bộ dữ liệu kiểm tra riêng là phần dùng để huấn luyện huấn luyện (tập *Shl*) chỉ còn khoảng 70%.

Tập ban đầu và do vậy bỏ phí quá nhiều dữ liệu để kiểm tra. Dưới đây là phương pháp kiểm tra chéo khác cho phép sử dụng ít dữ liệu kiểm tra hơn.

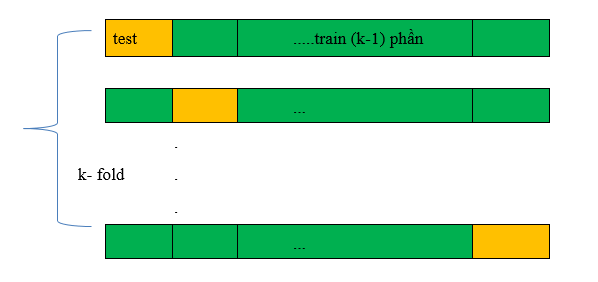
**Kiểm tra chéo k-fold** (k-fold cross validation).

Cách thực hiện:

1. Chia ngẫu nhiên tập dữ liệu ban đầu *S* thành *k* tập dữ liệu có kích thước gần bằng nhaunhau S1, S2,…, Sk.
2. Lặp lại thủ tục sau k lần với i= 1, 2..,k
3. Dùng tập Si làm tập kiểm tra.Gộp K-1 tập còn lại thành tập huấn luyện. Huấn luyện mô hình cần đánh giá trên tập huấn luyện.
4. Đánh giá độ chính xác của mô hình trên tập kiểm tra, Độ chính xác của mô hình tính bằng trung bình cộng độ chính xác trên k lần kiểm tra ở bước trên.
5. Chọn mô hình có độ chính xác trung bình lớn nhất.

**Ưu điểm**: Chính của kiểm tra chéo k- fold là nhiều dữ liệu hơn được sử dụng cho huấn luyện. Mỗi ví dụ được sử dụng để kiểm tra đúng 1 lần, trong khi được sử dụng trong tập huấn luyện k-1 lần.

**Nhược điểm**: Phải huấn luyện và đánh giá mô hình k lần, do vậy mất nhiều thời gian. Thông thường, phương pháp này được sử dụng với k=10. Giá trị này vừa cho kết quả khách quan vừa không đòi hỏi huấn luyện mô hình quá nhiều.



Hình 3.1: Đánh giá cross- validation k fold

1. Giới thiệu bộ dữ liệu NSL- KDD
2. ***Giới thiệu.***

NSL-KDD là tập dữ liệu được tinh chỉnh của tập dữ liệu gốc KDD99 bằng cách:

* + Loại bỏ các bản ghi trùng lặp
  + Số lượng các bản ghi đủ lớn với tập huấn luyện và kiểm tra

Ưu điểm của NSL- KDD: Khắc phục được hạn chế việc dư thừa trong bộ dữ liệu gốc (không có bản ghi dự phòng trong tập huấn luyện), số lượng các bản ghi dư thừa, dẫn đến thuật toán sẽ thiên vị cho những bản ghi xuất hiện nhiều lần (thường xuyên hơn, do đó hạn chế đối với các bản ghi xuât hiện ít lần và gây hại cho các mạng như tấn công U2R)

Số lượng các bản ghi trong bộ huấn luyện và kiểm tra là hợp lý vừa đủ lớn, cho nên có thể chạy thử nghiệm trên bộ hoàn chỉnh mà không cần ngẫu nhiên chọn một phần nhỏ mà chi phí không lớn. Do đó kết quả đánh giá của các công trình nghiên cứu khác nhau sẽ nhất quán. Phù hợp để đánh giá các thuật toán.

1. ***Mô tả tập dữ liệu NSL KDD.***

+ Các tập dữ liệu dùng trong tập dữ liệu NSL KDD được mô tả trong bảng 3.1 dưới đây:

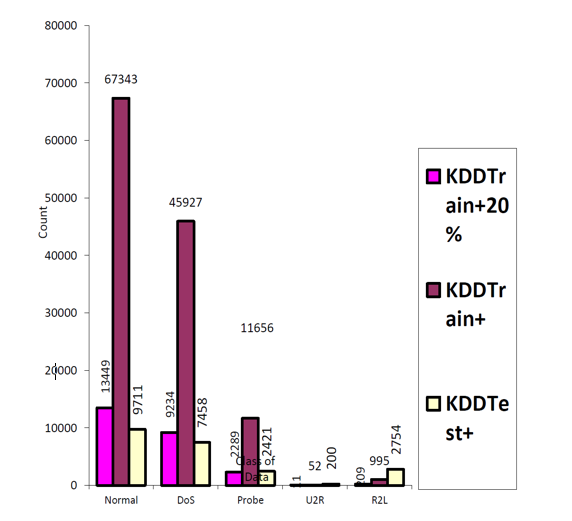
Bảng 3. 1: Danh sách và mô tả tập file NSL- KDD

|  |  |  |
| --- | --- | --- |
| **TT** | **Tập dữ liệu** | **Mô tả** |
| 1 | KDD Train+.ARFF | Tập huấn luyên đầy đủ định dạng ARFF |
| 2 | KDD Trian+.TXT | Tập hợp đầy đủ NSL-KDD bao gồm nhãn loại tấn công và mức độ khó khăn ở định dạng CSV |
| 3 | KDDTrian+\_20 percent.ARFF | Tập con 20% của KDDTrain+.arff |
| 4 | KDD Trian+\_20 Percent.TXT | Tập hợp con 20% của tệp KDDTrain +. txt |
| 5 | KDDTest+.ARFF | Tập đầy đủ kiểm tra nhãn nhị phân ở định dạng ARFF |
| 6 | KDDTest+.TXT | Tập NSL-KDD kiểm tra bao gồm các nhãn tấn công ở mức độ khó ở định dạng CSV |

Thông tin và số lượng các bản ghi phân bố trong các tập dùng để thử nghiệm được thể hiện trong bảng 3.2 và hình 3.2 bên dưới.

Bảng 3. 2: Thông tin chi tiết các bản ghi trong các tập dữ liệu

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Tập dữ liệu** | **Số bản ghi tương ứng với từng loại tấn công** | | | | | **Tổng số bản ghi** |
| **Normal** | **Dos** | **Probe** | **R2L** | **U2R** |
| KDDTrain+ | 67343 | 45927 | 11656 | 52 | 995 | 125973 |
| KDDTrain+\_20Percent | 13449 | 9234 | 2289 | 1 | 209 | 25182 |
| KDDTest+ | 9711 | 7458 | 2421 | 200 | 2754 | 22544 |



Hình 3.2: Phân bố các bản ghi [1]

**+ Các thuộc tính của bản ghi:** Mỗi bản ghi bao gồm 41 thuộc tính thể hiện các đặc trưng khác nhau của luồng thông tin và được gán nhãn hoặc là tấn công hoặc là bình thường.

Bảng 3. 3: Mô tả các thuộc tính mỗi bản ghi

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **STT** | | **Thuộc tính** | | **Mô tả** | | **Ví dụ** | |
| 1 | | Duration | | Thời gian kết nối (s) | | 0 | |
| 2 | | Protocol\_type | | Kiểu giao thức | | Tep | |
| 3 | | Service | | Dịch vụ mạng sử dụng | | Fpt\_data | |
| 4 | | Flag | | Tình trạng kết nối (cờ)  (bình thường hay lỗi) | | SF | |
| 5 | | Src\_bytes | | Lượng dữ liệu chuyển từ nguồn tới đích | | 491 | |
| 6 | | Dst\_bytes | | Số byte dữ liệu từ đích tới nguồn | | 0 | |
| 7 | | Land | | 1 Nếu kết nối là cùng số cổng với máy chủ. Ngược lại=0 | | 0 | |
| 8 | | Wrong\_fragm | | Số lượng các phân mảng “lỗi” trong kết nối | | 0 | |
| 9 | | Urgent | | Số lượng các gói dữ liệu urgent (ưu tiên) trong kết nối.Gói ưu tiên là gói cờ urgent được gán hoạt động | | 0 | |
| **STT** | **Thuộc tính** | | **Mô tả** | | **Ví dụ** | |
| 10 | Hot | | Chỉ số các “hot” trong nội dung như tài khoản đăng nhập hệ thống, thư mục, khởi tạo và thực thi chương trình | | 0 | |
| 11 | Num\_failed\_logins | | Số lần đăng nhập lỗi | | 0 | |
| 12 | Logged\_in | | Trạng thái đăng nhập  (= 1 nếu đăng nhập thành công  = 0 nếu không) | | 0 | |
| 13 | num\_compromised | | Số lần trong tình trạng tấn công | | 0 | |
| 14 | Root\_shell | | 1 Nếu root thực thi sell, tron g trường hợp ngược lại =0 | | 0 | |
| 15 | Su\_attempted | | =1 nếu “su-root” thực thi câu lệnh, trong trường hợp ngược lại  =0 | | 0 | |
| 16 | Num\_root | | Số lượng truy cập tài khoản “root”, hoặc thao tác thực thi khi kết nối từ tài khoản “root” | | 0 | |
| 17 | Num\_file\_creations | | Số lượng file được tạo ra khi kết nối | | 0 | |
| 18 | Num\_shells | | Số lượng thực thi shell | | 0 | |
| 19 | Num\_access\_files | | Số lượng thao tác các file xử lý | | 0 | |
| 20 | Num\_outbound\_cmds | | Số lượng tập lệnh trong một phiên kết nối ftp | | 0 | |
| 21 | Is\_hot\_login | | =1 nếu đăng nhập thuộc danh sách “hot” hoặc admin  =0 nếu ngược lại | | 0 | |
| 22 | Is\_guest\_login | | =1 nếu đăng nhập là khách  = 0 nếu ngược lại | | 0 | |
| STT | Thuộc tính | | | Mô tả | | Ví dụ |
| 23 | Count | | | Số kết nối đến cùng địa chỉ đích như kết nối hiện thời trong 2 lần kết nối trước | | 2 |
| 24 | Srv\_count | | | Số kết nối tới cùng dịch vụ (số cổng) như kết nối hiện tại trong vòng 2s | | 2 |
| 25 | Serror\_rate | | | Tỉ lệ phần trăm các kết nối hoạt động gán cờ flag(4), s0,s1,s2 hay s3 trong tập hợp các kết nối count(23) | | 0 |
| 26 | Srv\_serror\_rate | | | Gán khởi tạo cờ flag(4), s0, s1,s2 hoặc s3 trong tập hợp các kết nối (24) | | 0 |
| 27 | Rerror\_rate | | | Tỉ lệ Phần trăm các kết nối được kích hoạt gán cờ falg(4) REJ, trong số các kết nối tại biến đếm count(23) | | 0 |
| 28 | Srv\_rerror\_rate | | | Tỉ lệ phần trăm kết nối được kích hoạt gán cờ falg(4) REJ, trong các kết nối srv\_count(24) | | 0 |
| 29 | Same\_srv\_rate | | | Tỉ lệ phần trăm kết nối tới cùng một dịch vụ, trong các kết nối count(23) | | 1 |
| 30 | Diff\_srv\_rate | | | Tỉ lệ phần trăm kết nối đến các dịch vụ khác nhau, trong các kết nối của count(23) | | 0 |
| 31 | Srv\_diff\_host\_rate | | | Tỉ lệ phẩn trăm của kết nối đến các máy đích khác nhau, trong các kết nối trong srv\_count(24) | | 0 |
| **STT** | | | **Thuộc tính** | | **Mô tả** | | **Ví dụ** |
| 32 | | | Dst\_host\_count | | Số lượng kết nối đến cùng địa chỉ IP | | 150 |
| 33 | | | Dst\_host\_srv\_count | | Số lượng kết nối đến cùng một cổng | | 25 |
| 34 | | | Dst\_host\_same\_srv\_rate | | Tỉ lệ kết nối đến cùng một dịch vụ, trong các kết nối của dst\_host\_count (32) | | 0.17 |
| 35 | | | Dst\_host\_diff\_srv\_rate | | Tỉ lệ kết nối đến các dịch vụ khác nhau, trong các kết nối của dst\_host\_count (32) | | 0.03 |
| 36 | | | Dst\_host\_same\_srv\_port\_rate | | Tỉ lệ kết nối đến cùng một cổng, trong các kết nối của dst\_host\_srv\_count (33) | | 0.17 |
| 37 | | | Dst\_host\_srv\_diff\_host\_rate | | Tỉ lệ kết nối đến các máy đích khác nhau, trong các kết nối của dst\_host\_srv\_c ount (33) | | 0 |
| 38 | | | Dst\_host\_serror\_rate | | Tỉ lệ kết nối được kích hoạt cờ flag(4), s1,s2 hoặc s3, trong các kết nối của dst\_host\_count (32) | | 0 |
| 39 | | | Dst\_host\_srv\_serror\_rate | | Tỉ lệ kết nối được kích hoạt cờ flag(4), s1,s2 hoặc s3, trong các kết nối của dst\_host\_srv\_count (33) | | 0 |
| 40 | | | Dst\_host\_rerror\_rate | | Tỉ lệ kết nối được kích hoạt cờ flag (4) REJ, trong các kết nối của dst\_host\_count (32) | | 0.05 |
| 41 | | | Dst\_host\_srv\_rerror\_rate | | Tỉ lệ kết nối được kích hoạt cờ flag (4) REJ, trong các kết nối của dst\_host\_srv\_count (33) | | 0 |

1. ***Các tấn công trong tập dữ liệu NSL- KDD.***

Dữ liệu trong bộ dữ liệu NSL- KDD được dán nhãn là bình thường hoặc tấn công, các tấn công trong tập dữ liệu NSL- KDD được loại thể hiện dưới bảng 3.4

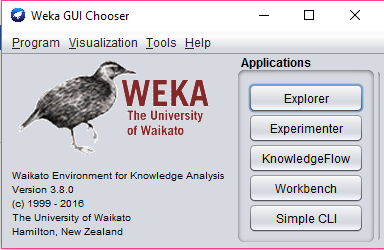
Bảng 3. 4: Các tấn công trong tập dữ liệu NSL-KDD

|  |  |
| --- | --- |
| **Tấn công** | **Kiểu tấn công** |
| DoS | Back,Land,Neptune,Pod,Smurf,Teardrop,Mailbomb,Processtable,Udpstorm,Apache2,Worm |
| Probe | Satan,IPsweep,Nmap,Portsweep,Mscan,Saint |
| R2L | Guess\_password,Ftp\_write,Imap,Phf,Multihop, Warezmaster,Xlock,Xsnoop,Snmpguess, Snmpgetattack,Httptunnel,Sendmail, Named |
| U2R | Buffer\_overflow,Loadmodule,Rootkit,Perl,Sqlattack,Xterm,Ps |

1. Công cụ thử nghiệm và chuẩn dữ liệu đầu vào
2. ***Giới thiệu công cụ Weka.***

+ Weka là một công cụ phần mềm viết bằng Java phục vụ lĩnh vực học máy và khai phá dữ liệu.

+ Tính năng chính của Weka: Một tập các công cụ tiền xử lý dữ liệu, các giải thuật học máy, khai phá dữ liệu, và các phương pháp thí nghiệm đánh giá; giao diện đồ họa dễ dàng thực hiện; và một môi trường cho phép so sánh các giải thuật học máy và khai phá dữ liệu.



Hình 3.3: Giao diện màn hình chính của Weka.

+ Các giao tiếp cơ bản.

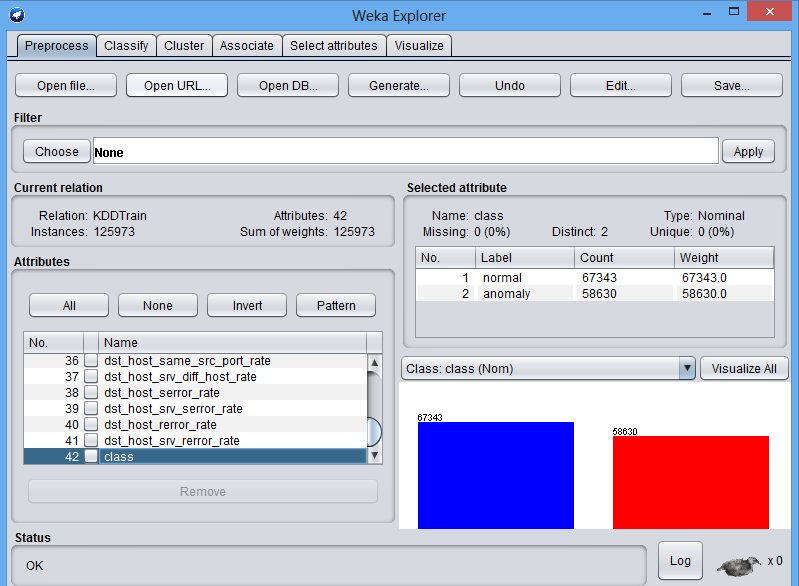
Simle CLI: giao diện đơn giản kiểu dòng lệnh.

Giao diện đồ họa: Cách giao tiếp này thông qua cửa sổ, bảng chọn, giao diện này rất phù hợp với việc triển khai nhanh chóng nhiệm vụ khai phá dữ liệu từ việc chuẩn bị dữ liệu cho đến việc xuất các kết quả của thuật toán. Chi tiế như dưới đây:

* *Explorer:* môi trường cho phép sử dụng tất cả các khả năng của weka để khai phá dữ liệu.
* *Experimeter*: Môi trường cho phép tiến hành các thí nghiệm và thực hiện các kiểm tra thống kê giữa các mô hình học máy.
* *KnowledgeFlow*: Môi trường cho phép tương tác đồ họa kiểu kéo thả để thiết kế các bước (các thành phần) của một thí nghiệm.

Giao diện lập trình: Giao diện lập trình sử dụng java của weka cho phép người dùng tùy biến, tự động hóa và tích hợp thuận tiện các pha khác nhau của quá trình khai phá dữ liệu vào ứn dụng.

+ Để chạy thực nghiệm này chúng ta chọn “Explorer”: giao diện cho phép sử dụng tất cả các cơ sở của nó bằng cách lựa chọn menu.



Hình 3.4: Môi trường explorer khi nạp một tập dữ liệu

+ Trong môi trường explorer có các menu:

*Preprocess:* để nạp và thay đổi (xử lý) dữ liệu làm việc.

*Classify:* để huấn luyện và kiểm tra các mô hình học máy.

*Clusters:* để học các nhóm từ dữ liệu (phân cụm).

*Associate:* để khám phá các luật kết hợp từ dữ liệu.

*Select attributes:* để xác định và lựa chọn các thuộc tính liên quan (quan trọng) nhất của dữ liệu.

Visualize Để xem (hiển thị) biểu đồ tương tác 2 chiều đối với dữ liệu.

+ Một số thuật toán được hỗ trợ bởi weka:

Naïve Bayes classifier and Bayesian networks.

Decision trees

Support vector machines.

Neural networks.

...

+ Lựa chọn các tùy chọn cho việc kiểm tra trong (test options):

*Use training set.* Bộ phân loại học được sẽ được đánh giá trên tập học.

*Supplied test set*. Sử dụng một tập dữ liệu khác (với tập huấn luyện) để cho việc đánh giá.

*Cross-validation*. Tập dữ liệu sẽ được chia đều thành k tập (folds) có kích thước xấp xỉ nhau, và bộ phân loại học được sẽ được đánh giá bởi phương pháp cross-validation.

*Percentage split.* Chỉ định tỷ lệ phân chia tập dữ liệu.

+ Khả năng tích hợp của weka: vì weka là một bộ công cụ học tập và khai thác dữ liệu bằng java thông qua một chương trình làm việc được cung cấp sẵn hoặc tự biên soạn bằng dòng lệnh. Weka cho phép bạn lấy dữ liệ từ các nguồn dữ liệu hiện có trình điều khiển JDBC. Với weka, bạn có thể thực hiện các bước: xử lý dữ liệu, phân nhóm, phân loại, hồi quy, các luật kết hợp. Bô công cụ weka được sử dụng rộng rãi và bây giờ hỗ trợ các khía cạnh dữ liệu lớn (big data) bằng cách liên kết với Hadoop (phần mềm framework hỗ trợ các ứng dụng phân tán dữ liệu) dể khai thác dữ liệu theo cụm.

1. ***Chuẩn dữ liệu đầu vào.***

Để có thể tiến hành thực nghiệm trên Weka thì điều bắt buộc là phải đưa dữ liệu về một trong các định dạng được Weka quy định sẵn như. csv,. arff...Trong luân văn này, sẽ sử dụng tập tin có định dạng ARFF (Atrribute-Regation File Format).

Mô hình ARFF có hai phần riêng biệt Phần đầu tiên là thông tin header, theo sau đó là các thông tin dữ liệu. Header của file ARFF chứa tên các mối quan hệ, một danh sách các thuộc tính và loại của chúng.

@relation 'KDDTest' 🡪 tên tập dữ liệu.

@attribute 'duration' real 🡪 thuộc tính và kiểu dữ liệu thuộc tính.

@attribute 'protocol\_type' {'tcp','udp', 'icmp'}.

@attribute 'is\_host\_login' {'0', '1'}.

@attribute 'is\_guest\_login' {'0', '1'}.

@attribute 'class' {'normal', 'anomaly'}.

Một số kiểu dữ liệu:

Numeric: dữ liệu dạng số Ví dụ: @ATTRIBUTE name numeric.

Nominal: dữ liệu rời rạc Ví dụ: @ATTRIBUTE class {normal,anomal}.

String: dữ liệu chuỗi Ví dụ: @ATTRIBUTE name string Date.

Date: dữ liệu kiểu ngày Ví dụ: @ATTRIBUTE discovered date.

1. Thực nghiệm và đánh giá
2. ***Thực nghiệm.***

Máy tính sử dụng cho quá trình chạy mô hình đánh giá có bộ xử lý Intel core i3, tần số 2.2GHz 2.2 GHz, 4.0 GB ram. Bộ công cụ weka phiên bản 3.8. Dữ liệu đầu vào cho thực nghiệm là tập KDDtrain+.arff chứa 125973 bản ghi, số thuộc tính 42 (cả nhãn). Các thuật toán được áp dụng là cây quyết định, svm, mạng neural nhân tạo.

Sử dụng việc chuẩn hóa dữ liệu mặc định.

SVM sử dụng hàm hai hàm nhân polykenel và puk.

MLP với 1,2, và 3 lớp ẩn.

Các bước thực hiện một mô hình phân lớp:

**Bước 1**. Nạp dữ liệu trong preprocess, nhấn chọn “choose”

Chọn tập dữ liệu huấn luyện: KDDTrain+.arff

Trong tab classify lựa chọn mô hình phân lớp tương ứng cho ở bảng sau:

Bảng 3. 5: Tóm tắt thao tác thử nghiệm mô hình.

|  |  |
| --- | --- |
| **Kỹ thuật phân lớp** | **Thao tác chọn** |
| Decision tree  (Cây quyết định) | J48  RandomForest  *Tham số mặc định* |
| SVM  (máy vector hỗ trợ) | *Choose→weka→filters-functions*  SMO  Trong cửa sổ *weka.gui.genericOjectEditor*  Chọn Kernel: lần lượt là puk và polykernel |
| Artificial Neural Networks  (mạng neural nhân tạo) | *Choose→weka→filters-functions*  *MultilayerPerceptron*  Trong cửa sổ *weka.gui.genericOjectEditor*  ở lần lượt thay ở *hide layer* giá trị  *a*  *o*  *30,20*  *30,20,10*  *Tương ứng với 1, 2,3 lớp ẩn* |

Tùy chọn kiểm tra trong test option: chọn phương pháp đánh giá cross- validation, với k=10.

Nháy start để chạy.

Lưu kết quả.

**Bước 2**. Chọn tùy chọn *Supplied test set* lần lượt open 3 tệp

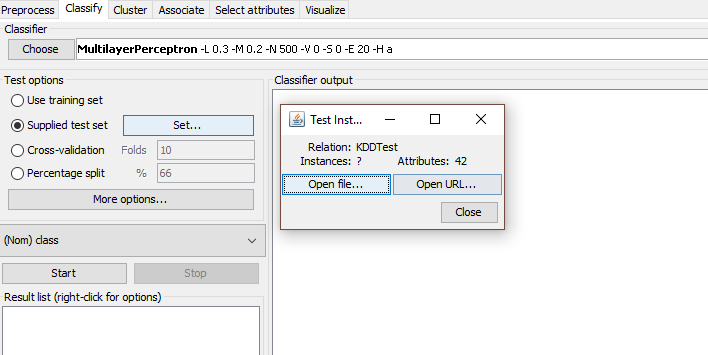
+ KDDTrain+\_20Percent.arff

+ KDDTest-21.arff

+ KDDTest+.arff

Để kiểm thử và lưu kết quả

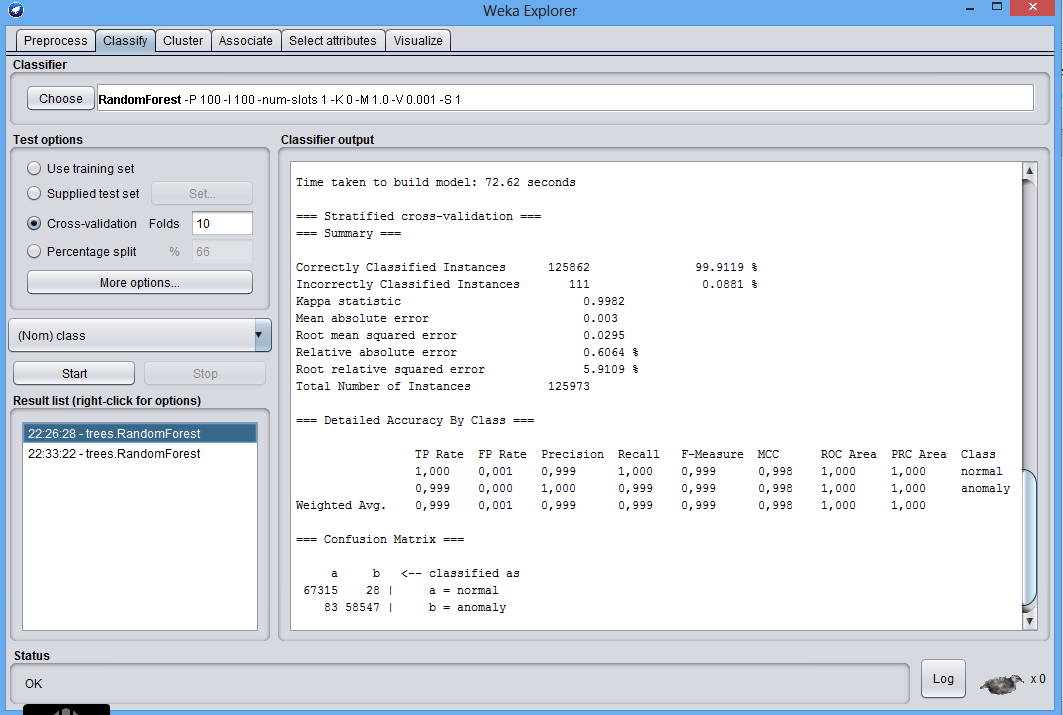
Thao tác thử nghiệm được mô tả ở hình 3.5 dưới đây



Hình 3.5: Kiểm thử mô hình

1. ***Kết quả***

Mô hình cây quyết định, bằng công cụ weka có kết quả như hình 3.6, và 3.7, 3.8



Hình 3.6: Phân lớp với Randomforest

Chi tiết kết quả phân lớp của randomforest (Hình 3.6), đánh giá cross- validation với k=10 trên tập huấn luyện

=== Classifier model (full training set) ===

=== Summary ===

Correctly Classified Instances 125862 99.9119 %

Incorrectly Classified Instances 111 0.0881 %

Mean absolute error 0.003

Root mean squared error 0.0295

Relative absolute error 0.6064%

Root relative squared error 5.9109 %

Total Number of Instances 125973

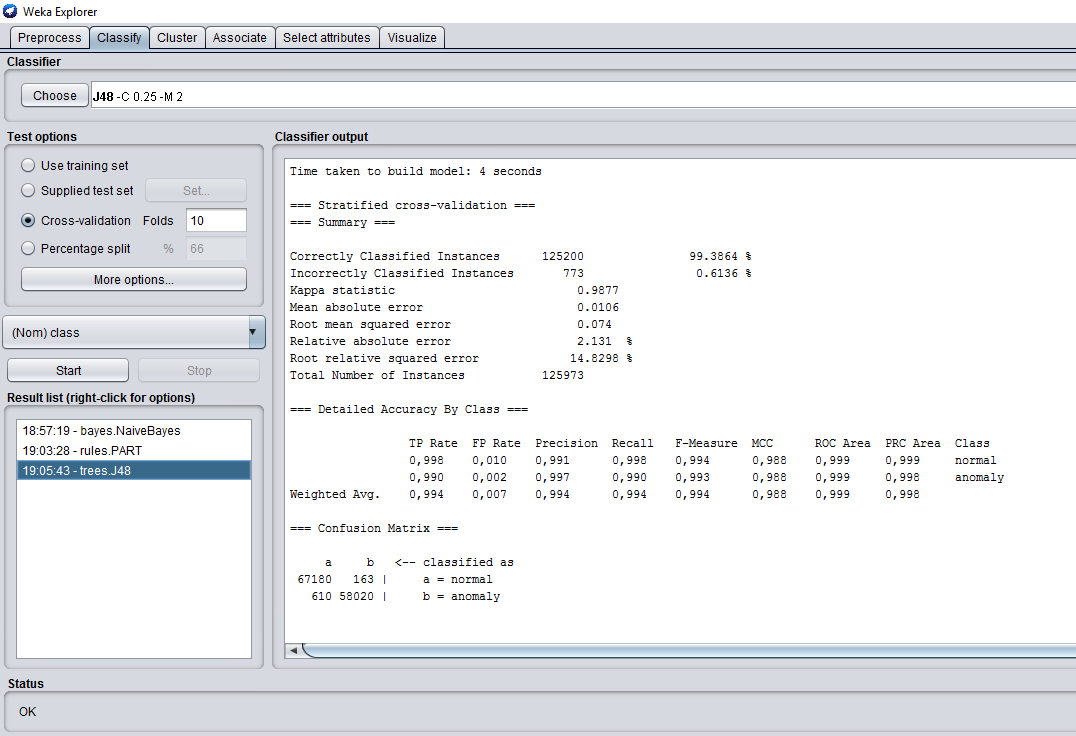
Chi tiết chỉ lệ phát hiện chính xác với từng lớp.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | ROC Area | PRC Area | Class |
| 1,000 | 0,001 | 0,999 | 1,000 | 0,999 | 0,998 | 1,000 | 1,000 | Normal |
| 0,999 | 0,000 | 1,000 | 0,999 | 0,999 | 0,998 | 1,000 | 1,000 | anormal |
| 0,999 | 0,001 | 0,999 | 0,999 | 0,999 | 0,998 | 1,000 | 1,000 | Weighted avg |

=== Confusion Matrix ===

|  |  |
| --- | --- |
| - Chương trình dự đoán đúng 67315/67343 mẫu (với lớp a- normal), 28 mẫu gán là anomal  -- Với lớp b-normal: chương trình dự đoán đúng 58547/58630 mẫu | a b <-- classified as  67315 28 | a = normal  83 58547 | b = anomaly |

Kết quả phân lớp của j48 (Hình 3.8), đánh giá cross- validation với k=10 === Classifier model (full training set) ===



Hình 3.7: Phân lớp với j48

Chi tiết

=== Summary ===

Correctly Classified Instances 125200 99.3864 %

Incorrectly Classified Instances 773 0.6136 %

Mean absolute error 0.0106

Root mean squared error 0.074

Relative absolute error 2,131%

Root relative squared error 14.8298 %

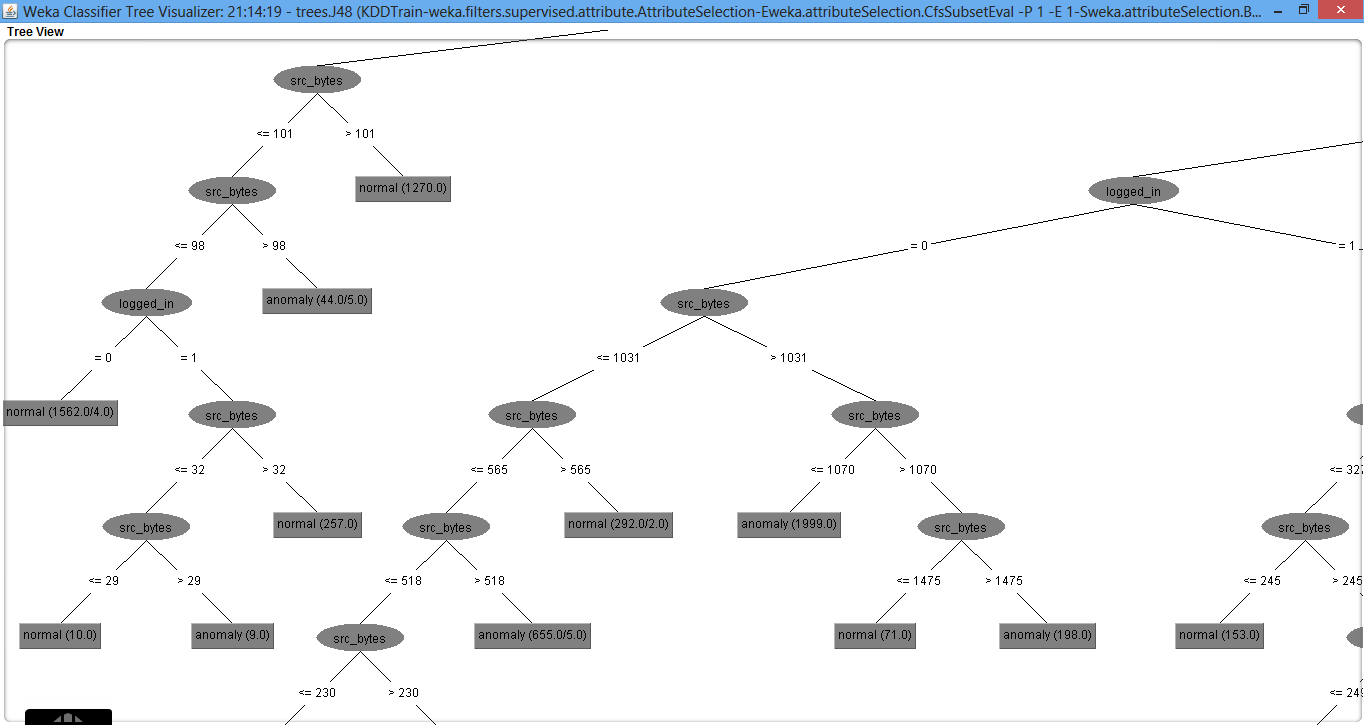
Total Number of Instances 125973

=== Detailed Accuracy By Class ===

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | ROC Area | PRC Area | Class |
| 0,998 | 0,010 | 0,991 | 0,998 | 0,994 | 0,988 | 0,999 | 0,999 | Normal |
| 0,990 | 0,002 | 0,997 | 0,990 | 0,993 | 0,988 | 0,999 | 0,998 | anormal |
| 0,994 | 0,007 | 0,94 | 0,994 | 0,994 | 0,988 | 0,999 | 0,998 | Weighted avg |

=== Confusion Matrix ===

|  |  |
| --- | --- |
| a b <-- classified as  67180 163 | a = normal  610 58020 | b = anomaly | - Với lớp a-normal,chương trình dự đoán đúng 67180 mẫu , 163 mẫu a gán là b-anomal  -- Với lớp b-normal: chương trình dự đoán đúng 58020, sai 610 |



Hình 3.8: Một phần cây quyết định J48

Kết quả phân lớp của Kết quả phân lớp của SMO- Polykernel, đánh giá cross- validation với k=10

=== Classifier model (full training set) ===

=== Summary ===

Correctly Classified Instances 122699 97.401 %

Incorrectly Classified Instances 3274 2.599 %

Kappa statistic 0.9477

Mean absolute error 0.026

Root mean squared error 0.1612

Relative absolute error 5.2229 %

Root relative squared error 32.32 %

Total Number of Instances 125973

Chi tiết độ chính xác phát hiện từng lớp.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | ROC Area | PRC Area | Class |
| 0,985 | 0,039 | 0,967 | 0,985 | 0,976 | 0,948 | 0,973 | 0,960 | Normal |
| 0,961 | 0,015 | 0,982 | 0,961 | 0,972 | 0,948 | 0,973 | 0,962 | Anormal |
| 0,974 | 0,028 | 0,974 | 0,974 | 0,974 | 0,948 | 0,973 | 0,961 | Weighted avg |

=== Confusion Matrix ===

|  |  |
| --- | --- |
| a b <-- classified as  66337 1006 | a = normal  2268 56362 | b = anomaly | - Với lớp a-normal,chương trình dự đoán đúng 66337 mẫu , 1006 mẫu a gán là b-anomal  - Với lớp b-normal: chương trình dự đoán đúng 56362, sai 2268 |

Kết quả phân lớp của Kết quả phân lớp của SMO- Puk, đánh giá cross- validation với k=10

=== Classifier model (full training set) ===

=== Summary ===

Correctly Classified Instances 125358 99.5118 %

Incorrectly Classified Instances 615 0.4882 %

Kappa statistic 0.9902

Mean absolute error 0.0049

Root mean squared error 0.0699

Relative absolute error 0.9811 %

Root relative squared error 14.0078 %

Total Number of Instances 125973

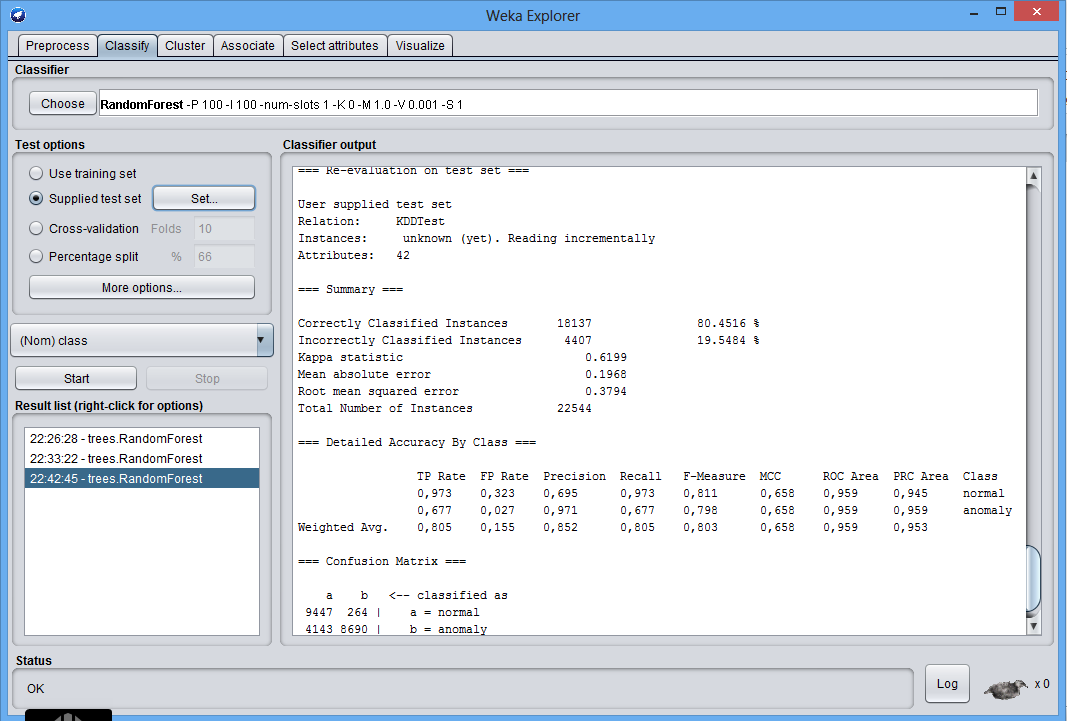
=== Detailed Accuracy By Class ===

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | ROC Area | PRC Area | Class |
| 0,997 | 0,007 | 0,994 | 0,997 | 0,995 | 0,990 | 0,995 | 0,992 | Normal |
| 0,993 | 0,003 | 0,997 | 0,993 | 0,995 | 0,990 | 0,995 | 0,993 | anormal |
| 0,995 | 0,005 | 0,995 | 0,995 | 0,995 | 0,990 | 0,995 | 0,993 | Weighted avg |

=== Confusion Matrix ===

|  |  |
| --- | --- |
| a b <-- classified as  67149 194 | a = normal  421 58209 | b = anomaly | - Với lớp a-normal,chương trình dự đoán đúng 67149 mẫu , 194 mẫu a gán là b-anomal  - Với lớp b-normal: chương trình dự đoán đúng 58209, sai 421 |

Với các tập kiểm thử kết quả, ta thu được như hình 3.9, 3.10



Hình 3.9: Kết quả kiểm thử phân lớp với radomforest

Kết quả phân lớp với smo-puk, đánh giá trên tập kiểm thử

=== Summary ===

Correctly Classified Instances 17720 78.6018 %

Incorrectly Classified Instances 4824 21.3982 %

Kappa statistic 0.5837

Mean absolute error 0.214

Root mean squared error 0.4626

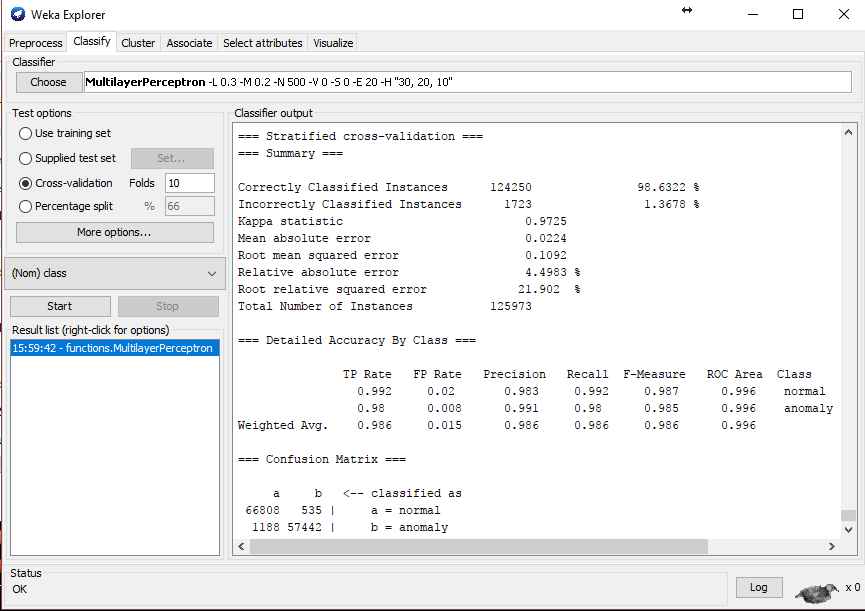
Total Number of Instances 22541

=== Detailed Accuracy By Class ===

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | ROC Area | PRC Area | Class |
| 0,950 | 0,338 | 0,680 | 0,950 | 0,793 | 0,619 | 0,806 | 0,668 | Normal |
| 0,662 | 0,050 | 0,945 | 0,662 | 0,779 | 0,619 | 0,806 | 0,818 | anormal |
| 0,786 | 0,174 | 0,831 | 0,786 | 0,785 | 0,619 | 0,806 | 0,753 | Weighted avg |

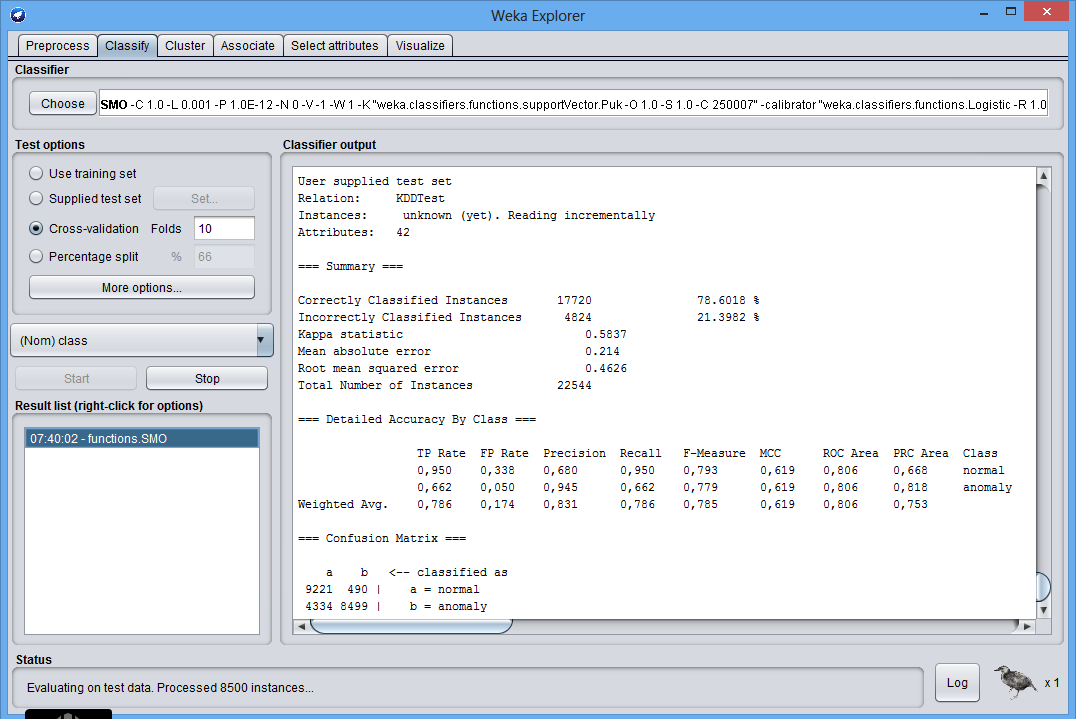
=== Confusion Matrix ===

|  |  |
| --- | --- |
| a b <-- classified as  9221 490 | a = normal  4334 8499 | b = anomaly | - Với lớp a-normal,chương trình dự đoán đúng 9221 mẫu , 490 mẫu a gán là b-anomal  - Với lớp b-normal: chương trình dự đoán đúng 8499, sai 4334 |

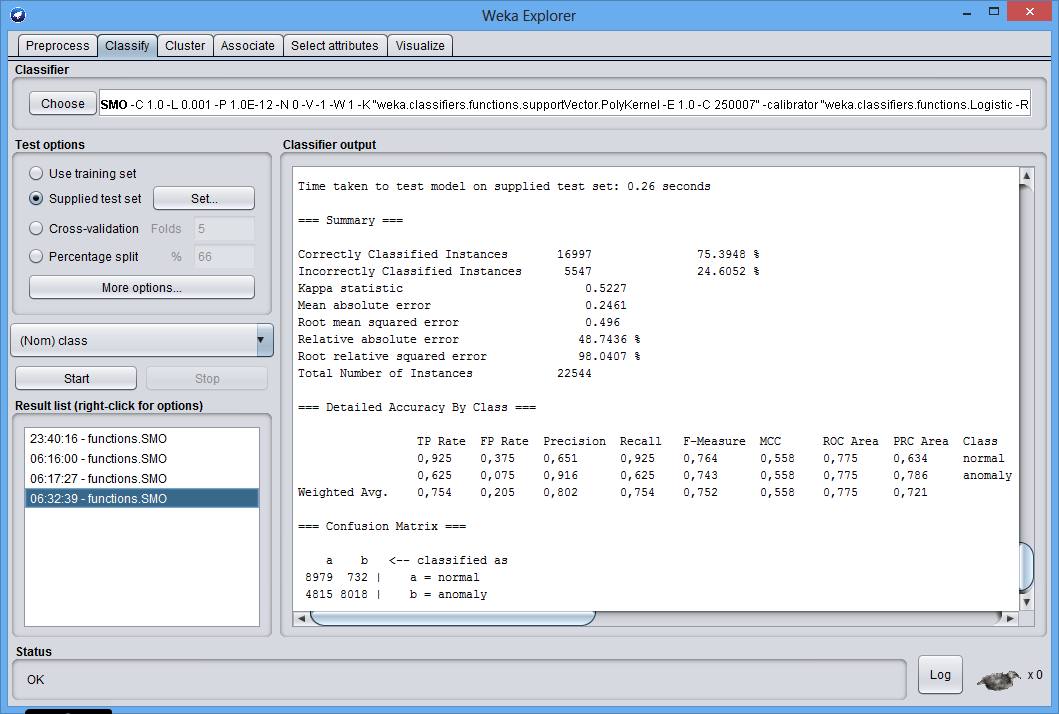


Hình 3.10: Phân lớp với MLP

* Mô hình phân lớp j48 và randomforest cho kết quả phân lớp chính xác >99%, với thuật toán random có tỉ lệ phát hiện chính xác lớn hơn. Kiểm định trên tập KDDtest cho kết quả trong lân cận của 80%, với tập KDDtest-21 là 64% và cây quyết định với số node lá là 129, bậc của cây là 221 (cây quyết định J48)
* Với mô hình SVM, thực nghiệm với hàm nhân *polykenel* mặc định và *puk,* được kết quả với tham số puk là lớn hơn,và thời gian huấn luyện lớn hơn, trung bình độ chính xác ~98%.
* Mô hình mạng neural nhân tạo: với 1 lớp ẩn, ta có thể chọn tham số tương ứng với số nút trong lớp ẩn là (a,o,t,i), tham số a cho kết quả lớn nhất với số nút ẩn bằng trung bình cộng của số thuộc tính và phân lớp. Thử nghiệm với 2 và 3 lớp ẩn tham số tương ứng là (30-20-10 và 30-20; 30-15). Ta được kết quả với tham số 30- 20 tốt hơn trong các tham số, cả về tỉ lệ phát hiện và thời gian huấn luyện.
* Kiểm thử các thuật toán bằng tập KDDtest ta được các kết quả phân loại chính xác trong khoảng từ 76% đến 80%, trên tập KDDtest-21 tỉ lệ thấp hơn từ 50% đên 64% ( Hình 3.10, 3.11, 3.12)

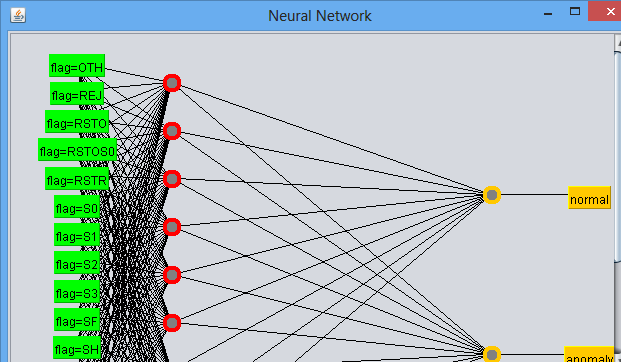


Hình 3.11: Kết quả kiểm thử phân lớp với SMO puk



Hình 3.12: Kết quả kiểm thử phân lớp với SMO-polykenel

Hình 3.13 Thể hiện một phần mạng neural (MLP-a)



Hình 3.13: Mạng neural nhân tạo

Kết quả chạy các mô hình được thể hiện tóm tắt ở bảng 3.6 (train ) và 3.7 (kiểm thử) dưới đây

Bảng 3. 6: Tóm tắt kết quả phân lớp các thuật toán trên tập KDDTrain

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuật toán** | **Tham số** | **Trung bình precision** | **Tỉ lệ phát hiện trung bình (%)** |
| RandomForest | -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1 0 | 0.999 | 99.9 |
| J48 | C 0.25 M 2 | 0.998 | 99.7 |
| SMO  Polykernel | weka.classifiers.functions.SMO -C 1.0 -L 0.001 -P  1.0E-12 -N 0 -V -1 -W 1 -K  "weka.classifiers.functions.supportVector.PolyKerne l -E 1.0 -C 250007" -calibrator "weka.classifiers.functions.Logistic -R 1.0E-8 -M -1 - num-decimal-places 4" | 0.974 | 97.4 |
| SMO  Puk | weka.classifiers.functions.SMO -C 1.0 -L 0.001 -P 1.0E-12 -N 0 -V -1 -W 1 -K "weka.classifiers.functions.supportVector.Puk -O 1.0 -S 1.0 -C 250007" -calibrator "weka.classifiers.functions.Logistic -R 1.0E-8 -M -1 -num-decimal-places 4" | 0.995 | 99.5 |
| MLP-a | -L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a | 0.985 | 98.5 |
| MLP- o | -L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H o | 0.978 | 97.7 |
| MLP -30-20-10 | L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H "30, 20, 10" | 0.986 | 98.6 |
| MLP 30-20 | -L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H "30, 20" | 0.992 | 99.2 |

Bảng 3. 7: Tóm tắt kết quả phân lớp các thuật toán trên tập KDDtest

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuật toán** | **Trung bình precision** | **Tỉ lệ phát hiện trung bình (%)** | **Ma trận sai số** |
| RandomForest | 0.852 | 80,5 | a b <-- classified as  9447 264 | a = normal  4143 8690 | b = anomaly |
| J48 | 0.858 | 81,5 | a b <-- classified as  9448 263 | a = normal  3990 8833 | b = anomaly |
| SMO-Polykernel | 0.802 | 75,4 | a b <-- classified as  8979 732 | a = normal  4815 8018 | b = anomaly |
| SMO- Puk | 0.831 | 78,6 | a b <-- classified as  9221 490 | a = normal  4334 8499 | b = anomaly |
| MLP-a | 0.817 | 77,7 | a b <-- classified as  9011 700 | a = normal  4324 8509| b = anomaly |
| MLP- o | 0.789 | 73,4 | a b <-- classified as  8940 771 | a = normal  5205 7628 | b = anomaly |
| MLP -30-20-10 | 0.835 | 77,6 | a b <-- classified as  9405 306 | a = normal  4741 8092 | b = anomaly |
| MLP 30-20 | 0.816 | 77,7 | a b <-- classified as  66881 462 | a = normal  488 58142 | b = anomaly |

* Biểu đồ so sánh kết quả phân lớp giữa các thuật toán

Hình 3.14: So sánh độ chính xác

Hình 3.15: Trung bình PRECISION

1. ***Đánh giá.***

Từ kết quả ở 3.4.2 ta thấy

* Các thuật toán đều cho kết quả phân lớp chính xác trên tập huấn luyện (lớn hơn 97%), tuy nhiên kiểm định trên tập kiểm tra thì kết quả nhỏ hơn, kết quả xung quanh 77%. Trong đó mô hình cây quyết định cho kết quả tốt hơn hai mô hình SVM và ANN, tỉ lệ phân loại chính xác lớn hơn 80%.
* Các thuật toán đều có kết quả rất thấp khi kiểm thử trên tập *KDDtest-21,* (tỉ lệ phân loại chính xác lớn hơn 50%, cao nhất là 64.9% phân lớp j48) cho nên dữ liệu đầu vào ảnh hưởng đến hiệu quả phân lớp của mô hình.
* Thuật toán SVM và mạng neural có thời gian xây dựng mô hình lớn (nhiều hơn 24h), các thuật toán cây ra quyết định có ưu điểm thời gian huấn luyện rất ngắn và hiệu quả cao, đặc biệt là khi xác định các hành vi sử dụng mạng bình thường (nomal).
* SVM có độ chính xác so sánh bằng hoặc tốt hơn các thuật toán dựa trên cây quyết định.
* Qua biểu đồ so sánh các kết quả, ta thấy mỗi thuật toán có thời gian thực hiện khác nhau,có mực độ chính xác riêng, nhưng chênh lệch nhau không nhiều.

**Để nâng cao hiệu qủa của các thuật toán ta cần**

Mỗi thuật toán có điểm mạnh với một số loại tấn công như SVM và ANN cho kết quả tốt khi phát hiện tấn công dos, probe..., cây quyết định cho kết quả cao khi phân lớp phát hiện tấn công hay bình thường, nên để nâng cao hiệu quả ta có thể sử dụng mô hình phân loại nhiều bước

* Phân loại hành vi bất thường
* Phân loại hành vi tấn công

KẾT LUẬN

1. Kết quả đạt được

Luận văn tiến hành nghiên cứu tổng quan về bài toán xâm nhập và phát hiện xâm nhập và thực nghiệm với tập dữ liệu sử dụng các thuật toán như cây quyết định, máy vector hỗ trợ, mạng neural nhân tạo để đánh giá.

Những kết quả chính mà luận văn đạt được:

* Nghiên cứu và tìm hiểu về bài toán xâm nhập và phát hiện xâm nhập, trình bày một số phương pháp xâm nhập và phát hiện xâm nhập đã có trước.
* Nghiên cứu và tìm hiểu về thuật toán cây quyết định, máy vector hỗ trợ, mạng neural nhân tạo
* Xây dựng mô hình huấn luyện và kiểm thử với bộ dữ liệu lấy được mô tả với các thuật toán tìm hiểu, thay đổi các tham số để được kết quả đánh giá chính xác nhất.

1. Hạn chế

Luận văn tập trung tìm hiểu và sử dụng một bộ dữ liệu NSL-KDD để đánh giá.

1. Hướng phát triển

* Cải thiện hiệu suất, tăng tốc độ xử lý dữ liệu với dữ liệu lớn.
* Xây dựng hệ thống hoàn chỉnh với tập dữ liệu lớn để tự động phát hiện xâm nhập với nhiều thuật toán khác nhau cho kết quả cao nhất.

DANH MỤC TÀI LIỆU THAM KHẢO

**Tài liệu Tiếng Anh**

1. A Study on NSL-KDD Dataset for IntrusionDetection System Based on ClassificationAlgorithms, international of advaced research in computer and communicaation engineerinvol. E, issue 6, june 2015.

<https://vi.scribd.com/document/342463237/A-Study-on-NSL-KDD-Dataset->pdf

1. Vipin Kumar, Himadri Chauhan, Dheeraj Panwar, “K-Means Clustering Approach to Analyze NSL-KDD Intrusion Detection Dataset”, International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-3, Issue-4, September 2013.
2. M. Shyu, S. Chen, K. Sarinnapakorn, and L. Chang, “A novel anomaly detection scheme based on principal mponent classifier,” Proceedings of the IEEE Foundations and New Directions of Data Mining Workshop, in conjunction with the Third IEEE International Conference on Data Mining (ICDM03), pp. 172–179, 2003.
3. Ghorbani, Lu.,.Tavallaee,.Network.Intrusion.Detection.and.Prevention.Con...
4. <http://khcn.cinet.vn/articledetail.aspx?articleid=1867&sitepageid=455#sthash.cy6gZnkj.dpbs>.

**Tài liệu Tiếng Việt**

1. Ứng dụng mạng neural nhân tạo trong phát hiện xậm nhập mạng dựa vào tần suất lời gọi hệ thống với bộ dữ liệu ADFA-LD. Nguyễn Việt Hùng Bộ môn An toàn thông tin Khoa Công nghệ Thông tin Học viện Kỹ thuật Quân sự.
2. Nhập môn trí tuệ nhân tạo. Thầy Từ Minh Phương.
3. https://vi.wikipedia.org
4. Bài giảng môn an toàn thông tin nâng cao, Thầy Hoàng Xuân Dậu, Học viện Công nghệ Bưu chính viễn thông