ĐẠI HỌC QUỐC GIA TP.HCM
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC & KỸ THUẬT MÁY TÍNH

LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC

MỘT CÁCH TIẾP CẬN HỒI QUY CHO BÀI TOÁN DỰ ĐOÁN SỐ NGÀY NẰM VIỆN

HỘI ĐỒNG: KHOA HỌC MÁY TÍNH
GVHD: TS. Nguyễn An Khương
GVPB: TS. Nguyễn Tiến Thịnh

—o0o—

SVTH: Nguyễn Huỳnh Huy (1511252)

TP. HỒ CHÍ MINH, 09/2020

Lời cam đoan

Tôi xin cam đoan rằng, luận văn tốt nghiệp "Một cách tiếp cận hồi quy cho bài toán dự đoán số ngày nằm viện" là công trình nghiên cứu của tôi dưới sự hướng dẫn của TS. Nguyễn An Khương, xuất phát từ nhu cầu thực tiễn và nguyện vọng tìm hiểu của bản thân tôi. Ngoại trừ các kết quả tham khảo từ các công trình khác đã ghi rõ trong luận văn, các nội dung trình bày trong luận văn này là kết quả nghiên cứu do chính tôi thực hiện và kết quả của luận văn chưa từng công bố trước đây dưới bất kỳ hình thức nào.

Thành phố Hồ Chí Minh, 09/2020

Nhóm tác giả

Lời cảm ơn

Đầu tiên, tôi xin dành lời cảm ơn chân thành nhất gửi đến thầy của tôi, tiến sĩ Nguyễn An Khương. Luận văn này không thể hoàn thành nếu thiếu sự hướng dẫn tận tâm của thầy. Thầy Khương là giảng viên hướng dẫn của tôi trong suốt thời gian năm năm học đại học. Ngoài các kiến thức chuyên môn, làm việc cùng thầy trong thời gian dài còn giúp tôi học được các đức tính cần thiết của một người làm khoa học: sự trung thực, sự cẩn thận, sự chính xác và tư duy phản biện. Vượt ra ngoài khuôn khổ học thuật, thầy luôn bên cạnh hỗ trợ tôi trong những thời điểm tôi cảm thấy khó khăn và bế tắc nhất.

Tôi xin cảm ơn những bạn bè và anh chị thân thiết đã cùng tôi tham gia dự án "Sử dụng các mô hình học máy để dự đoán số ngày nằm viện của bệnh nhân" trong suốt ba năm vừa qua. Đó là các bạn Văn Minh Hào, Nguyễn Thành Phương, chị Nguyễn Thị Ngọc Mai, anh Đào Nguyễn Quốc Vinh, anh Nguyễn Tấn Đức. Những kiến thức về thống kê, y khoa và kinh nghiệm xử lí dữ liệu học được từ các bạn, các anh chị trong quá trình tham gia dự án này đã giúp tôi hoàn thành tốt luận văn tốt nghiệp này.

Trong quá trình hoàn thành đề tài này, tôi nhận được sự giúp đỡ rất nhiệt tình của nhiều anh chị nghiên cứu sinh từ các trường đại học trên thế giới. Tôi xin chân thành cảm ơn chị Emma Rocheteau (nghiên cứu sinh đại học Cambridge) đã có những lời khuyên hữu ích khi cùng tôi thảo luận về độ khó của các điều kiện thí nghiệm trong nghiên cứu của tôi. Tôi xin chân thành cảm ơn hai anh Xuling(Shirly) Wang (nghiên cứu sinh đại học Toronto), Matthew McDermott (nghiên cứu sinh viện công nghệ Massachusetts). Những trao đổi với hai anh đã giúp quá trình trích xuất và xử lí dữ liệu của tôi diễn ra suôn sẻ.

Cuối cùng, tôi muốn dành những tình cảm trân trọng nhất gửi đến ba và mẹ của tôi, những người đã hi sinh rất nhiều để tôi có cơ hội học tập ở những môi trường tốt nhất. Con yêu ba mẹ rất nhiều!

Thành phố Hồ Chí Minh, 09/2020

Nhóm tác giả

Tóm tắt luận văn

Dự đoán chính xác thời gian nằm viện của bệnh nhân là một công việc mang lại nhiều lợi ích. Một mặt, thông tin dự đoán hỗ trợ bệnh viện trong việc quản lý cơ sở hạ tầng và nguồn nhân lực một cách hiệu quả. Mặt khác, thông tin này cũng giúp cho bệnh nhân và gia đình chủ động trong việc chuẩn bị đầy đủ về tài chính để chi trả viện phí. Dựa trên các nghiên cứu về bài toán dự đoán số ngày nằm viện trước đó, chúng tôi đề xuất hai cải tiến nhằm cải thiện kết quả dự đoán. Thứ nhất, chúng tôi đề xuất một nhóm đặc trưng dữ liệu mới (được trình bày chi tiết trong Mục 4.5), chúng tôi gọi là **group-diagnoses**. Đề xuất này của chúng tôi giúp giảm chỉ số **mean absolute error (MAE)** trên tập dữ liệu kiểm thử từ 5.5 ngày xuống còn 4.9 ngày (Tiểu mục 6.3.1). Thứ hai, chúng tôi đề xuất một mô hình hai giai đoạn (two-stage model) (Tiểu mục 4.8.2), với tên gọi **divide-classify-regress**. Đề xuất thứ hai này giúp dự đoán đúng 25 bệnh nhân sẽ nằm viện dài ngày trên tổng số 291 bệnh nhân nằm viện dài ngày (trên 30 ngày) (Tiểu mục 6.3.4).

Chúng tôi hi vọng các kết quả trong luận văn này sẽ đóng góp một phần nhỏ để góp phần giải quyết bài toán đem lại rất nhiều lợi ích thực tế này.

Mục lục

Danh sách hình vẽ

Danh sách bảng

Chương 1

Giới thiệu

1.1 Giới thiệu đề tài

Trong bối cảnh kinh tế ngày càng phát triển, nhu cầu chi tiêu cho chăm sóc y tế của người dân ngày càng tăng. Điều này được minh chứng thông qua các số liệu trong báo cáo năm 2019 của tổ chức y tế thế giới (WHO) về tình hình chi tiêu cho y tế của các quốc gia [1]. Theo đó, từ năm 2000 đến năm 2017, chi tiêu cho y tế trên toàn cầu đạt tốc độ tăng trưởng 3.9% một năm, trong khi đó tốc độ tăng trưởng bình quân hàng năm của nền kinh tế toàn cầu chỉ đạt 3.0%. Đặc biệt, trong cùng khoảng thời gian này, mức chi cho y tế đang tăng nhanh ở các nước có mức thu nhập thấp và thu nhập trung bình trên toàn cầu, với mức tăng khoảng 7.8% qua mỗi năm. Bên cạnh đó, báo cáo của WHO còn cho biết hiện còn hơn 35% chi tiêu y tế đến từ nguồn chi trả trực tiếp của người dân. Điều này dẫn đến hệ quả là trên thế giới có khoảng 100 triệu người bị đẩy vào tình trạng cực nghèo mỗi năm. Những con số khổng lồ đó đặt ra một vấn đề quan trọng cho hệ thống y tế: cần phải tối ưu hóa các khâu trong dịch vụ y tế để có thể giảm bớt chi phí. Một trong những biện pháp cơ bản nhất có thể được áp dụng là tăng sự hiệu quả trong khâu quản lí bệnh viện. Trong đó, việc ước lượng được chính xác thời gian nằm viện của bệnh nhân là một yêu cầu rất quan trọng. Điều này, về phía bệnh viện, giúp họ chủ động trong việc phân bố nguồn lực (bác sĩ, y tá, giường bệnh) cho việc điều trị và giảm chi phí cho bệnh nhân. Mặt khác, về phía bệnh nhân, việc dự đoán thời gian nằm viện chính xác sẽ giúp bệnh nhân và gia đình lên kế hoạch trước về tài chính để chuẩn bị cho việc nằm viện.

1.2 Mục tiêu của đề tài

Mục tiêu của đề tài này thể hiện qua tên của nó, chính là *"Một cách tiếp cận hồi quy cho bài toán dự đoán số ngày nằm viện"*. Chúng tôi phân tích, kết hợp nhiều thông tin liên quan đến bệnh nhân để dự doán số ngày nằm viện. Cụ thể, dữ liệu đầu vào là các thông tin cá nhân của bệnh nhân như tuổi, chế độ bảo hiểm của bệnh nhân, hình thức nhập viện, các chỉ số xét nghiệm của bệnh nhân, các chẩn đoán bệnh của bệnh nhân. Kết quả trả về là ước lượng thời gian nằm viện của bệnh nhân (tính theo ngày). Để đạt được mục tiêu trên, chúng tôi mong muốn xây dựng

hệ thống sử dụng các mô hình học máy (đặc biệt là hồi quy) để dự đoán chính xác số ngày nằm viện của bệnh nhân. Chúng tôi chỉ tận dụng các thông tin của bệnh nhân trong 24 giờ đầu tiên sau khi nhập viện. Điều này giúp tăng tính tổng quát của bài toán và giúp việc dự đoán có nhiều ý nghĩa thực tế.

1.3 Phạm vi của đề tài

Trong đề tài này, chúng tôi sử dụng cơ sở dữ liệu bệnh nhân được thu thập từ bệnh viện Beth Israel Deaconess Medical Center ở Boston, Hoa kỳ. Đây là cơ sở dữ liệu những bệnh nhân nhập viện trong thời gian từ năm 2001 đến năm 2012 của bệnh viện này. Chúng tôi chỉ nghiên cứu trên những bệnh nhân nhập viện và được đưa vào khoa hồi sức cấp cứu (ICU). Ngoài ra, chúng tôi chỉ nghiên cứu trên những bệnh nhân có thời gian nằm viện tối đa là 90 ngày. Ngoài những giới hạn bên trên, chúng tôi không có ràng buộc nào khác, điều này giúp bài toán của chúng tôi được thực hiện trong điều kiện tổng quát nhất có thể.

1.4 Cấu trúc của luận văn

Luận văn bao gồm bảy chương, có bố cục như sau

- **Chương 1:** Giới thiệu và đưa ra cái nhìn tổng quan về đề tài *Một cách tiếp cận hồi quy cho bài toán dự đoán số ngày nằm viện*.
- **Chương 2:** Trình bày các công trình liên quan đến bài toán dự đoán số ngày nằm viện của bệnh nhân. Từ đó rút ra được các ưu nhược điểm các phương pháp hiện thời. Trên cơ sở các nhận xét đó để tiến hành cải tiến.
- **Chương 3:** Cơ sở lý thuyết nền tảng cho các phương pháp, quá trình thực hiện đề tài.
- **Chương 4, 5, 6:** Các chương này lần lượt trình bày phương pháp nghiên cứu, quá trình thực hiện, các thí nghiệm và kết quả đạt được của quá trình nghiên cứu. Đây là phần trọng tâm của của luận văn.
- **Chương 7:** Tổng kết lại toàn bộ quá trình thực hiện, kết quả đạt được, những hạn chế và hướng mở rộng của đề tài trong tương lai.
- **Phụ lục:** Trình bày ý nghĩa các xét nghiệm được dùng trong nghiên cứu này.

Chương 2

Các công trình liên quan

Trong chương này, chúng tôi nêu ra và nhận xét một số công trình liên quan đến bài toán dự đoán số ngày nằm viện của bệnh nhân.

Về mặt phương pháp, các công trình liên quan đến bài toán dự đoán số ngày nằm viện có thể được chia làm hai loại chính: (1) sử dụng các mô hình phân lớp để quy bài toán về việc dự đoán các khoảng thời gian nằm viện của bệnh nhân (ví dụ: 0-3 ngày, 3-7 ngày, etc.), (2) sử dụng các mô hình hồi quy nhằm ước lượng một cách chính xác số ngày nằm viện của bệnh nhân. Ngoài hai cách tiếp cận trên, một số tác giả cũng đề xuất một số mô hình lai để tăng hiệu quả việc dự đoán.

Trong nghiên cứu của Hyunyoung Baek và cộng sự [2], nhóm tác giả đã phân tích các thông tin ảnh hưởng đến thời gian nằm viện của bệnh nhân, cụ thể bao gồm: số bệnh mà mỗi bệnh nhân được bác sĩ chẩn đoán, loại bảo hiểm mà bệnh nhân đó sử dụng, số lần phẫu thuật, số lần phải chuyển khoa, mức độ nghiêm trọng của bệnh nhân. Với các đặc trưng nói trên, nhóm tác giả đã sử dụng mô hình hồi quy tuyến tính để dự đoán số ngày nằm viện. Kết quả của họ đạt được có độ lỗi mean absolute error là 4,68 ngày. Ngoài ra, họ còn sử dụng bộ phân lớp rừng ngẫu nhiên (random forest) để phân loại bệnh nhân nằm dài ngày với độ chính xác của mô hình đạt được là 97,32%. Điểm yếu của công trình này theo chúng tôi có các vấn đề như sau. Trong bài báo này, nhóm tác giả không trình bày rõ ràng các đặc trưng mà họ sử dụng trong các mô hình. Ví dụ, đặc trưng *insurance type* là một biến nhận gía trị rời rạc, bao gồm nhiều loại bảo hiểm khác nhau, nhưng họ không thực hiện bước *one-hot encoding* mà lại sử dụng trực tiếp đặc trưng này trong mô hình. Điểm thứ hai là họ chỉ chia dữ liệu huấn luyện/kiểm thử theo tỉ lệ $80/20$ và không thực hiện bước validation, điều này có thể dẫn đến vấn đề overfiting khi huấn luyện mô hình. Cuối cùng, các tác giả chỉ trình bày kết quả phân loại được 97,32%, kết quả này tuy cao nhưng không có nhiều ý nghĩa, vì số lượng bệnh nhân nằm viện ngắn ngày rất nhiều so với số bệnh nhân nằm viện dài ngày, dẫn đến mô hình dự đoán bị thiên vị về phần dữ liệu bệnh nhân nằm viện ngắn ngày. Các tác giả nên dùng thêm một số độ đo khác với tình huống dữ liệu bị mất cân bằng như thế này.

Muhlestein và các cộng sự, trong [3] nghiên cứu thời gian nằm viện của bệnh nhân sau khi phẫu thuật u não. Họ đã phân tích và kết luận các yếu tố ảnh hưởng lớn đến thời gian nằm viện của bệnh nhân u não, bao gồm: có phải là phẫu thuật thiết yếu hay không (non-elective surgery), có bị viêm phổi trước khi phẫu thuật hay không, tỉ lệ lượng Natri bất thường trong cơ thể, có bị

sụt giảm cân nặng trước khi phẫu thuật hay không, chủng tộc. Trong số đó, đặc trưng cuộc phẫu thuật có phải thiết yếu hay không có ảnh hưởng lớn nhất đến thời gian nằm viện. Điều này rất hợp lí, vì những phẫu thuật loại này nhằm vào các bệnh nhân mà yêu cầu phẫu thuật là cực kì cần thiết đến mạng sống của họ. Đặc trưng có ảnh hưởng kế tiếp là bệnh nhân có bị viêm phổi trước khi vào phẫu thuật hay không. Đặc trưng này rất đáng lưu ý vì so với những bệnh nhân không mắc bệnh phổi và những bệnh nhân có bệnh phổi trước khi vào phẩu thuật u não, thời gian nằm viện của hai nhóm này lần lượt là 7.6 và 20.4 ngày. Về phương pháp, nhóm tác giả đã sử dụng phương pháp ensemble regression(tích hợp nhiều mô hình học máy hồi quy đơn lẻ để cho kết quả dự đoán tốt hơn) cho việc dự đoán số ngày nằm viện. Nhóm tác giả kết luận là phương phép ensemble regression giúp cải thiện kết quả, tuy nhiên điều này không rõ ràng vì họ không trình bày kết quả các mô hình baseline để so sánh.

Trong công trình [4], Whellanet và đồng sự đã nghiên cứu các yếu tố liên quan tới thời gian nằm viện của các bệnh nhân bị suy tim, thông qua các dữ liệu lâm sàng có sẵn tại thời điểm nhập viện. Nhóm tác giả đã kết luận rằng các bệnh nhân có thời gian nằm viện lâu thường mắc cùng lúc nhiều bệnh và mức độ nghiêm trọng cũng cao hơn so với các bệnh nhân có thời gian nằm viện ngắn.

Trong công trình của Gentimis và các đồng nghiệp [5], họ đã sử dụng mô hình phân lớp để dự đoán thời gian nằm viện của bệnh nhân trên bộ dữ liệu MIMIC-III. Tuy kết quả của họ khá tốt với độ chính xác khoảng 80%, điều kiện thí nghiệm trong nghiên cứu này lại khá đơn giản. Họ chỉ quan tâm phân loại xem một bệnh nhân có nằm viện quá 5 ngày hay không.

Nghiên cứu của Rouzbahman và các cộng sự [6] tập trung vào giả thiết: liệu nếu ta phân cụm nhóm bệnh nhân trước khi sử dụng các mô hình học máy để dự đoán, thì kết quả có cải thiện hay không. Để đạt được điều đó, họ sự dụng một mô hình hai giai đoạn. Ở giai đoạn đầu tiên, họ sử dụng thuật toán phân cụm K-means với rất nhiều giá trị k khác nhau để phân cụm bệnh nhân. Tiếp đó các mô hình học máy sẽ được huấn luyện riêng trên từng cụm dữ liệu bệnh nhân. Với một bệnh nhân trong tập kiểm tra, họ sẽ tính toán dùng khoảng cách Euclid để xác định bệnh nhân ấy có khả năng là thuộc cụm nào nhất. Sau khi xác định xong, mô hình học máy huấn luyện trên cụm dữ liệu đó sẽ được dùng để dự đoán với bệnh nhân trong tập kiểm tra được nói ở trên. Đây là một ý tưởng hay và thú vị. Vấn đề lớn nhất ở nghiên cứu này là ở việc họ sử dụng khoảng cách Euclid. Vì dữ liệu y tế thì thường có số chiều lớn và rất thưa (nhiều đặc trưng không có giá trị), mà đây là nhược điểm lớn của khoảng cách Euclid.

Trong công trình của Harutyunyan và cộng sự [7], họ tập trung vào giải thiết: liệu nếu kết hợp việc dự đoán số ngày nằm ICU với các yêu cầu khác, ví dụ như dự đoán bệnh nhân có tử vong hay không, thì kết quả dự đoán số ngày nằm viện có cải thiện hay không. Họ xây dựng một mô hình học nhiều nhiệm vụ một lúc (multi-task learning) dựa trên mô hình học sâu LSTM (long-short term memory). Kết quả dự đoán số ngày nằm ICU của họ với mô hình multi-task này đã cải thiện đáng kể so với khi dùng từng mô hình đơn lẻ. Tuy nhiên, một lưu ý ở đây là họ dự đoán số ngày nằm ICU chứ không phải dự đoán số ngày nằm viện Đây là một bài toán dễ hơn bài toán dự đoán số ngày nằm viện vì thông thường số ngày nằm viện sẽ lớn hơn số ngày nằm ICU rất nhiều.

Cuối cùng, chúng tôi thảo luận về hai luận văn (cùng làm về bài toán dự đoán thời gian nằm viện của bệnh nhân) có liên quan đến nghiên cứu của chúng tôi. Luận văn thứ nhất, “Dự đoán thời gian nằm viện bằng học máy”, được thực hiện bởi anh Đào Nguyễn Quốc Vinh, sinh viên khóa 2014 ngành Khoa học và Kỹ thuật máy tính, Đại học Bách Khoa thành phố Hồ Chí Minh. Luận văn thứ hai có tên “Ứng dụng học máy trong việc dự đoán thời gian nằm viện”. Tác giả của luận văn thứ hai là chị Nguyễn Thị Ngọc Mai, học viên cao học trường đại học Công Nghệ thành phố Hồ Chí Minh (HUTECH). Cả hai nghiên cứu này cùng nằm trong dự án "Dự đoán số ngày nằm viện của bệnh nhân", được tiến hành từ năm 2017-2020 dưới sự hướng dẫn của tiến sĩ Nguyễn An Khương. ngoài ra, hai luận văn nói trên đều làm việc trên bộ dữ liệu bệnh án các bệnh nhân có thời gian nhập viện ừ tháng 12 năm 2013 đến tháng 7 năm 2017 của bệnh viện Thống Nhất, thành phố Hồ Chí Minh. Nội dung của hai luận văn trên đều đề xuất chia thời gian nằm viện của bệnh nhân thành nhiều khoảng nhỏ (0-3 ngày, 3-7 ngày, 7-15 ngày, 15 đến 30 ngày, và nhóm có thời gian nằm viện trên 30 ngày), sau đó sử dụng các mô hình học máy multi-class classification để dự đoán khoảng thời gian nằm viện của bệnh nhân. Kết quả phân lớp của các mô hình học máy được dùng trong hai nghiên cứu này còn rất khiêm tốn, với độ chính xác đều dưới 50%. Kết quả này có nguyên nhân chính đến từ sự mất cân bằng nghiêm trọng giữa các lớp trong bộ dữ liệu. Khác với hai luận văn được đề cập ở trên, trong nghiên cứu này, chúng tôi mô hình hóa bài toán dự đoán số ngày nằm viện dưới dạng một bài toán hồi quy. Ngoài ra, chúng tôi làm việc trên toàn bộ cơ sở dữ liệu các bệnh nhân, không giới hạn loại bệnh. Trong khi đó, hai luận văn trước đó chỉ giới hạn nghiên cứu những bệnh nhân bị mắc bệnh tiểu đường. Điểm khác biệt cuối cùng giữa nghiên cứu của chúng tôi so với hai luận văn đã đề cập bên trên là về tập dữ liệu. Khác với họ, chúng tôi sử dụng cơ sở dữ liệu MIMIC-III, một cơ sử dữ liệu mở rất phổ biến trong cộng đồng nghiên cứu y khoa.

Trên đây chúng tôi đã tóm tắt quá trình nghiên cứu các công trình của các tác giả. Từ đây chúng tôi rút ra được những phương pháp tiếp cận, các đặc trưng và đề xuất thêm các đặc trưn gmới cho bài toán của mình. Phần so sánh các kết quả sẽ được chúng tôi trình bày cụ thể ở Chương 6.

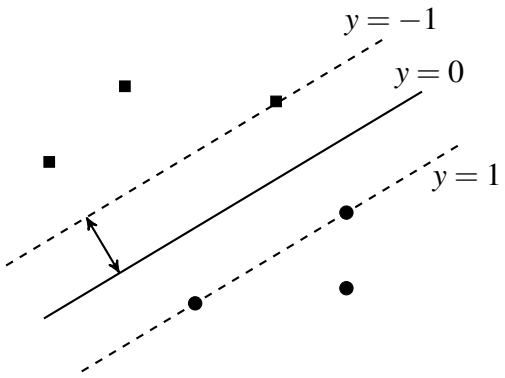
Chương 3

Cơ sở lý thuyết

3.1 Máy vector hỗ trợ

Máy vector hỗ trợ (*Support Vector Machine*, viết tắt là *SVM*) [8] được Vapnik giới thiệu lần đầu vào năm 1992 và đã trở thành một trong những giải thuật học máy phổ biến nhất bởi khả năng phân loại tốt trên những tập dữ liệu kích thước không quá lớn. SVM là mô hình học máy được dùng cho cả bài toán phân loại và hồi quy.

Giải thuật SVM xây dựng một mô hình phân loại có giám sát dựa trên tập dữ liệu huấn luyện đã được gán nhãn, từ đó có thể phân loại cho các dữ liệu mới. Mô hình SVM xem dữ liệu như là các điểm trong không gian, những dữ liệu thuộc các lớp khác nhau sẽ được phân chia rõ ràng và khoảng cách càng lớn càng tốt bởi ranh giới quyết định (*decision boundary*) như Hình 3.1.



Hình 3.1. Ranh giới quyết định $y = 0$ cùng với lề.

Dữ liệu đầu vào của mô hình được yêu cầu là phải phân chia tuyến tính (*linearly separable*). Tuy nhiên, không phải lúc nào dữ liệu cũng thỏa mãn yêu cầu trên. Với trường hợp dữ liệu không thể phân chia tuyến tính (*non-linearly separable*), ta áp dụng thủ thuật gọi là *Kernel Trick* để ánh xạ dữ liệu đó vào những chiều không gian khác mà ở đó dữ liệu thỏa mãn yêu cầu.

Mô hình tuyến tính dùng để phân loại dữ liệu gồm hai lớp có dạng tổng quát

$$y(x) = w^T \phi(x) + b, \tag{3.1}$$

với $\phi(x)$ là hàm ánh xạ điểm dữ liệu x vào một không gian nào đó. Tập huấn luyện gồm N vector $x_1, x_2, \ldots, x_N$, các mục tiêu tương ứng $t_1, t_2, ..., t_N$ với $t_n \in \{-1, 1\}$ và các điểm dữ liệu mới x được phân loại dựa vào dấu của $y(x)$.

Chúng ta giả định rằng tập dữ liệu huấn luyện là phân chia tuyến tính trong không gian được ánh xạ. Do đó, theo định nghĩa, tồn tại ít nhất một sự lựa chọn của các tham số w và b sao cho Hàm (3.1) thỏa mãn $y(x_n) > 0$ cho các điểm có $t_n = +1$ và $y(x_n) < 0$ cho các điểm có $t_n = -1$. Vì vậy, $t_n y(x_n) > 0$ với mỗi điểm dữ liệu trong tập huấn luyện.

Từ phương trình tổng quát (3.1), chúng ta nhận thấy có rất nhiều giải pháp để phân loại dữ liệu chính xác (tương ứng với các giá trị w và b khác nhau) nên cần tìm giải pháp mà sai số là nhỏ nhất có thể. SVM tiếp cận vấn đề này thông qua khái niệm gọi là lề (*margin*), được định nghĩa là khoảng cách nhỏ nhất giữa ranh giới quyết định và điểm dữ liệu bất kì.

Trong SVM, ranh giới quyết định được chọn sao cho cực đại hóa giá trị của lề thì sẽ cho ra sai số là cực tiểu [9].

Khoảng cách vuông góc từ điểm x đến siêu phẳng định nghĩa bởi $y(x) = 0$ ($y(x)$ có dạng (3.1)) được cho bởi $\dfrac{|y(x)|}{\|w\|}$. Chúng ta chỉ quan tâm tới các giải pháp mà tất cả các điểm dữ liệu được phân loại chính xác, sao cho $t_n y(x_n) > 0$ với mọi điểm n. Như vậy khoảng cách của một điểm x_n đến siêu phẳng được cho bởi

$$\frac{t_n y(x_n)}{\|w\|} = \frac{t_n(w^T \phi(x_n) + b)}{\|w\|}. \tag{3.2}$$

Lề được tính bởi khoảng cách vuông góc từ siêu phẳng đến điểm x_n gần nhất từ tập dữ liệu. Ta muốn tối ưu hóa các tham số w và b để cực đại hóa khoảng cách. Do đó lề cực đại được xác định bởi

$$\arg\max_{w,b} \left\{ \frac{1}{\|w\|} \min_n [t_n(w^T \phi(x_n) + b)] \right\}. \tag{3.3}$$

Chúng ta lấy $\dfrac{1}{\|w\|}$ ra bên ngoài (3.2) bởi vì w không phụ thuộc vào n. Việc giải bài toán này là cực kỳ phức tạp, do đó ta chuyển đổi nó về một dạng tương đương nhưng độ phức tạp thấp hơn. Để giải quyết vấn đề này, ta thực hiện co dãn (*rescale*) $w \to kw$ và $b \to kb$ để thiết lập

$$t_n(w^T \phi(x_n) + b) = 1 \tag{3.4}$$

cho điểm dữ liệu gần nhất so với siêu phẳng quyết định. Trong trường hợp này, tất cả các điểm sẽ thỏa mãn ràng buộc

$$t_n(w^T \phi(x_n) + b) \geq 1, \qquad n = 1, \ldots, N. \tag{3.5}$$

Trường hợp này được xem như sự biểu diễn chính tắc (*canonical representation*) của các điểm dữ liệu đối với siêu phẳng quyết định (*decision hyperplane*). Đối với những điểm dữ liệu thỏa mãn đẳng thức, ràng buộc được xem là *active*, còn lại thì được xem là *inactive*. Theo định nghĩa, luôn luôn có ít nhất một ràng buộc active do luôn luôn có một điểm dữ liệu gần nhất, và khi mà lề đạt

giá trị lớn nhất thì sẽ có ít nhất hai ràng buộc active. Bài toán tối ưu yêu cầu ta tìm giá trị lớn nhất của $\|w\|^{-1}$, tương đương giá trị nhỏ nhất của $\|w\|^2$, từ đó chúng ta giải quyết bài toán tối ưu

$$\arg\min_{w,b} \frac{1}{2}\|w\|^2, \tag{3.6}$$

với ràng buộc đã cho bởi (3.5). Hệ số $\frac{1}{2}$ được thêm vào nhằm thuận tiện trong việc tính đạo hàm trong quá trình giải bài toán tối ưu. Trong biểu thức (3.6), tham số b đã biến mất, tuy nhiên nó vẫn được ngầm định trong ràng buộc ở (3.5) bởi vì những thay đổi của w phải thông qua b.

Để giải quyết bài toán tối ưu có ràng buộc này, ta sử dụng các nhân tử Lagrange $a_n \geq 0$ với mỗi a_n ứng với một ràng buộc ở (3.5), ta có hàm Lagrange

$$L(w,b,a) = \frac{1}{2}\|w\|^2 - \sum_{n=1}^{N} a_n\{t_n(w^T\phi(x_n)+b)-1\}, \tag{3.7}$$

trong đó $a = (a_1, a_2, ..., a_N)^T$. Lấy đạo hàm riêng của $L(w,b,a)$ theo w và b, và cho chúng bằng 0, ta được hai biểu thức điều kiện:

$$w = \sum_{n=1}^{N} a_n t_n \phi(x_n), \tag{3.8}$$

$$0 = \sum_{n=1}^{N} a_n t_n. \tag{3.9}$$

Triệt tiêu w và b trong $L(w,b,a)$ bằng cách sử dụng các điều kiện trên cho ta biểu thức kép (*dual representation*)

$$\tilde{L}(a) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m k(x_n, x_m), \tag{3.10}$$

ở đây $k(x_n, x_m)$ là hàm Kernel được định nghĩa $k(x_n, x_m) = \phi(x_n)^T\phi(x_m)$, với a chịu các ràng buộc:

$$a_n \geq 0, \qquad n = 1, 2, ..., N, \tag{3.11}$$

$$\sum_{n=1}^{N} a_n t_n = 0. \tag{3.12}$$

Việc biến đổi từ biểu thức gốc (3.7) sang biểu thức kép (3.10) đã chuyển bài toán từ M biến (số chiều của w, hay số chiều không gian) về N biến (số chiều của a, hay số mẫu dữ liệu). Trong những trường hợp mà $M < N$, việc đưa bài toán về dạng biểu thức kép đem lại rất nhiều bất lợi. Tuy nhiên, nó cho phép mở rộng mô hình trong trường hợp là dữ liệu là không thể phân chia tuyến tính. Bên cạnh đó, mô hình phân loại SVM có thể sử dụng một cách hiệu quả với các không gian mà ở đó số chiều lớn hơn số điểm dữ liệu, bao gồm cả những không gian vô hạn chiều.

Để phân loại các điểm dữ liệu mới bằng mô hình vừa huấn luyện, ta sẽ xét dấu của $y(x)$. Thay w ở (3.8), ta được

$$y(x) = \sum_{n=1}^{N} a_n t_n k(x, x_n) + b. \tag{3.13}$$

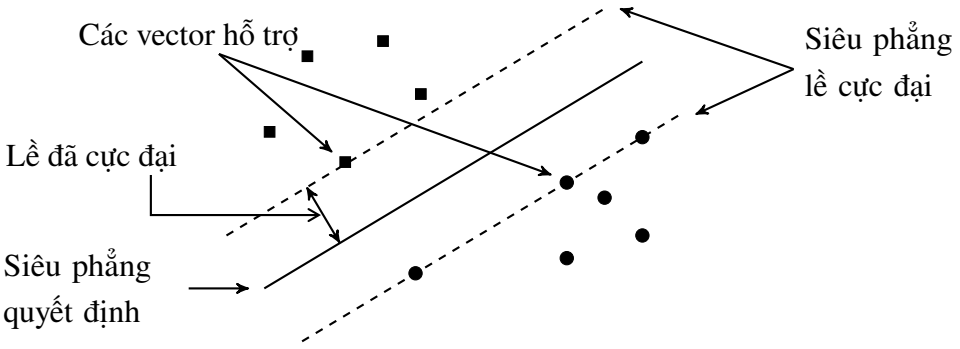
Theo phương pháp nhân tử Lagrange, biểu thức (3.7) cần thỏa mãn 3 điều kiện sau:

$$a_n \geq 0, \tag{3.14}$$

$$t_n y(x) - 1 \geq 0, \tag{3.15}$$

$$a_n \{t_n y(x) - 1\} = 0. \tag{3.16}$$

Ta thấy, với mỗi điểm dữ liệu , $a_n = 0$ hoặc $t_n y(x) = 1$. Những điểm mà có $a_n = 0$ thì sẽ không xuất hiện trong biểu thức ở (3.13) và do đó không có tác dụng phân loại. Những điểm còn lại được gọi là vector hỗ trợ (*support vector*) và do chúng thỏa mãn $t_n y(x) = 1$, chúng sẽ tương ứng với những điểm nằm ngay trên siêu phẳng lề cực đại (*maximal margin hyperplane*), minh họa ở Hình 3.2. Đặc trưng này là cốt lõi cho ứng dụng thực tế của SVM. Khi mô hình được huấn luyện, một số lượng lớn các điểm dữ liệu huấn luyện sẽ bị loại bỏ, chỉ có các vector hỗ trợ là còn giữ lại.



Hình 3.2. Minh họa các vector hỗ trợ.

Khi đã giải được bài toán (3.10) và tìm được a, ta sẽ xác định giá trị của tham số b bằng cách ghi nhận rằng bất kỳ vector hỗ trợ x_n nào đều thỏa mãn $t_n y(x) = 1$. Sử dụng (3.13), ta có

$$t_n \left(\sum_{m \in S} a_m t_m k(x_n, x_m) + b \right) = 1, \tag{3.17}$$

với S là tập hợp các chỉ số của các vector hỗ trợ. Nhân hai vế cho t_n, với $t_n^2 = 1$, biến đổi ta nhận được

$$b = \frac{1}{|S|} \sum_{n \in S} \left(t_n - \sum_{m \in S} a_m t_m k(x_n, x_m) \right). \tag{3.18}$$

3.2 Rừng ngẫu nhiên

3.2.1 Mô hình đóng gói

Cây quyết định CART mô tả ở trên đưa ra kết quả dự đoán dựa trên việc xét điều kiện tại các nút để rẽ nhánh cho tới khi tới nút lá, các điều kiện này phụ thuộc nhiều vào tập huấn luyện và do đó sẽ chạy tốt trên tập huấn luyện nhưng khi có sự thay đổi nhỏ trong tập huấn luyện cũng sẽ tạo nên một điều kiện rẽ nhánh khác, từ đó tạo ra một mô hình hoàn toàn khác biệt. Như vậy, CART sẽ có phương sai cao (*high-variance*) và độ lệch thấp (*low-bias*) khi càng phát triển độ sâu của cây. Mô hình đóng gói (*Bagging*) hay còn gọi là sự kết hợp mẫu (*bootstrap aggregation*) là một phương pháp giúp giảm phương sai nhưng không ảnh hưởng nhiều tới độ lệch của một mô hình học máy được đề xuất bởi Breiman [10].

Với một tập dữ liệu gồm n quan sát độc lập $Z_1, Z_2, \ldots, Z_n$ có phân phối chuẩn $\mathcal{N}(\mu, \sigma^2)$, khi đó, phương sai khi lấy trung bình các mẫu là σ^2/n. Do đó, một phương pháp để giảm phương sai và tăng độ chính xác của của một phương pháp học máy là lấy B tập huấn luyện từ tổng thể, xây dựng mô hình dự đoán riêng biệt $\hat{f}^i(x)$ cho mỗi tập huấn luyện $i, 1 \leq i \leq B$ và mô hình cuối cùng, $\hat{f}_{avg}(x)$, xác định kết quả dự đoán bằng cách lấy trung bình các mô hình này

$$\hat{f}_{avg}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^b(x).$$

Tuy nhiên số lượng tập huấn luyện trong thực tế thường không nhiều. Do đó, các tập huấn luyện cho từng mô hình được sinh ra dựa trên phương pháp lấy mẫu có hoàn lại (*bootstrapping*) [10]. Khi các mô hình được xây dựng cho mỗi tập huấn luyện là cây quyết định CART, thì những cây quyết định này đều được phát triển sâu đến hết mức có thể, không thực hiện cắt tỉa cây. Như vậy, các cây cho các tập huấn luyện này đều có độ lệch thấp và phương sai cao. Trong bài toán phân loại, nhãn được dự đoán cuối cùng của một quan sát $\hat{G}(x)$ được xác định bằng cách bỏ phiếu (*majority vote*) các kết quả của những mô hình $\hat{f}^i(x)$, hay nói cách khác chính là nhãn được dự đoán nhiều nhất của B cây đối với quan sát tương ứng

$$\begin{aligned}\hat{f}_{bag}(x) &= (p_1, p_2, \ldots, p_K),\\ \hat{G}(x) &= \arg\max_k \hat{f}_{bag}(x).\end{aligned}$$

Trong đó, p_k là tỉ lệ các cây dự đoán quan sát x có nhãn là lớp k.

Một hạn chế của các cây đã đóng gói của mô hình (*bagged trees*) là có sự tương quan giữa các cây với nhau. Ví dụ khi trong tập huấn luyện có một đặc trưng mạnh thì trong các cây, đặc trưng mạnh này có thể sẽ là lần chia tách đầu tiên. Điều này sẽ làm cho sự dự đoán của các cây sẽ tương tự nhau, hay nói cách khác các cây không độc lập với nhau. Gọi $\rho > 0$ là hệ số tương quan giữa hai cây bất kỳ trong các cây đã đóng gói của mô hình. Khi đó phương sai của trung bình các cây sẽ là

$$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2. \tag{3.19}$$

Như vậy, khi $B \to \infty$ thì số hạng thứ hai trong (3.19) sẽ mất đi và chỉ còn lại số hạng thứ nhất. Do đó, sự tương quan của các cây tạo nên sự hạn chế cho các cây đã đóng gói của mô hình.

3.2.2 Rừng ngẫu nhiên

Rừng ngẫu nhiên (*Random forest*) thực hiện cải tiến bagging trên cây bằng một tinh chỉnh nhỏ để giảm sự tương quan (*decorrelating*) của các cây. Giống như mô hình đóng gói, các cây quyết định cũng được xây dựng từ các tập huấn luyện được sinh ra bằng cách lấy mẫu có hoàn lại. Nhưng khi xây dựng các cây tại thời điểm tìm biến, điểm chia tách một mẫu ngẫu nhiên của $m \leq p$ đặc trưng được chọn từ tập đầy đủ của p đặc trưng để thực hiện tìm kiếm biến, điểm chia tách. Hay nói cách khác, việc chia tách chỉ thực hiện tìm kiếm trong m đặc trưng trên tổng số p đặc trưng. Như vậy, trung bình $(p-m)/p$ điểm chia tách có những đặc trưng mạnh sẽ không được xem xét, do đó tạo cơ hội cho các đặc trưng khác. Quá trình này được gọi là giảm sự tương quan của các cây trong rừng quyết định. Các bước xây dựng rừng quyết định có thể được viết lại như sau:

1. Với $b = 1,2,3,\ldots,B$
   (a) Tạo một mẫu Z^* bằng cách lấy mẫu có hoàn lại với kích thước N từ tập huấn luyện.
   (b) Xây dựng cây quyết định T_b cho mẫu Z^* vừa tạo ở (a) bằng cách chia nhị phân với các bước sau cho nút lá của cây hiện tại cho đến khi số mẫu thuộc nút là nhỏ hơn hoặc bằng n_{min}, một giá trị cho trước.
      i. Chọn ngẫu nhiên m biến từ p biến.
      ii. Tìm biến, điểm chia tách tốt nhất từ m biến.
      iii. Tách nút thành hai nút con.
2. Các cây đã được xây dựng xong trong rừng $\{T_b\}_1^B$.

Nhãn $\hat{C}_{rf}^B(x)$ của một quan sát x được xác định như sau: Gọi K là tập hợp các nhãn có trong dữ liệu, $\hat{C}_b(x)$ là nhãn mà cây thứ b trong rừng dự đoán cho x. Khi đó

$$\hat{f}_{rf}^B(x) = \{p_k = \frac{1}{B}\sum_{b=1}^{B} \mathbb{1}\{\hat{\mathbb{C}}_{\mathbb{b}}(\mathbb{x}) = \mathbb{k}\} | k \in K\},$$
$$\hat{C}_{rf}^B(x) = \arg\max_k \hat{f}_{rf}^B(x).$$

3.3 Gradient Boosting

Năm 1999, Friedman đề xuất một kỹ thuật học máy mới, gọi là *Gradient Boosting* (viết tắt là GB) [11]. Kỹ thuật này xây dựng nên mô hình bằng cách kết hợp các mô hình yếu hơn (gọi là *base learner* hoặc *weak learner*). Gọi tập huấn luyện mà ta có là $S = \{(x_1,y_1),(x_2,y_2),...,(x_N,y_N)\}$, mục tiêu của kỹ thuật này là tìm được một hàm $\hat{F}(x)$ xấp xỉ hàm F^* lý tưởng có thể làm cho hàm mất mát (*loss function*) $L(y,F(x))$ đạt giá trị kì vọng nhỏ nhất , cụ thể là:

$$\hat{F} = \arg\min_{F} E(x,y)[L(y,F(x)], \tag{3.20}$$

với $L(y,F(x))$ là hàm số khả vi với biến $F(x)$. Một ví dụ cụ thể cho hàm sai số là hàm sai số bình phương trong hồi quy Gauss, được tính bởi $L = C(y-F(x))^2$ với C là hằng số bất kì.
Giả sử ta có tập dữ liệu như trên cùng với một mô hình F. Ý tưởng của kỹ thuật *GB* là ta có thể cộng thêm một ước lượng h nào đó vào F, qua đó thu được mô hình $F+h$ với độ chính xác cao hơn. Trong trường hợp $F+h$ chưa thỏa mãn mong muốn của chúng ta, ta có thể cộng tiếp một ước lượng khác vào. Trong điều kiện lý tưởng, ta muốn tìm được ước lượng h sao cho $F(x)+h(x) = y$ hay $h(x) = y-F(x)$. Do đó, ta sẽ huấn luyện trên tập $X^{'} = \{(x_1, y_1 - F(x_1)), (x_2, y_2 - F(x_2)), ..., (x_N, y_N - F(x_N))\}$. Tuy nhiên, ta thấy rằng $y-F(x)$ chính là *gradient âm* của hàm sai số bình phương $\frac{1}{2}(y-F(x))^2$. Do đó, để tổng quát hóa ta sẽ huấn luyện h dựa trên gradient âm của $L(y,F(x))$. Giải thuật *GB* sẽ trải qua một số lượng bước M xác định. Ở bước khởi tạo $(m=0)$ ta sẽ gán

$$F_0(x) = \arg\min_{\gamma} \sum_{n=1}^{N} L(y_i, \gamma). \tag{3.21}$$

Sau đó ở mỗi bước lặp m từ 1 đến M, ta thực hiện như sau

1. Tính gradient âm $r_{im} = \frac{\partial}{\partial f} L(y_i, F(x_i))$ tại $F_{m\text{-}1}(x_i)$ với $i \in [1,N]$
2. Huấn luyện h với tập dữ liệu $X^{'} = \{(x_i, x_{im})\}$ với $i \in [1,N]$
3. Tính $\gamma_m = \arg\min_{\gamma} \sum_{n=1}^{N} L(y_i, F_{m\text{-}1}(x_i) + \gamma h_m(x_i)$
   Trong đó γ_m được gọi là *step length factor*
4. Tính $F_m(x) = F_{m\text{-}1}(x) + \gamma_m h_m(x)$

Trong trường hợp *base learner* ta sử dụng là một *cây quyết định*, kĩ thuật này được gọi là *gradient tree boosting*. Khi đó h sẽ có dạng:

$$h(x, (b_i, R_j)_1{}^J) = \sum_{j=1}^{J} b_j l(x \in R_j). \tag{3.22}$$

Ở đây $\{R_j\}_1{}^J$ là các vùng không gian giao nhau mà cùng nhau có thể bao phủ toàn bộ không gian giá trị của x, mỗi không gian được biểu diễn bằng một nốt lá. Hàm $l(.)$ trả về 1 nếu tham số của nó là đúng, ngược lại thì $l(.)$ trả về 0, $\{b_j\}_1{}^J$ là hệ số của *base learner* , giúp xác định ranh giới giữa các vùng. Vì các vùng không gian giao nhau, nên biểu thức (3.22) tương đương với khẳng định: nếu $x \in \mathrm{R}_j$ thì $h(x) = b_j$.
Khi đó, biểu thức

$$F_m(x) = F_{m\text{-}1}(x) + \gamma_m h_m(x) \tag{3.23}$$

sẽ trở thành

$$F_m(x) = F_{m\text{-}1}(x) + \gamma_m \sum_{j=1}^{J} b_{jm} l(x \in R_{jm}) \tag{3.24}$$

hay ta có thể viết (3.24) lại như sau:

$$F_m(x) = F_{m\text{-}1}(x) + \sum_{j=1}^{J} \gamma_{jm} l(x \in R_{jm}), \tag{3.25}$$

với $\gamma_{jm} = \gamma_j b_{jm}$. Hệ số γ_{jm} sẽ được tính bởi

$$\{\gamma_{jm}\}_1^J = \arg\min_{\{\gamma_{j_1}\}}^{J} \sum_{i=1}^{N} L(y_i, F_{m-1}(x)) + \sum_{j=1}^{J} \gamma_j l(x \in R_{jm})). \tag{3.26}$$

Do tính chất không giao nhau của các vùng, (3.27) sẽ được viết lại như sau

$$\gamma_{jm} = \arg\min_{\gamma} \sum_{x_i \in R_{jm}} L(y_i, F_{m-1}(x_i) + \gamma)). \tag{3.27}$$

3.4 Thư viện sử dụng

Chúng tôi sử dụng ngôn ngữ lập trình Python phiên bản 3.7 để thực hiện công việc. Các thư viện mã nguồn mở chúng tôi sử dụng: Scikit-learn 0.19.1 [12], NumPy 1.15.3 [13] và một số thư viện khác.

Scikit-learn là một thư viện mã nguồn mở được viết bằng ngôn ngữ lập trình Python. Thư viện này hiện thực hầu hết các mô hình học máy hiện tại bao gồm cả học có giám sát và học không có giám sát. Thư viện cũng cung cấp các công cụ cho quá trình đọc dữ liệu, tiền xử lý, trích xuất đặc trưng và có sẵn nhiều bộ dữ liệu mẫu cho các ví dụ. Đây là một thư viện dễ sử dụng, hiệu năng tốt cho làm việc nghiên cứu.

NumPy là một gói cơ bản cho các tính toán khoa học sử dụng Python. Thư viện này cung cấp các công cụ toán rất hữu ích: mảng N chiều, các hàm liên quan đến đại số tuyến tính,... Các tính toán trong Numpy đều đã được tối ưu để xử lý song song, tăng hiệu năng tính toán và tích hợp với cả các ngôn ngữ và hệ cơ sở dữ liệu khác.

Ngoài ra, chúng tôi còn sử dụng một số thư viện khác như *Pandas 0.21.0* [14] để xử lý các tập tin dữ liệu dưới dạng Dataframe (gồm tập huấn luyện và tập kiểm tra).

Chương 4

Phương pháp nghiên cứu

Với mục tiêu đề tài đã đặt ra ở Mục 1.2, trong chương này, chúng tôi sẽ trình bày phương pháp nghiên cứu cho bài toán.

4.1 Định nghĩa bài toán

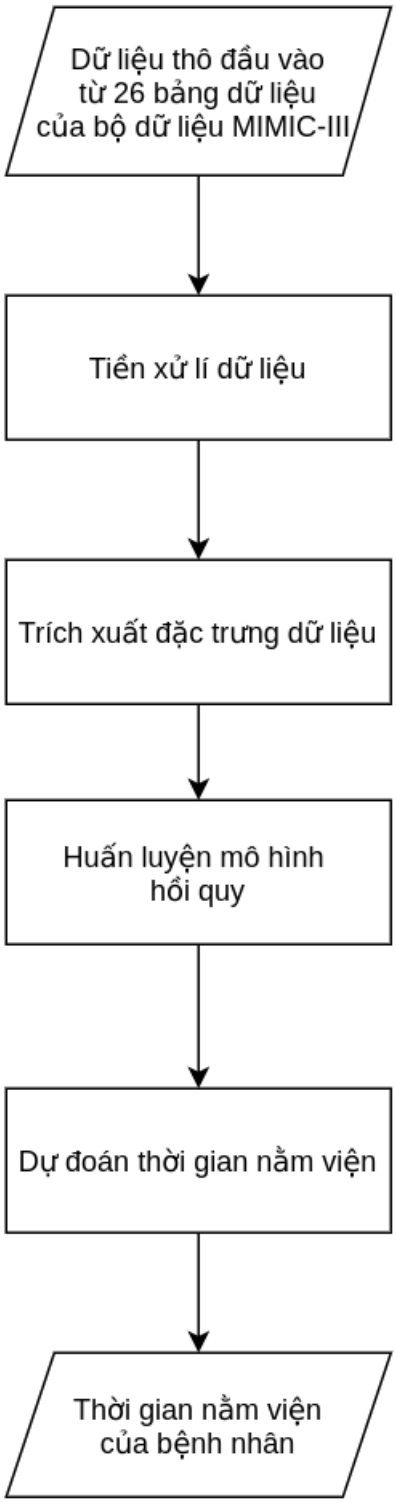
Tổng quan của bài toán: Cho dữ liệu đầu vào *D* là các thông tin của bệnh nhân khi nhập viện (ICU) trong 24 giờ đầu tiên, hệ thống sẽ dự đoán số ngày nằm viện của bệnh nhân.

- Dữ liệu đầu vào
  - **Thông tin bệnh nhân**: là các thông tin về tuổi, giới tính, loại bảo hiểm mà bệnh nhân sử dụng, loại hình điều trị ICU mà bệnh nhân được tiếp nhận để điều trị.
  - **Thông tin xét nghiệm**: là các chỉ số các xét nghiệm của bệnh nhân được thực hiện trong 24 giờ sau khi bệnh nhân tiếp nhận điều trị ICU.
  - **Chẩn đoán bệnh**: là dữ liệu về các loại bệnh mà bệnh nhân có thể đã mắc phải theo chẩn đoán của bác sĩ.
- Kết quả đầu ra: Số ngày nằm viện của bệnh nhân.

4.2 Kiến trúc hệ thống

Sau quá trình nghiên cứu, chúng tôi muốn đưa ra hệ thống cho việc *dự đoán thời gian nằm viện của bệnh nhân*. Hệ thống được minh họa bằng sơ đồ ở Hình 4.1 gồm các thành phần (*module*) khác nhau. Mỗi module đảm nhận một chức năng và giữa các module có một trình tự thực hiện nhất định, đầu ra của module này là đầu vào của module khác. Hệ thống nhận vào dữ liệu từ module đầu vào là thông tin bệnh nhân và các thông tin về bệnh án của bệnh nhân, thực hiện xử lý và trả về kết quả là số ngày nằm viện của bệnh nhân.

- **Dữ liệu đầu vào:** Dữ liệu đầu vào của hệ thống là bộ dữ liệu MIMIC-III, gồm tất cả thông tin về bệnh nhân trong mỗi lần nhập viện. Các thông tin này được lưu trữ trong 26 bảng.



Hình 4.1. Sơ đồ mô tả kiến trúc hệ thống dự đoán số ngày nằm viện.

- **Module 1 - Tiền xử lý dữ liệu:** Module này bao gồm nhiều công đoạn. Đầu tiên chúng tôi lấy ra các thông tin cần thiết của bệnh nhân từ 26 bảng dữ liệu thô ban đầu. Tiếp đến chúng tôi loại bỏ những dữ liệu bị sai lệch trong tập dữ liệu. Tiếp theo, chúng tôi lọc các thông tin xét nghiệm của bệnh nhân trong 24 giờ đầu tiên bệnh nhân tiếp nhận điều trị ICU. Cuối cùng, chúng tôi cần tính toán số ngày nằm viện thực tế của bệnh nhân dựa trên thông tin thời gian nhập viện và xuất viện trong bệnh án.
- **Module 2 - Trích xuất đặc trưng:** Sau khi thực hiện bước tiền xử lí dữ liệu, kết quả đầu ra của bước này sẽ được sử dụng để trích xuất các đặc trưng cho mô hình học máy. Các thông tin về xét nghiệm sẽ được lấy hai lần, lần đầu xét nghiệm và lần xét nghiệm cuối cùng trong ngày. Các đặc trưng nhận giá trị rời rạc sẽ được áp dụng kĩ thuật one-hot encoding. Đặc trưng về chẩn đoán bệnh của bác sĩ sẽ được tính toán dựa trên công thức mà chúng tôi đề xuất.
- **Module 3 - Huấn luyện mô hình học máy** Chúng tôi sử dụng các mô hình hồi quy *Gradient Boosting*, *Support Vector Machine* và *Random Forest* cho việc dự đoán. Trong bước này, chúng tôi thực hiện quá trình validation trên tập dữ liệu huấn luyện để kiểm tra tính tổng quát của mô hình.
- **Module 4 - Dự đoán thời gian nằm viện:** Với mô hình và bộ tham số đã được huấn luyện trong Module 3, chúng tôi thực hiện dự đoán thời gian nằm viện của các bệnh nhân.

4.3 Dữ liệu đầu vào

Trong nghiên cứu này, chúng tôi sử dụng bộ dữ liệu MIMIC-III. Bộ dữ liệu MIMIC-III bao gồm các dữ liệu y tế được thu thập tại Trung tâm y tế Beth Israel Deaconess ở Boston, Massachusetts, Hoa Kỳ. Bộ dữ liệu này chứa dữ liệu từ 38597 bệnh nhân khác nhau có thời gian nhập viện trong khoảng thời gian từ năm 2001 đến 2012. Đồng thời, nó được kết hợp từ hai cơ sở dữ liệu riêng biệt đó là cơ sở dữ liệu CareVue và cơ sở dữ liệu Metavision. Mục tiêu khi tạo ra MIMIC-III là nhằm cung cấp một cơ sở dữ liệu bệnh nhân đa dạng cho các loại phân tích y tế khác nhau.

Từ trước đến nay, dữ liệu y tế luôn là một trong những loại dữ liệu khó tiếp cận nhất đối với các nhà nghiên cứu vì những vấn đề về quyền riêng tư với các thông tin bệnh nhân. Với bộ dữ liệu MIMIC-III, các trở ngại đó đã được loại bỏ. Các thông tin cá nhân của bệnh nhân được mã hóa để chống lại việc tái định danh bệnh nhân, trong khi vẫn giữ lại được các thông tin quan trọng về mặt y học để phục vụ cho nghiên cứu. Kể từ khi được công bố vào năm 2016 [15], bộ dữ liệu MIMI-III đã được trích dẫn trong các nghiên cứu 1838 lần. Nó được sử dụng rộng rãi trong các nghiên cứu khác nhau về y học: từ dự đoán số ngày nằm viện của bệnh nhân [5] [7], dự đoán khả năng tử vong của bệnh nhân [16] [17], đến dự đoán phác đồ điều trị của bệnh nhân [7]. Có thể nói, không một bộ dữ liệu y tế mở nào khác có được sức ảnh hưởng sâu rộng đối với các nghiên cứu y học hơn cơ sở dữ liệu MIMIC-III.

Mặc dù MIMIC-III là một cơ sở dữ liệu mở, quyền truy cập chỉ được cấp sau khi hoàn thành một quy trình được xác định. Chúng tôi cần phải chứng minh việc sử dụng bộ dữ liệu MIMIC là

hợp lí, và chỉ phục vụ mục đích khoa học. Hơn nữa, danh tính của người yêu cầu truy cập bộ dữ liệu MIMIC-III, danh tính của giáo sư hướng dẫn, và thông tin về trường đại học nơi chúng tôi đang theo học cũng cần được cung cấp chi tiết để được xem xét. Trong quá trình hoàn thành luận văn này, chúng tôi đã hoàn thành chương trình đào tạo liên quan đến dữ liệu nói trên và được cấp quyền truy cập cơ sở dữ liệu MIMIC-III. Giấy chứng nhận được đặt tại phụ lục.

Bộ dữ liệu MIMIC-III gồm có 26 bảng, chứa đựng thông tin của tất cả bệnh nhân mỗi lần nhập viện. Chi tiết nội dung các bảng trong bộ dữ liệu MIMIC-III được thể hiện qua bảng Bảng 4.1 và bảng Bảng 4.2.

Bảng 4.1. Mô tả nội dung các bảng trong bộ dữ liệu MIMIC-III.

STT	Tên file	Kích thước	Nội dung
1	ADMISSIONS	(58976, 19)	Bảng ADMISSIONS đưa ra các thông tin liên quan đến một bệnh nhân nhập viện
2	CALLOUT	(34499, 24)	Bảng CALLOUT cung cấp thông tin về kế hoạch ra khỏi ICU của bệnh nhân
3	CAREGIVERS	(7567, 4)	Bảng này cung cấp các thông tin liên quan đến người chăm sóc. Ví dụ, nó sẽ xác định người chăm sóc là y tá, bác sĩ y khoa...
4	CHARTEVENTS	(330712483, 15)	CHARTEVENT chứa tất cả các dữ liệu biểu đồ có sẵn cho một bệnh nhân
5	CVEVENTS	(573146, 12)	Bảng CPTEVENT chứa danh sách các mã thuật ngữ của thủ tục hiện tại được lập hóa đơn cho bệnh nhân. Điều này có thể hữu ích để xác định xem các quy trình nhất định đã được thực hiện chưa
6	D-CPT	(134,9)	Bảng này cung cấp một số thông tin cấp cao về mã thuật ngữ thủ tục. Tuy nhiên, thông tin chi tiết cho các mã riêng lẻ là không có sẵn
7	D_ICD_DIAGNOSES	(14567, 4)	Bảng này xác định mã ICD-9 của các chẩn đoán bệnh của bệnh nhân
8	D_ICD_PROCEDURES	(3882,4)	Bảng này xác định mã ICD-9 cho các điều trị áp dụng cho bệnh nhân
9	D_ITEMS	(12487, 10)	Bảng D_ITEMS định nghĩa ITEMID, đại diện cho các phép đo trong cơ sở dữ liệu
10	D_LABITEMS	(753, 6)	D_LABITEMS chứa các định nghĩa cho tất cả ITEMID liên quan đến các phép đo trong phòng thí nghiệm trong cơ sở dữ liệu MIMIC-III
11	DATETIMEEVENTS	(4485937, 14)	Cơ sở dữ liệu chứa tất cả các phép đo thời gian về một bệnh nhân trong ICU
12	DIAGNOSES$_{I}CD$	(651047, 5)	Bảng này xác định mã ICD-9 để chẩn đoán
13	DRGCODES	(125557, 8)	Bảng này xác định mã HCFA-DRG và APR-DRG cung cấp thông tin liên quan đến chẩn đoán được ghi nhận chủ yếu cho mục đích thanh toán và hành chính
14	ICUSTAYS	(61532, 12)	Bảng này cung cấp thông tin liên quan đến thời gian nằm viện của ICU
15	INPUTEVENTS_CV	(17527935, 22)	Bảng này chứa dữ liệu của các sự kiện đầu vào chất lỏng (huyết thanh, thuốc tiêm tĩnh mạch, insulin,...) liên quan đến nguồn cơ sở dữ liệu Carevue trong các đợt ICU

Bảng 4.2. Mô tả nội dung các bảng trong bộ dữ liệu MIMIC-III (tiếp theo).

STT	Tên file	Kích thước	Nội dung
16	INPUT_EVENTS_MV	(3618991, 31)	Bảng này chứa các dữ liệu đầu vào của bệnh nhân
17	LABEVENTS	(27854055, 9)	Chứa tất cả các phép đo trong phòng thí nghiệm trong một thời gian nhất định của bệnh nhân, bao gồm cả dữ liệu bệnh nhân
18	MICROBIOLOGYEVENTS	(631726, 16)	Chứa thông tin vi sinh, bao gồm các xét nghiệm được thực hiện và độ nhạy cảm
19	NOTEEVENTS	(2083180, 9)	Bảng này chứa tất cả các ghi chú thủ công cho bệnh nhân bởi người chăm sóc
20	OUTPUTEVENTS	(4349218, 13)	Bảng này chứa dữ liệu đầu ra cho bệnh nhâ
21	PATIENTS	(46520, 8)	Bảng này chứa thông tin của các bệnh nhân nhập viện: ngày tháng năm sinh, giới tính,...
22	PRESCRIPTIONS	(4156450, 19)	Bảng này chứa các mục nhập đơn hàng liên quan đến thuốc, hay đơn thuốc
23	PROCEDUREEVENTS_MV	(258066, 25)	Bảng này chứa các quy trình cho bệnh nhân
24	PROCEDURES_ICD	(17527935, 22)	Bảng này chứa các thủ tục ICD cho bệnh nhân, đáng chú ý nhất là các thủ tục ICD-9
25	SERVICES	(73343, 6)	Bảng SERVICES mô tả dịch vụ kèm theo khi bệnh nhân được nhập viện. Các dịch vụ này có thể tự chọn hoặc phát sinh trong quá trình điều trị
26	TRANSFERS	(261897, 13)	Bảng này chứa các vị trí thực tế cho bệnh nhân trong suốt thời gian nằm viện

4.4 Tiền xử lý dữ liệu

Trong phần này, chúng tôi muốn trình bày chi tiết các bước tiền xử lý cụ thể cho bộ dữ liệu MIMIC-III.

4.4.1 Trích xuất các thông tin cần thiết từ 26 bảng dữ liệu ban đầu

Đầu tiên, chúng tôi sử dụng quy trình trích xuất thông tin được đề xuất bởi Wang và các cộng sự [18] để lấy ra các thông tin cần thiết của bệnh nhân để phục vụ cho việc dự đoán thời gian nằm viện. Lí do chúng tôi sử dụng quy trình này là vì: các xét nghiệm trong bộ dữ liệu MIMIC-III rất nhiều, tuy nhiên có nhiều xét nghiệm cùng một nhóm. Nhóm tác giả trong bài báo trên đã cộng tác với đội ngũ bác sĩ có chuyên môn để tìm cách gộp các xét nghiệm có ý nghĩa tương tự nhau lại thành một xét nghiệm duy nhất, do đó giúp giảm tính thưa của dữ liệu, đồng thời không làm mất đi ý nghĩa y tế của xét nghiệm. Trong mỗi lần nhập viện, bệnh nhân có thể có nhiều lần cần điều trị ICU, chúng tôi chỉ xét lần điều trị ICU đầu tiên lúc bệnh nhân vừa mới vào viện để làm cơ sở dự đoán số ngày nằm viện của mỗi bệnh nhân.

Giải thích cho cách lựa chọn trên của chúng tôi như sau: chúng tôi chỉ lựa chọn lấy thông tin về lần tiếp nhận điều trị ICU đầu tiên của bệnh nhân vì lí do là: các bệnh nhân trong bộ dữ liệu MIMIC-III tiếp nhận điều trị ICU lần đầu ngay sau khi nhập viện, sử dụng thông tin ở thời gian nằm giúp việc dự đoán được diễn ra sớm nhất có thể.

Sau bước này chúng tôi có được các thông tin về bệnh nhân và thông tin về các xét nghiệm của bệnh nhân được thể hiện qua bảng Bảng 4.3 bên dưới. Thực tế có nhiều thông tin được rút trích hơn, tuy nhiên trong bảng Bảng 4.3 chúng tôi chỉ trình bày một số thông tin quan trọng nhất liên quan đến bài toán của mình.

Các thông tin quan trọng trong bảng Bảng 4.3:

- **gender**: Giới tính bệnh nhân,
- **ethenicity**: chủng tộc của bệnh nhân,
- **age**: tuổi (bệnh nhân trên 90 tuổi đã được đổi thày 300 tuổi vì lí do riêng tư dữ liệu),
- **insurance**: loại xét nghiệm mà bệnh nhân sử dụng,
- **admitime**: thời gian bệnh nhân nhập viện,
- **dischtime**: thời gian bệnh nhân xuất viện,
- **intime**: thời gian bệnh nhân vào ICU lần đầu,
- **outtime**: thời gian bệnh nhân ra khỏi ICU lần đầu,
- **los icu**: thời gian nằm ICU lần đầu trong lần nhập viện,
- **admission type**: bệnh nhân nhập viện với hình thức gì, có phải là cấp cứu hay không,

- **first careunit**: lần điều trị ICU đầu tiên khi nhập viện của bệnh nhân thuộc loại ICU nào.

Bảng 4.3. Một số thông tin cơ bản của bệnh nhân.

Gender	Ethnicity	Age	Insurance	Admittime	Dischtime	Los ICU	Admission type	First careunit
M	WHITE	76.5268	Medicare	2101-10-20 19:08:00	2101-10-31 13:58:00	6.0646	EMERGENCY	MICU
F	WHITE	47.845	Private	2191-03-16 00:28:00	2191-03-23 18:41:00	1.6785	EMERGENCY	MICU
F	WHITE	65.9407	Medicare	2175-05-30 07:15:00	2175-06-15 16:00:00	3.6729	ELECTIVE	SICU
M	UNKNOWN	41.7902	Medicaid	2149-11-09 13:06:00	2149-11-14 10:15:00	5.3231	EMERGENCY	MICU
F	WHITE	50.1483	Private	2178-04-16 06:18:00	2178-05-11 19:00:00	1.5844	EMERGENCY	SICU
M	WHITE	72.3724	Medicare	2104-08-07 10:15:00	2104-08-20 02:57:00	7.6348	ELECTIVE	SICU
F	WHITE	39.8661	Medicaid	2167-01-08 18:43:00	2167-01-15 15:15:00	3.666	EMERGENCY	CCU
F	WHITE	47.4543	Private	2134-12-27 07:15:00	2134-12-31 16:05:00	2.071	ELECTIVE	CSRU
M	WHITE	50.8416	Private	2167-10-02 11:18:00	2167-10-04 16:15:00	1.2885	EMERGENCY	CCU
M	WHITE	300.003	Medicare	2108-08-05 16:25:00	2108-08-11 11:29:00	1.3017	EMERGENCY	TSICU

Bảng 4.4. Chỉ số xét nghiệm của bệnh nhân.

Subject id	Hours in	Alanine aminotransferase	Albumin	Albumin ascites	Albumin pleural	Albumin urine	Alkaline phosphate
3	0	25.0	1.8	Nan	Nan	Nan	73.0
3	1	Nan	Nan	Nan	Nan	Nan	Nan
3	2	Nan	Nan	Nan	Nan	Nan	Nan
3	3	Nan	Nan	Nan	Nan	Nan	Nan
3	4	Nan	Nan	Nan	Nan	Nan	Nan
3	5	Nan	Nan	Nan	Nan	Nan	Nan
3	6	Nan	Nan	Nan	Nan	Nan	Nan
3	7	Nan	Nan	Nan	Nan	Nan	Nan
3	8	Nan	Nan	Nan	Nan	Nan	Nan
3	9	Nan	Nan	Nan	Nan	Nan	Nan
3	10	Nan	Nan	Nan	Nan	Nan	Nan
3	11	Nan	Nan	Nan	Nan	Nan	Nan
3	12	Nan	Nan	Nan	Nan	Nan	Nan
3	13	Nan	Nan	Nan	Nan	Nan	Nan
3	14	Nan	Nan	Nan	Nan	Nan	Nan
3	15	Nan	Nan	Nan	Nan	Nan	Nan
3	16	Nan	Nan	Nan	Nan	Nan	Nan
3	17	Nan	Nan	Nan	Nan	Nan	Nan
3	18	Nan	Nan	Nan	Nan	Nan	Nan
3	19	Nan	Nan	Nan	Nan	Nan	Nan

Bảng 4.4 mô tả một phần các xét nghiệm có trong bộ dữ liệu MIMIC-III, chi tiết toàn bộ các xét nghiệm và ý nghĩa từng xét nghiệm được trình bày chi tiết trong Appendix A. Ở đây chúng tôi chỉ minh họa một số thông tin xét nghiệm, cụ thể như sau:

- **subject id**: Mã bệnh nhân,
- **hours in**: Thời gian lấy xét nghiệm tính từ lúc vào ICU. Ở đây, cứ cách một giờ, thì bệnh nhân sẽ được đo lại các chỉ số xét nghiệm. Dĩ nhiên, không phải xét nghiệm nào cũng được đo, và có nhiều xét nghiệm chỉ được đo một vài lần. Đó là lí do bảng 4.4 có rất nhiều giá trị trống (Nan),
- **alanine aminotransferase**: Alanine aminotransferase (ALT) là một loại men được tìm thấy ở trong tế bào gan và có vai trò hỗ trợ phân giải protein để cơ thể có thể dễ dàng hấp thụ. Khi gan có dấu hiệu viêm hoặc tổn thương, lượng ALT trong máu tăng lên, do vậy khi nghi ngờ bệnh về gan, bác sĩ thường sử dụng xét nghiệm này để chẩn đoán bệnh,
- **albumin**: Albumin là một thành phần protein quan trọng nhất của huyết thanh, chiếm đến 58-74% lượng protein toàn phần. Xét nghiệm này dùng để chẩn đoán các bệnh tiểu đường hay huyết áp cao, một số loại bệnh có ảnh hưởng xấu đến chức năng của thận,
- **albumin ascites**: Khi chỉ số albumin trong máu giảm, có thể dẫn đến tình trạng báng bụng, cổ trướng, tức là tình trạng ứ đọng dịch ở khoang phúc mạc. Đây là dấu hiệu củabệnh xơ gan.
- **albumin pleural**: Khi lượng albumin trong máu giảm, có thể là một trong những nguyên nhân dẫn đến tràn dịch màn phổi,
- **albumin urine**: Albumin urine là một xét nghiệm được thực hiện để đo lượng albumin trong nước tiểu. Một lượng nhỏ albumin trong nước tiểu là dâu hiệu cảnh báo sớm bệnh nhân có nguy cơ tổn thương thận,
- **Alkaline Phosphatase**: Xét nghiệm Alkaline Phosphatase được thực hiện để đo hoạt độ enzym phosphatase kiềm trong máu ở những bệnh nhân nghi ngờ mắc các bệnh về gan hoặc xương.

4.4.2 Tiền xử lí những phần dữ liệu bị sai số

Trong phần này chúng tôi tiên hành tiền xử lí những dữ liệu bị sai số sau khi thực hiện công đoạn trích xuất thông tin ở Tiểu mục 4.4.1. Sai số này có thể xảy ra trong quá trình ghi bệnh án hoặc trong quá trình thu thập dữ liệu.

Đầu tiên, chúng tôi phát hiện trong dữ liệu có nhiều bệnh nhân có thời gian bắt đầu điều trị ICU lần đầu tiên diễn ra trước thời gian nhập viện. Chúng tôi trích một phần dữ liệu thể hiện điều đó trong Bảng 4.5 bên dưới. Điều này có thể lí giải là nhiều bệnh nhân nhập viện trong tình trạng khẩn cấp (bị bệnh nặng, tai nạn hoặc thai phụ sinh sản) nên được đưa vào ICU trước khi

làm thủ tục nhập viện. Chúng tôi giải quyết những trường hợp này bằng cách thay thời gian nhập viện bởi thời gian vào ICU lần đầu tiên.

Bảng 4.5. Các bệnh nhân có thời gian nhập viện trước thời gian vào ICU lần đầu.

Subject id	Hadm id	ICU stay id	Admittime	Intime
35	166707	282039	2122-02-10 11:15:00	2122-02-10 09:39:59
347	119310	268893	2118-05-23 13:30:00	2118-05-23 10:56:34
373	104540	211665	2198-08-27 12:15:00	2198-08-27 09:17:10
409	105471	224525	2159-09-17 12:00:00	2159-09-17 10:33:00
487	160958	228805	2130-10-31 12:15:00	2130-10-31 11:04:31
796	127253	253353	2137-11-11 12:45:00	2137-11-11 10:23:13
810	168366	257899	2136-01-18 21:13:00	2136-01-18 01:10:26
957	145966	292666	2162-03-22 11:45:00	2162-03-22 10:42:04
1108	108891	208969	2198-04-16 16:00:00	2198-04-16 13:06:00
1417	105034	236094	2129-06-17 23:30:00	2129-06-17 22:48:53

Chúng tôi còn quan sát từ dữ liệu có một số bệnh nhân có thời gian nhập viện xảy ra sau thời gian xuất viện. Điều này là vô lý, do đó chúng tôi loại bỏ những bệnh nhân này khỏi bộ dữ liệu. Các trường hợp bệnh nhân này được minh họa qua Bảng 4.6 dưới đây.

Bảng 4.6. Các bệnh nhân có thời gian nhập viện xảy ra sau thời gian xuất viện.

Subject id	Hadm id	ICU stay id	Admittime	Dischtime
181	102631	246694	2153-10-12 09:49:00	2153-10-12 06:29:00
516	187482	213831	2197-07-31 20:18:00	2197-07-31 01:10:00
1381	181430	291798	2189-01-02 14:25:00	2189-01-02 12:00:00
2420	135098	227111	2184-12-01 19:28:00	2184-12-01 16:50:00
2677	108011	209967	2128-04-16 12:28:00	2128-04-16 12:00:00
2858	190088	293048	2108-09-25 15:29:00	2108-09-25 12:00:00
3229	161198	254078	2134-11-30 18:19:00	2134-11-30 12:00:00
3915	198555	240959	2178-04-07 16:31:00	2178-04-07 12:00:00
5452	133470	205244	2142-02-20 13:40:00	2142-02-20 12:00:00
5567	182804	233453	2181-07-09 15:31:00	2181-07-09 12:00:00
6832	164068	208677	2142-05-23 13:20:00	2142-05-23 12:00:00
7343	105061	256947	2116-10-09 22:26:00	2116-10-09 12:00:00
8604	102784	218796	2110-01-06 15:15:00	2110-01-06 12:00:00
8754	171918	252180	2169-05-15 16:58:00	2169-05-15 02:03:00
9010	129762	235982	2117-11-26 17:27:00	2117-11-26 12:00:00

Tiếp theo, chúng tôi phát hiện có 373 bệnh nhân có tổng thời gian nằm viện nhỏ hơn tổng thời gian nằm ICU. Điều này có thể xảy ra vì như phân tích Bảng 4.5 ở trên, một số bệnh nhân có tình trạng khẩn cấp phải vào điều trị ICU trước khi làm thủ tục nhập viện. Tuy nhiên, theo chúng tôi với những bệnh nhân có chênh lệch giữa tổng thời gian nằm viện và thời gian nằm ICU lần đầu lớn chúng cần bị loại bỏ, vì ta không rõ nguyên nhân gây ra hiện tượng đó. Từ hai lập

luận bên trên, chúng tôi đề xuất một giải pháp dung hòa để xử lí vấn đề này như sau: loại bỏ ra khỏi tập dữ liệu những bệnh nhân có thời gian nằm ICU lần đầu lớn hơn tổng thời gian nằm viện với điều kiện khoảng cách giữa 2 khoảng thời gian này không vượt quá 0,5 ngày(điều kiện dung hòa). Khi đó, đối với những trường hợp bệnh nhân được giữ lại nhờ điều kiện dung hòa như trên, chúng tôi sẽ gán tổng thời gian nằm viện chính là thời gian nằm ICU lần đầu.

Bảng 4.7. Các bệnh nhân có thời gian nằm viện nhỏ hơn thời gian nằm ICU lần đầu.

Subject id	Hadm id	ICU stay id	Los ICU	Total los
4331	120294	265287	8.6387	4.5889
5102	119479	214860	34.6725	11.9465
6912	143307	298739	12.5727	7.4236
8697	121838	201587	4.3936	0.2646
9035	103660	238582	26.5245	23.3521
17964	115118	290365	15.6206	12.0764
19617	127959	219554	3.7042	0.5757
27394	174983	281675	10.4183	3.6639
29175	118292	204904	37.2832	2.4306
30405	113826	286420	16.3562	2.4625
31031	106468	259584	6.9243	2.0653
32599	183164	237018	16.509	13.3132
51935	132798	220357	4.7136	0.8681
57518	105024	209071	5.6285	1.1757
84853	112507	295319	3.444	0.2639
94189	134392	221631	4.804	1.2097
95991	173297	207431	7.4876	3.7083

Bước cuối cùng trong phần này, chúng tôi tiến hành loại bỏ những bệnh nhân nằm viện quá 90 ngày. Lí do chúng tôi thực hiện điều này vì số lượng bệnh nhân dạng này rất ít (55 bệnh nhân). Những bệnh nhân này có thể xem là trường hợp ngoại lệ. Hơn nữa, việc chỉ sử dụng chẩn đoán trong thời gian đầu nhập viện để dự đoán với khoảng thời gian quá lớn (trên 90 ngày) là không thực tế và cũng không khả thi.

4.5 Trích xuất đặc trưng

Trong phần này, chúng tôi trình bày các đặc trưng được sử dụng để dự đoán số ngày nằm viện của bệnh nhân. Chúng tôi dựa trên các nghiên cứu trước đó cho bài toán dự đoán số ngày nằm viện, đặc biệt là các nghiên cứu trên bộ dữ liệu MIMIC-III, các tác giả thường đề xuất sử dụng hai nhóm đặc trưng sau đây:

- Nhóm đặc trưng các chỉ số xét nghiệm của bệnh nhân.
- Nhóm đặc trưng các thông tin cơ bản của bệnh nhân.

Chúng tôi gọi nhóm đặc trưng dữ liệu này là **baseline-feature**. Trong luận văn này, chúng tôi sẽ sử dụng lại hai nhóm đặc trưng này làm nền tảng cho các so sánh với cải tiến của chúng tôi. Ở đây, có một lưu ý là các bộ dữ liệu y tế khác nhau thì các thông tin chi tiết thường cũng rất khác nhau, nên tuy rằng hai nhóm đặc trưng được sử dụng là giống với các nghiên cứu trước đó, nhưng về chi tiết từng đặc trưng riêng trong mỗi nhóm có thể không giống nhau hoàn toàn.

Đầu tiên, chúng tôi trích xuất các chỉ số xét nghiệm của bệnh nhân. Sau khi trải qua các bước tiền xử lí, chúng tôi thu được 104 đặc trưng, là các xét nghiệm của bệnh nhân. Để điều kiện bài toán được tổng quát, chúng tôi quyết định chỉ lấy các xét nghiệm thu được trong 24 giờ đầu tiên sau khi bệnh nhân tiếp nhận điều trị ICU. Lưu ý rằng với mỗi bệnh nhân, mỗi lần được tiếp nhận điều trị ICU, họ có thể được xét nghiệm nhiều lần với cùng một loại xét nghiệm. Cũng có những chỉ số xét nghiệm mà bệnh nhân chỉ được đo một lần, hoặc thậm chí không được đo lần nào. Điều này là hợp lí vì bác sĩ chỉ cần thông tin các chỉ số xét nghiệm mà có thể giúp họ chẩn đoán bệnh mà họ nghi ngờ bệnh nhân mắc phải. Ngoài ra, khi một chỉ số xét nghiệm được đo nhiều lần trong ngày, thì xu hướng của các chỉ sỗ xét nghiệm có thể quyết định tình hình bệnh của bệnh nhân, do đó chúng tôi quyết định lấy mỗi chỉ số xét nghiệm hai lần, lần đo đầu tiên trong ngày và lần đo cuối cùng trong ngày. Nếu hai lần đo ra kết quả như nhau, hoặc xét nghiệm đó chỉ được đo một lần, thì chúng tôi xem như là chỉ số xét nghiệm không thay đổi. Như vậy, với cách trích xuất như trên, chúng tôi thu được $104 * 2 = 208$ đặc trưng về xét nghiệm. Chi tiết các đặc trưng xét nghiệm có trong phần phụ lục.

Trong phần tiếp theo, chúng tôi sẽ trình bày các đặc trưng liên quan đến thông tin bệnh nhân:

- **age**: Tuổi của bệnh nhân,
- **gender**: Giới tính của bệnh nhân. Có hai loại giới tính, nên chúng tôi sẽ dùng kĩ thuật one-hot encoding để mã hóa thành 2 đặc trưng là male và female,
- **insurance**: Có tất cả 5 loại bảo hiểm trong bộ dữ liệu. Chúng tôi mã hóa dùng one-hot encoding để thu được 5 đặc trưng dữ liệu. Chi tiết các loại bảo hiểm được trình bày trong phần phụ lục,
- **icu type**: Có tất cả 5 hình thức điều trị ICU trong bộ dữ liệu này. Kĩ thuật one-hot encoding được dùng để trích xuất được 5 đặc trưng dữ liệu.

Như vậy sau bước trích xuất đặc trưng từ các thông tin của bệnh nhân, chúng ta có thêm được 13 đặc trưng dữ liệu.

Cuối cùng, chúng tôi đề xuất thêm một nhóm đặc trưng mới dựa trên chẩn đoán bệnh của bác sĩ. Đầu tiên ta quan sát Bảng 4.8 dưới về các chẩn đoán bệnh của bệnh nhân. Cột cuối cùng trong bảng này là mã bệnh của bệnh mà bác sĩ chẩn đoán cho bệnh nhân. Mỗi bệnh nhân sẽ được chẩn đoán nhiều bệnh khác nhau. Hệ thống mã các loại bệnh được ghi theo chuẩn ICD-9 (International Classification of Diseases), đây là chuẩn quốc tế để mã hóa các chẩn đoán bệnh của bác sĩ. Phiên bản ICD được sử dụng trong bộ dữ liệu MIMIC-III là phiên bản 9. Hiện nay, nhiều bệnh viện đã tiến đến sử dụng các phiên bản ICD-10 và ICD-11. hệ thống ICD-9 mã hóa

mỗi loại bệnh bằng một con số, nhiều mã bệnh sau đó sẽ được gộp thành một nhóm các bệnh có liên quan đến nhau. Cụ thể mã các nhóm bệnh trong hệ thống ICD-9 được mô tả bên dưới:

- **Infectious (mã 001-139)**: Đây là các mã bệnh có liên quan đến bệnh truyền nhiễm và ký sinh trùng,
- **Neoplasms (mã 140-239)**: Đây là mã bệnh có liên quan đến bệnh ung thư,
- **Endocrine (mã 240-279)**: Đây là mã các bệnh liên quan đến nội tiết, dinh dưỡng, chuyển hóa và rối loạn miễn dịch,
- **Blood (mã 280-289)**: Đây là mã các bệnh về máu,
- **Mental (mã 290-319)**: Đây là mã các bệnh liên quan đến chứng rối loạn tâm thần,
- **Nervous (mã 320-389)**: Đây là mã các bệnh liên quan đến hệ thần kinh và các giác quan,
- **Circulatory (mã 390-459)**: Đây là các mã bệnh liên quan đến hệ tuần hoàn,
- **Respiratory (mã 460-519)**: Đây là mã các bệnh liên quan đến hệ hô hấp,
- **Digestive (mã 520-579)**: Đây là mã các bệnh có liên quan đến hệ tiêu hóa,
- **Genitourinary (mã 580-629)**: Đây là mã các bệnh có liên quan đến hệ sinh dục,
- **Pregnancy (mã 630-679)**: Đây là mã các bệnh có liên quan đến các biến chứng trong thời gian mang thai, sinh nở và hậu sản,
- **Skin (mã 680-709)**: Đây là mã các bệnh có liên quan đến da,
- **Muscular (mã 710-739)**: Đây là mã các bệnh có liên quan đến hệ thống cơ xương,
- **Congenital (mã 740-759)**: Đây là mã các bệnh có liên quan đến các dị tật bẩm sinh,
- **Prenatal (mã 760-779)**: Đây là mã các bệnh có liên quan đến thời kỳ chu sinh,
- **Misc (mã 780-799)**: Đây là mã các bệnh liên quan đến các triệu chứng, dấu hiệu và tình trạng chưa xác định rõ.
- **Injury (mã 800-999)**: Đây là mã các bệnh liên quan đến các chấn thương và bệnh ngộ độc,
- **Misc (danh sách ICD-9 mã E và mã V)**: Đây là danh sách các bệnh liên quan đên các tác động bên ngoài gây thương tích và một số phân loại bổ sung.

Quay trở lại nội dung bảng Bảng 4.8, mỗi mã chẩn đoán trong cột **Code ICD-9** có 3 chữ số đầu tạo thành một loại bệnh trong danh sách hệ thống bệnh ICD-9 ở trên, các chữ số còn lại là sự phân loại bệnh vào các nhóm bệnh nhỏ hơn. Chúng ta hoàn toàn có thể sử dụng mỗi mã bệnh nhân như là một đặc trưng dữ liệu, nhận giá trị 0 nếu bệnh nhân được chẩn đoán mắc bệnh và

Bảng 4.8. Thông tin chẩn đoán bệnh của bác sĩ.

Row id	Subject id	Hadm id	Seq num	Code ICD-9
1297	109	172335	1.0	40301
1298	109	172335	2.0	486
1299	109	172335	3.0	58281
1300	109	172335	4.0	5855
1301	109	172335	5.0	4254
1302	109	172335	6.0	2762
1303	109	172335	7.0	7100
1304	109	172335	8.0	2767
1305	109	172335	9.0	7243
1306	109	172335	10.0	45829
1307	109	172335	11.0	2875
1308	109	172335	12.0	28521
1309	109	172335	13.0	28529
1310	109	172335	14.0	27541

nhận giá tị 1 nếu bệnh nhân không được chẩn đoán mắc bệnh. Tuy nhiên, cách lấy dặc trưng như vậy sẽ làm bùng nổ số đặc trưng vì số mã bệnh chính theo hệ thống ICD-9 theo danh sách ở trên đã đến hơn 1000 bệnh, chưa kể trong mỗi loại bệnh lại có nhiều bệnh từ các nhóm nhỏ hơn để mô tả bệnh được chính xác hơn như cách Bảng 4.8 mô tả bệnh. Từ các phân tích trên, chúng tôi đề xuất cách tạo một nhóm đặc trưng liên quan đến chẩn doán, được gọi là **group diagnoses**. Cụ thể, chúng tôi làm như sau:

1. Đầu tiên, với mỗi mã bệnh trong cột **Code ICD-9** trong Bảng 4.8, chúng tôi chỉ giữ lại 3 chữ số đầu tiên, lí do cho việc làm này của chúng tôi là các nhóm nhỏ của cùng một bệnh có khả năng tương đồng vè mặt y tế cao.

2. Để thu hẹp miền giá trị các chẩn đoán hơn nữa, chúng tôi dựa vào danh sách 18 nhóm các mã bệnh ở trên. Với mỗi mã bệnh sau khi thực hiện mã hóa ở bước số một sẽ được mã hóa tiếp một lần nữa bằng cách gán nhãn nhóm bệnh tương ứng với mỗi mã bệnh.

3. Cuối cùng, vì mỗi bệnh nhân sẽ được bác sĩ chẩn đoán có nhiều bệnh khác nhau, nên chúng tôi sẽ cộng dồn các giá trị có cùng mã sau khi thực hiện bước số hai. Các giá trị này chính là giá trị của các đặc trưng mới.

Với cách làm này, số đặc trưng thu được từ các chẩn đoạn bệnh của bệnh nhân được rút gọn xuống chỉ còn 18, tương ứng với số nhóm bệnh trong hệ thống ICD-9.

Để minh họa cho quá trình tạo ra đặc trưng **group diagnoses** ở trên, chúng tôi đưa ra một ví dụ trên một bệnh nhân cụ thể. Xét bệnh nhân có mã bệnh nhân là 109 và mã nhập viện là 172335 như bảng 4.8, sau khi thực hiện bước thứ nhất trong quy trình trích xuất nhóm đặc trưng **group diagnoses**, sẽ thu được thông tin như bảng 4.9, trong đó cột **recode** lã mã bệnh được mã hóa từ cột **icd9 code**.

Bảng 4.9. Thông tin chẩn đoán bệnh sau khi mã hóa bước một.

Row id	Subject id	Hadm id	Seq num	Code ICD-9	Recode
1297	109	172335	1.0	40301	403
1298	109	172335	2.0	486	486
1299	109	172335	3.0	58281	582
1300	109	172335	4.0	5855	585
1301	109	172335	5.0	4254	425
1302	109	172335	6.0	2762	276
1303	109	172335	7.0	7100	710
1304	109	172335	8.0	2767	276
1305	109	172335	9.0	7243	724
1306	109	172335	10.0	45829	458
1307	109	172335	11.0	2875	287
1308	109	172335	12.0	28521	285
1309	109	172335	13.0	28529	285
1310	109	172335	14.0	27541	275

Tiếp theo, sau khi thực hiện bước thứ hai, ta thu được thông tin như Bảng 4.10, trong đó cột **Category** là tên nhóm bệnh tương ứng với mã bệnh tương ứng trong cột **Recode** của Bảng 4.9.

Bảng 4.10. Thông tin chẩn đoán bệnh sau khi mã hóa bước hai.

Row id	Subject id	Hadm id	Seq num	Code ICD-9	Recode	Category
1297	109	172335	1.0	40301	6	circulatory
1298	109	172335	2.0	486	7	respiratory
1299	109	172335	3.0	58281	9	genitourinary
1300	109	172335	4.0	5855	9	genitourinary
1301	109	172335	5.0	4254	6	circulatory
1302	109	172335	6.0	2762	2	endocrine
1303	109	172335	7.0	7100	11	skin
1304	109	172335	8.0	2767	2	endocrine
1305	109	172335	9.0	7243	12	muscular
1306	109	172335	10.0	45829	6	circulatory
1307	109	172335	11.0	2875	3	blood
1308	109	172335	12.0	28521	3	blood
1309	109	172335	13.0	28529	3	blood
1310	109	172335	14.0	27541	2	endocrine

Cuối cùng, chúng tôi thực hiện bước thứ ba. Ví dụ bệnh nhân có mã bệnh 109 và mã nhập viện 172335 có 3 chẩn đoán bị bệnh liên quan đến máu, thì giá trị của đặc trưng dữ liệu **blood** sẽ là giá trị 3. Với cách làm đó, các đặc trưng bệnh của bệnh nhân này lần lượt có giá trịnhư sau: **circulatory** có giá trị 3, **respiratory** có gía trị 1, **genitourinary** có giá trị 2, **endocrine** có giá trị 3, **skin** có giá trị 1, **muscular** nhận giá trị 1. Các đặc trưng của các nhóm bệnh mà bệnh nhân không được chẩn đoán thì sẽ nhận giá trị 0.

4.6 Chuẩn bị dữ liệu cho quá trình học có giám sát

4.6.1 Tạo nhãn cho dữ liệu

Mục tiêu của bài toán của chúng tôi là dự đoán số ngày nằm viện của bệnh nhân. Chúng tôi tính toán số ngày nằm viện của bệnh nhân bằng cách lấy thời gian xuất viện của bệnh nhân trừ đi thời gian nhập viện. Lưu ý rằng con số này phải được tính toán sau khi thực hiện các bước tiền xử lí các vấn đề sai số trong dữ liệu.

4.6.2 Chuẩn bị bảng dữ liệu hoàn chỉnh cho việc huấn luyện mô hình học máy

Sau khi trích xuất các nhóm đặc trưng, chúng tôi thực hiện gom toàn bộ đặc trưng lại tạo thành một bản ghi mới. Bản ghi này sẽ là dữ liệu cho việc dự đoán số ngày nằm viện. Bản ghi này sẽ bao gôm 30,436 điểm dữ liệu và 238 đặc trưng dữ liệu.

4.6.3 Chuẩn hóa dữ liệu

Trước khi thực hiện huấn luyện mô hình, chúng tôi nhận thấy miền giá trị của các đặc trưng có mức chênh lệch rất lớn. Điều này xảy ra chủ yếu vì các chỉ số xét nghiệm được đo có đơn vị khác nhau, do đó có khoảng giá trị khác nhau. Hơn nữa, trong cùng một loại xét nghiệm, chỉ số cũng có sự biến động vì mức độ bệnh của mỗi bệnh nhân là rất khác nhau. Vì vậy, chúng tôi thực hiện chuẩn hóa dữ liệu bằng phương pháp MinMaxScaling để đưa giá trị của các đặc trưng về khoảng từ 0 đến 1. Có một điểm chú ý ở đây là chúng tôi sẽ thực hiện việc chuẩn hóa dữ liệu sau khi đã chia dữ liệu thành tập huấn luyện và tập kiểm tra, chứ không thực hiện trên toàn bộ dữ liệu ban đầu.

$$X_{scaled} = \frac{X - X_{min}}{X_{max} - X_{min}},$$

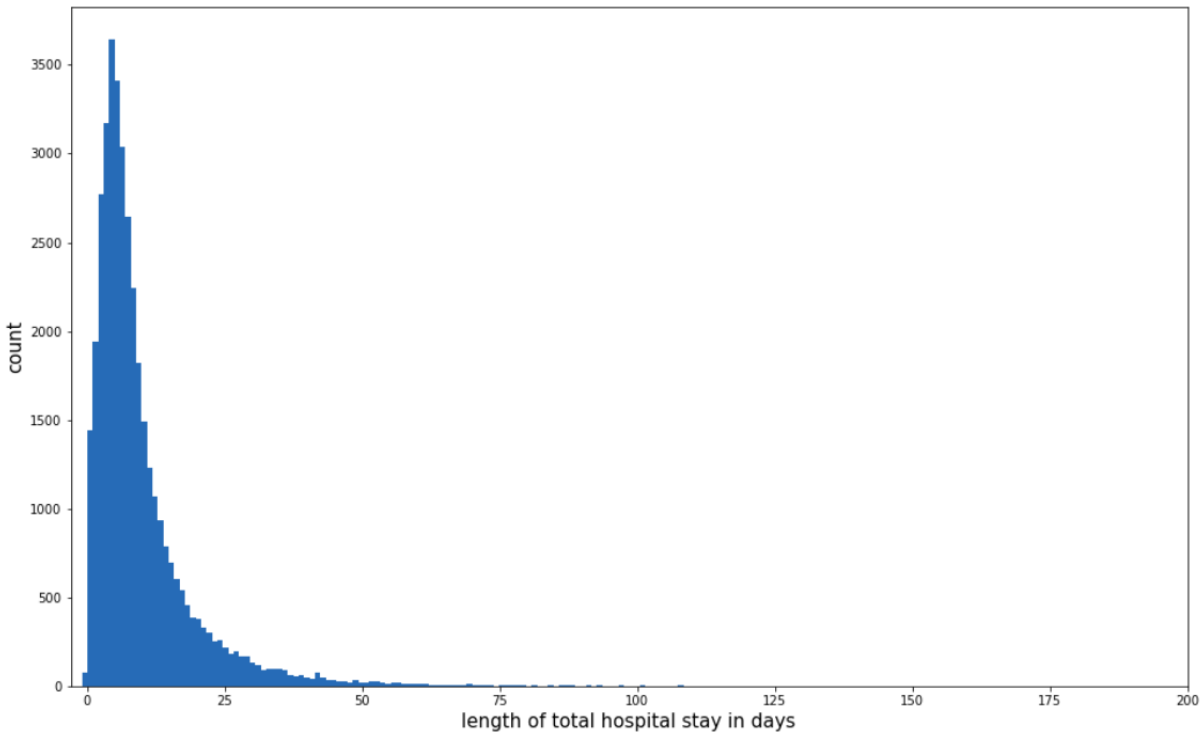
trong đó

- X_{max} là giá trị lớn nhất của X (giá trị lớn nhất trong các giá trị của đặc trưng đang thực hiện chuẩn hóa),
- X_{min} là giá trị nhỏ nhất của X (giá trị nhỏ nhất trong các giá trị của đặc trưng đang thực hiện chuẩn hóa),
- X là giá trị trước khi chuẩn hóa.
- X_{scaled} là giá trị sau khi chuẩn hóa.

4.7 Thăm dò dữ liệu

Trong phần này, chúng tôi tiến hành thăm dò một số đặc điểm của dữ liệu để có thể hiểu bảo chất dữ liệu.

Đầu tiên chúng tôi quan sát được phân bố thời gian nằm viện của bệnh nhân không đều, đa số bệnh nhân chỉ nằm viện ngắn ngày, điều này được thể hiện qua Hình 4.2 bên dưới.



Hình 4.2. Đồ thị mô tả phân bố số ngày nằm viện của bệnh nhân.

Cụ thể, chúng tôi tính toán thời gian trung bình nằm viện của bệnh nhân là 10.82 ngày. Trong tổng số 30436 bệnh nhân trong bộ dữ liệu, chỉ có 1492 bệnh nhân có thời gian nằm viện trên 30 ngày, chỉ bằng khoảng 5% số bệnh nhân nằm viện dưới 30 ngày. Đây là một vấn đề nghiêm trọng đối với các mô hình hồi quy, vì phần dữ liệu các bệnh nhân nằm viện dưới 30 ngày rất nhiều có thể làm mô hình học máy dự đoán tốt trên phần dữ liệu bệnh nhân nằm viện dưới 30 ngày, nhưng sẽ dự đoán không tốt đối với những bệnh nhân nằm viện trên 30 ngày.

Tiếp theo chúng tôi tiến hành quan sát về các loại bảo hiểm mà các bệnh nhân sử dụng. Chúng tôi nhận thấy những bệnh nhân tự chi trả viện phí có số ngày nằm viện thấp hơn hẳn những bệnh nhân có sử dụng bảo hiểm. Điều này rất hợp lí vì chúng ta đều biết chi phí y tế ở Mỹ (nơi bộ dữ liệu MIMIC-III được thu thập) vô cùng đắt đỏ nếu không có bảo hiểm. Các thông tin này được thể hiện qua Bảng 4.7 bên dưới.

Bảng 4.11. Thông tin bảo hiểm của bệnh nhân và số ngày nằm viện trung bình của các bệnh nhân mỗi loại bảo hiểm.

	Số bệnh nhân	Thời gian nằm viện trung bình (đơn vị: ngày)
Government insurance	906	10.6
Medicaid insurance	2443	12.46
Medicare insurance	16448	10.73
Private insurance	10282	10.68
Self Pay insurance	357	8.8

4.8 Mô hình học máy cho quá trình dự đoán số ngày nằm viện

4.8.1 Các mô hình kinh điển

Chúng tôi sử dụng bốn mô hình học máy để huấn luyện bộ dữ liệu và so sánh kết quả. Các mô hình đều là mô hình hồi quy, bao gồm: Gradient Boosting, Support Vector Regression, Random Forest. Trước khi thực hiện việc huấn luyện, chúng tôi chia tập dữ liệu thành hai tập là tập huấn luyện và tập kiểm tra. Tập huấn luyện gồm 80% dữ liệu. Tập kiểm tra gồm 20% dữ liệu còn lại, Trong quá trình thực hiện việc huấn luyện mô hình với tập huấn luyện, chúng tôi sử dụng kỹ thuật k-fold cross-validation để có thể đánh giá mức độ tổng quát hóa của mô hình đối với dữ liệu một cách tốt hơn.

4.8.2 Mô hình hai giai đoạn

Từ phân tích trong Mục 4.7 về số bệnh nhân nằm viện dưới 30 ngày và trên 30 ngày, chúng tôi đề xuất một mô hình hai giai đoạn để có thể cải thiện tính thực tế của việc dự đoán số ngày nằm viện. Mô hình cụ thể như sau:

1. Chia bộ dữ liệu D ra thành hai tập là tập huấn luyện và tập kiểm thử với tỉ lệ 80/20.
2. Trên tập huấn luyện:
   (a) Huấn luyện một bộ phân lớp bệnh nhân nằm viện trên 30 ngày hay dưới 30 ngày với dữ liệu trên tập huấn luyện.
   (b) Chia tập huấn luyện ra làm hai phần, phần bệnh nhân nằm viện dưới 30 ngày (D_1) và phân bệnh nhân nằm viện trên 30 ngày (D_2).
   (c) Thực hiện xây dựng hai mô hình hồi quy H_1 và H_2 lần lượt dự đoán số ngày nằm viện trên hai tập dữ liệu D_1 và D_2.
3. Trên tập kiểm tra, với mỗi bệnh nhân:

(a) Đưa các thông tin của bệnh nhân vào bộ phân lớp để dự đoán bệnh nhân có khả năng nằm viện trên 30 ngày hay dưới 30 ngày.

(b) Nếu bệnh nhân nằm viện dưới 30 ngày, sử dụng mô hình hồi quy H_1 đã được huấn luyện với tập dữ liệu D_1. Nếu bệnh nhân nằm viện trên 30 ngày, sử dụng mô hình hồi quy H_2 đã được huấn luyện với tập dữ liệu D_2.

4. Đầu ra là ước lượng số ngày nằm viện của bệnh nhân.

Chúng tôi lí giải tại sao xây dựng mô hình hai giai đoạn như trên. Đầu tiên, chúng tôi nhắc lại quan sát ở Mục 4.7 rằng số lượng bệnh nhân nằm viện trên 30 ngày rất ít so với số lượng bệnh nhân nằm viện dưới 30 ngày, do đó mô hình hồi quy có thể bị thiên lệch về phía bệnh nhân có thời gian nằm viện dưới 30 ngày, đặc biệt là các bệnh nhân nằm viện ngắn ngày (dưới 2 tuần). Khi sử dụng mô hình hồi quy như vậy trên những bệnh nhân nằm viện dài ngày, thậm chí nhiều bệnh nhân nằm viện trên hai tháng, thì kết quả hồi quy sẽ rất hạn chế. Nhận xét này chính là tiền đề của mô hình hai giai đoạn hai bước của chúng tôi.

Đầu tiên với mỗi bệnh nhân mới, họ sẽ được phân loại nằm viện trên 30 ngày hay dưới 30 ngày. Sau khi biết thông tin này thì chúng tôi sẽ sử dụng một trong hai mô hình hồi quy tương ứng được huấn luyện với tập bệnh nhân nằm viện trên 30 ngày hay trên tập dữ liệu bệnh nhân nằm viện dưới 30 ngày. Chúng tôi kì vọng mô hình hồi quy sẽ làm việc tốt hơn với ý tưởng này. Hơn nữa, trong tình huống xấu nhất là kết quả hồi quy với phương pháp hai giai đoạn này cho kết quả không tốt hơn các mô hình kinh điển, thì vẫn có ít nhất một ưu điểm là mô hình hai giai đoạn này đã phân loại được đúng một số lượng bệnh nhân nằm viện dài ngày. Điều này là rất có ý nghĩa vì đứng về góc nhìn của bệnh viện, họ cũng không lên kế hoạch quá xa cho bệnh nhân như vậy (trên 1 tháng), chỉ cần biết bệnh nhân nằm viện lâu hơn 30 ngày là đủ. Trong phần thí nghiệm, chúng tôi sẽ chứng minh ý tưởng này là khả thi, rằng mô hình hai giai đoạn tuy chỉ cho kết quả gần bằng các mô hình kinh điển, nhưng có ưu điểm là phân loại được một số lượng bệnh nhân nằm viện dài ngày.

Chương 5

Hiện thực hệ thống

5.1 Trích xuất thông tin từ dữ liệu gốc

Như đã trình bày ở mục 4.4.1, chúng tôi sẽ lấy các thông tin cần thiết từ 26 bảng dữ liệu ban đầu. Chúng tôi sử dụng quy trình được đề xuất bởi Wang và các cộng sự [18]. Họ đã hiện thực một package python với tên gọi MIMIC-Extract để giúp việc lấy các thông tin cần thiết được dễ dàng hơn. Ưu điểm đặc biệt trong quy trình đề xuất của MIMIC-Extract là có sự cộng tác với các chuyên gia y khoa, giúp gộp các xét nghiệm có y nghĩa tương tự nhau mà không làm mất đi ý nghĩa về mặt y học.

Để dễ dàng cho việc giải thích, chúng tôi sẽ đưa vào các đoạn mã hiện thực chương trình vào trong phần này.

Đầu tiên, chúng tôi download thư viện MIMIC-Extract từ trang github của project theo đường link[1].

Tiếp theo chúng tôi trình bày đoạn mã để lấy thông tin. Lưu ý rằng bộ dữ liệu MIMIC-III có kích thước rất lớn, việc xử lí dữ liệu được thực hiện trên hệ thống High-performance computing của Khoa Khoa học và Kỹ thuật Máy tính , Trường Đại học Bách Khoa thành phố Hồ Chí Minh. Các đoạn mã sau đều được thực thi thông qua terminal.

[1] MIMIC-Extract:A Data Extraction, Preprocessing, and Representation Pipeline for MIMIC-III

```
#In folder MIMIC-Extract, set current directory to utils/
    cd utils/

#Step 1: Setup env vars for current local system
    source ./setup_user_env.sh

#Step 2: Create conda environment
    conda env create --force -f ../mimic_extract_env_py36.yml
    conda activate mimic_data_extraction

#Step 3: Build Views for Feature Extraction
    cd $MIMIC_CODE_DIR/concepts
    psql -d mimic -f postgres-functions.sql
    bash postgres_make_concepts.sh

#Step 4: Set Cohort Selection
    cd $MIMIC_EXTRACT_CODE_DIR
    cd utils

#Step 5: Extract information from PSQL
    make build_curated_from_psql
```

Sau khi lần lượt chạy những đoạn command line trên, chúng tôi thu được kết quả gồm nhiều file ouput khác nhau, trong đó có 2 file quan trọng nhất là **patients.csv** chứa các thông tin bệnh nhân và **vitals_labs.csv** chứa các thông tin xét nghiệm của bệnh nhân.

5.2 Tiền xử lý dữ liệu

Trong mục này, chúng tôi trình bày mã nguồn cho việc xử lý các vấn đề sai số trong dữ liệu.

Đầu tiên, với những bệnh nhân có thời gian nhập viện sau thời gian tiếp nhận điều trị ICU lần đầu tiên, chúng tôi thay thời gian nhập viện bởi thời gian vào ICU.

```
#Get index of patients having hospital admission after first ICU admission
abnormal_amittime_patients_index = functools.reduce(lambda x, y: x + [y]
if patients.iloc[y,patients.columns.get_loc('admittime')] >
patients.iloc[y,patients.columns.get_loc('intime')]
else x +[], [i for i in range(0,patients.shape[0])] ,[])

#Change hospital admission time to first ICU admission time
for index in abnormal_amittime_patients_index:
    patients.iloc[index,patients.columns.get_loc('admittime')] =
    patients.iloc[index,patients.columns.get_loc('intime')]

#Check again if this abnomal behaviour exist
patients.loc[patients['admittime'] > patients['intime']]
```

Tiếp theo, chúng tôi loại bỏ các bệnh nhân có thời gian nhập viện xảy ra sau thời gian xuất viện.

```
# Abnormal behavior: Some patients have admittime > dischtime
patients.loc[patients['admittime'] > patients['dischtime']]

#list patients with abnormal behavior above
list_abnormal_admit_disch_gap =
list(patients.loc[patients['admittime'] >
patients['dischtime']].index.get_level_values('subject_id'))

#Delete patients with abnormal behavior
patients = patients.drop(list_abnormal_admit_disch_gap)

#delete vitals lab of patients with abnormal behavior above
vitals_data = vitals_data.drop(list_abnormal_admit_disch_gap)

# check again in two table patients and vitals_data
print(patients.loc[patients['admittime'] > patients['dischtime']])
```

Tiếp theo, chúng tôi thực hiện loại bỏ các bệnh nhân có số ngày nằm ICU lần đầu lớn hơn tổng số ngày nằm viện. Tuy nhiên, như đã phân tích trong Tiểu mục 4.4.2, chúng tôi tiến hành giữ lại những bệnh nhân mà số ngày nằm ICU lần đầu lớn hơn tổng số ngày nằm viện không lớn hơn 0,5 ngày.

```
# abnormal behavior: many hospital LOS < ICU LOS
gap_los_icu_hospital = list(patients['total_los'] -
patients['los_icu'])

#delete patients with total LOS < patients LOS ( to keep as much
#data as possible, we delete with condition)
list_los_total_smaller_icu = patients.loc[(patients['total_los']
- patients['los_icu']) <=
-0.5].index.get_level_values('subject_id')

patients = patients.drop(list_los_total_smaller_icu)
vitals_data = vitals_data.drop(list_los_total_smaller_icu)

# if los_icu > total_los => total_los := los icu
abnormal_icu_total_gap_index = functools.reduce(lambda x, y: x +
[y] if patients.iloc[y,patients.columns.get_loc('total_los')] <
patients.iloc[y,patients.columns.get_loc('los_icu')] else x +[],
[i for i in range(0,patients.shape[0])] ,[])

for index in abnormal_icu_total_gap_index:
    patients.iloc[index,patients.columns.get_loc('total_los')] =
    patients.iloc[index,patients.columns.get_loc('los_icu')]

#Check again if the abnomal behaviour exist
patients.loc[patients['total_los'] < patients['los_icu']]
```

Cuối cùng, chúng tôi loại bỏ các bệnh nhân nằm viện quá 90 ngày.

```
drop_long_los = patient_data.index[patient_data['total_los']
>= 30].tolist()

patient_data = patient_data.drop(index=drop_long_los)

patient_data = patient_data.reset_index(drop=True)
```

5.3 Trích xuất đặc trưng

Trong phần này, chúng tôi tiến hành trích xuất các đặc trưng đã đề xuất ở Mục 4.5. Việc trích xuất các đặc trưng trải qua nhiều giai đoạn và nhiều hàm chức năng khác nhau cho các đặc trưng khác nhau. Các đặc trưng sau khi được trích xuất sẽ được đưa vào bảng dữ liệu mới có dạng DataFrame với tên gọi `patient_data`.

Đầu tiên chúng tôi tiến hành trích xuất các đặc trưng liên quan đến xét nghiệm. Chúng tôi sẽ trích xuất dữ liệu xét nghiệm ngày đầu tiên sau khi bệnh nhân tiếp nhận điều trị ICU lần đầu tiên. Dữ liệu xét nghiệm được lấy hai lần, lần đo đầu tiên trong ngày và lần đo cuối cùng trong ngày.

```
# Get list vital_lab_name
vital_lab_name = list(vitals_data.columns.levels[0])
vital_lab_name = [c.replace(' ','_') for c in vital_lab_name]

# double vital_lab_list
vital_name_first_last = []
for i in vital_lab_name:
    vital_name_first_last.append(i)
    vital_name_first_last.append(str(i + '_last'))


# Create new dataframe with intervention name as column
patient_data = pd.DataFrame(columns = vital_name_first_last)

# Get subject id of all patient
patient_id = patients.index.get_level_values('subject_id')

# Add first and last observed measure of each vitals lab to patient_data
idx = pd.IndexSlice

for p_id in patient_id:
    a = vitals_data_filtered.loc[idx[p_id,:,:,:], idx[:,'mean']]
    vital_each_patient = []
    for i in range(0,104):
        if a.iloc[:,i].first_valid_index() == None:
            vital_each_patient.append(np.nan)
        else:
            index = a.iloc[:,i].first_valid_index()[3]
            vital_each_patient.append(a.iloc[index, i])
        if a.iloc[:,i].last_valid_index() == None:
```

```
            vital_each_patient.append(np.nan)
        else:
            index = a.iloc[:,i].last_valid_index()[3]
            vital_each_patient.append(a.iloc[index, i])

    patient_data.loc[p_id] = vital_each_patient
```

Trong DataFrame `patient_data` đến giai đoạn này đã có thông tin của các xét nghiệm của bệnh nhân. Tiếp theo, chúng tôi tiên hành mã hóa one-hot encoding các dặc trưng nhận giá trị rời rạc và đưa nó vào DataFrame `patient_data`. Đồng thời, đặc trưng nhận giá trị liên tục là tuổi cũng được đưa vào DataFrame `patient_data`.

```
#Add static variable to dataset
dummy_sex = pd.get_dummies(patients['gender'])
dummy_insurance = pd.get_dummies(patients['insurance'])
dummy_insurance.columns = [c.replace(' ', '_') for c in dummy_insurance.columns]

#One-hot encoding categorical variable
male = pd.Series(dummy_sex.M)
female = pd.Series(dummy_sex.F)
Government_insurance = pd.Series(dummy_insurance.Government)
Medicaid_insurance = pd.Series(dummy_insurance.Medicaid)
Medicare_insurance = pd.Series(dummy_insurance.Medicare)
Private_insurance = pd.Series(dummy_insurance.Private)
Self_Pay_insurance = pd.Series(dummy_insurance.Self_Pay)

#Add categorical variable to data-set
patient_data['male'] = male.values
patient_data['female'] = female.values
patient_data['Government_insurance'] = Government_insurance.values
patient_data['Medicaid_insurance'] = Medicaid_insurance.values
patient_data['Medicare_insurance'] = Medicare_insurance.values
patient_data['Private_insurance'] = Private_insurance.values
patient_data['Self_Pay_insurance'] = Self_Pay_insurance.values

#Add age of patient to data-set
age_patient = pd.Series(patients['age'])
patient_data['age'] = age_patient.values
```

Bước kế tiếp, chúng tôi tiến hành xây dựng đặc trưng mới (***group diagnoses***) mà chúng tôi đã đề xuất. Các bước để tiến hành trích xuất đặc trưng này đã được trình bày chi tiết trong Mục 4.5. Chúng tôi sẽ trích xuất dặc trưng này từ bảng `D_ICD_DIAGNOSES`,một trong 26 bảng dữ liệu trong bộ dữ liệu MIMIC-III ban đầu.

```
#Filter out E and V codes
icd_code['recode'] = icd_code['ICD9_CODE']
icd_code['recode'] = icd_code['recode'][~icd_code['recode'].str.contains
("[a-zA-Z]").fillna(False)]
icd_code['recode'].fillna(value='999', inplace=True)
icd_code['recode'] = icd_code['recode'].str.slice(start=0, stop=3, step=1)
icd_code['recode'] = icd_code['recode'].astype(int)

# ICD-9 Main Category ranges
icd9_ranges = [(1, 140), (140, 240), (240, 280), (280, 290), (290, 320),
(320, 390),
(390, 460), (460, 520), (520, 580), (580, 630), (630, 680), (680, 710),
(710, 740), (740, 760), (760, 780), (780, 800),
(800, 1000), (1000, 2000)]

# Associated category names
diag_dict = {0: 'infectious', 1: 'neoplasms', 2: 'endocrine', 3: 'blood',
             4: 'mental', 5: 'nervous', 6: 'circulatory', 7: 'respiratory',
             8: 'digestive', 9: 'genitourinary', 10: 'pregnancy', 11: 'skin',
             12: 'muscular', 13: 'congenital', 14: 'prenatal', 15: 'misc',
             16: 'injury', 17: 'misc'}

# Re-code in terms of integer
for num, cat_range in enumerate(icd9_ranges):
    icd_code['recode'] = np.where(icd_code['recode'].between(
    cat_range[0],cat_range[1]),
            num, icd_code['recode'])

# Convert integer to category name using diag_dict
icd_code['recode'] = icd_code['recode']
icd_code['cat'] = icd_code['recode'].replace(diag_dict)

# Create list of diagnoses for each admission
hadm_list = icd_code.groupby('HADM_ID')['cat'].apply(list).reset_index()
```

```
# Convert diagnoses list into hospital admission-item matrix
hadm_item = pd.get_dummies(hadm_list['cat'].apply
(pd.Series).stack()).sum(level=0)

# Add patients with chosen cohort selection to data frame
index_hadm_patients_table = list(patients.index.get_level_values('hadm_id'))
hadm_item = hadm_item.loc[index_hadm_patients_table]
patient_data = patient_data.reset_index(drop=True).join
(hadm_item.reset_index(drop=True))
```

5.3.1 Tạo nhãn cho dữ liệu

Chúng tôi tính toán số ngày nằm viện của bệnh nhân bằng cách lấy thời gian xuất viện trừ đi thời gian nhập viện. Dữ liệu để tính toán trong phần này được lấy từ bảng dữ liệu từ file `patients.csv`. Dữ liệu gốc được ghi theo đơn vị giờ, chúng tôi chuyển thành đơn vị ngày.

```
# Calculating total LOS
total_los_series = pd.Series(list(map(lambda x:
round(x.total_seconds()/(24*60*60),4),(patients.iloc[:,patients
.columns.get_loc('dischtime')] -
patients.iloc[:,patients.columns.get_loc('admittime')]))))

# Add to data frame patient_data
patient_data['total_los'] = total_los_series.values
```

5.4 Huấn luyện mô hình học máy

5.4.1 Các mô hình kinh điển

Đầu tiên, chúng tôi chia tập dữ liệu gồm 30436 bệnh nhân theo tỉ lệ 80/20 thành tập dữ liệu huấn luyện gồm 24348 bệnh nhân và tập dữ liệu kiểm thử gồm 6088 bệnh nhân.

```
# Get LOS from patient_data
total_los_patient = patient_data['total_los']

# Train test split
X_train, X_test, y_train, y_test =
train_test_split(patient_data, total_los_patient,
```

```
test_size=0.2, random_state=1)

#Drop LOS column from patient_data
X_train = X_train.drop(['total_los'], axis=1)
X_test = X_test.drop(['total_los'], axis=1)
```

Để huấn luyện mô hình học máy, chúng tôi sử dụng các mô hình hồi quy Gradient Boosting, Support Vector Machine, Random Forest, Linear Regression. Đối với mô hình Gradient Boosting, chúng tôi sử dụng hiện thực trong package XGboost. Chúng tôi sử dụng lớp `XGBRegressor` được hiện thực trong package XGboost như sau:

```
xgb_model = xgboost.XGBRegressor(gamma=0.4, learning_rate=0.03,
max_depth=6, n_estimators=200)
```

Các tham số khởi tạo của `XGBRegressor`được sử dụng là:

- `n_estimators` : số lượng cây được dùng cho mô hình.
- `gamma` : loss reduction tối thiểu để tạo thêm phân vùng cho một nút lá.
- `learning_rate` : tốc độ học cho quá trình tăng tốc (*boosting learning rate*).
- `max_depth` : độ sâu tối đa của cây.

Đối với mô hình Support Vector Machine, chúng tôi sử dụng lớp `SVR`trong hiện thực của sklearn:

```
svm_model = SVR(kernel='poly', random_state=1,
max_iter=2000, verbose=True)
```

Các tham số của `SVR` được chúng tôi sử dụng gồm:

- `kernel`: loại kernel được sử dụng.
- `random_state`: dùng để sinh sô ngẫu nhiên.
- `max_iter`: số bước lặp tối đa.
- `verbose`:in ra các thông báo trong quá trình huấn luyện mô hình.

Cuối cùng chúng tôi sử dụng lớp `RandomForestRegressor`được hiện thực trong thư viện sklearn:

```
rf_model = RandomForestRegressor(max_depth=14,random_state=1,
n_jobs=40, max_features=10)
```

Các tham số được chúng tôi sử dụng là:

- `max_depth`: độ sâu tối đa của cây.
- `random_state`: dùng để sinh số ngẫu nhiên.
- `n_jobs`: dùng để song song hóa quá trình chạy mô hình.
- `max_features`: số lượng đặc trưng cần thiết khi xem xét tỉa cây.

Các mô hình được sử dụng ở trên ngoài các tham số được trình bày, có có nhiều tham số khác. Đối với các tham số mà không được đề cập, chúng tôi sử dụng bộ tham số mặc định được hiện thực trong các thư viện.

Để đánh giá mức độ tổng quát hóa của bộ tham số được sử dụng, chúng tôi hiện thực một quy trình đánh giá chéo (k-fold cross-validation). Lí do chúng tôi hiện thực riêng quy trình này mà không sử dụng quy trình được đề xuất của thư viện sklearn là nhằm có thể tùy biến các yêu cầu riêng của chúng tôi như là: chuẩn hóa dữ liệu riêng cho mỗi vòng lặp trong quá trình kiểm tra chéo, sử dụng và đánh giá hiệu quả của mô hình hai giai đoạn sử dụng kiểm tra chéo (được trình bày trong Tiểu mục 4.8.2.

Đầu tiên, đối với các mô hình hồi quy kinh điển, chúng tôi hiện thực quy trình sau (đối với mô hình hai giai đoạn, quy trình đánh giá chéo sẽ được trình bày trong Tiểu mục 5.4.2 bên dưới).

```
kf = KFold(n_splits=10)
validation_score = []

for train_index, test_index in kf.split(X_train):
    #data preparation
    X_training = X_train[list(train_index)]
    y_training = y_train[list(train_index)]
    X_validating = X_train[list(test_index)]
    y_validating = y_train[list(test_index)]

    y_training = np.array(y_training)
    y_validating = np.array(y_validating)

    #scaling data
    mm_scaler_loop = MinMaxScaler()
    mm_scaler_loop.fit(X_training)
```

```
    X_training = mm_scaler_loop.transform(X_training)
    X_validating = mm_scaler_loop.transform(X_validating)

    #impute nan value with constant 0
    missingvalues = SimpleImputer(missing_values = np.nan,
    strategy = "constant", fill_value=0)
    missingvalues = missingvalues.fit(X_training)
    X_training = missingvalues.transform(X_training)

    missingvalues2 = SimpleImputer(missing_values = np.nan,
    strategy = "constant", fill_value=0)
    missingvalues2 = missingvalues.fit(X_validating)
    X_validating = missingvalues.transform(X_validating)

    #training and validating model
    svm_model.fit(X_training, y_training)

    validating_loss = mean_absolute_error(y_validating,

    svm_model.predict(X_validating))
    validation_score.append(validating_loss)
```

Ở trên đây, chúng tôi chỉ trình bày phần hiện thực cho mô hình hồi quy Support Vector machine, các mô hình còn lại hoàn toàn tương tự.

5.4.2 Mô hình hai giai đoạn

Trong phần này, chúng tôi trình bày phần mã hiện thực quy trình kiểm tra chéo được đề xuất trong Tiểu mục 4.8.2.

Đầu tiên, chúng tôi gán nhãn cho các bệnh nhân trên hai tập, tập huấn luyện và tập kiểm thử. Bệnh nhân sẽ có nhãn là 1 nếu nằm viện dưới 30 ngày, và có nhãn là 2 nếu nằm viện trên 30 ngày.

```
# Create label for train set
train_label = []
for los in y_train:
    if los < 30:
        train_label.append(1)
    else:
        train_label.append(2)
# Create label for test set
```

```
test_label = []
for los in y_test:
    if los < 30:
        test_label.append(1)
    else:
        test_label.append(2)
```

Tiếp theo, chúng tôi trình bày phần mã hiện thực mô hình hai giai đoạn với quy trình kiểm tra chéo do chúng tôi hiện thực.

```
kf = KFold(n_splits=10)
validation_score = []

for train_index, test_index in kf.split(X_train):
# Data preparation
    X_training = X_train.loc[list(train_index)]
    y_training = y_train[list(train_index)]
    X_validating = X_train.loc[list(test_index)]
    y_validating = y_train[list(test_index)]

# Get min-max scale on training data
    X_training_for_scale = X_training
    mm_scaler_loop = MinMaxScaler()
    mm_scaler_loop.fit(X_training_for_scale)

# Get label for classification step
    train_label_loop = []
    for los in y_training:
        if los < 30:
            train_label_loop.append(1)
        else:
            train_label_loop.append(2)

    validating_label_loop = []
    for los in y_validating:
        if los < 30:
            validating_label_loop.append(1)
        else:
            validating_label_loop.append(2)

# Prepare data for regression step
    X_train_1 = X_training[y_training < 30]
    X_train_2 = X_training[y_training >= 30]
    X_train_1 = mm_scaler_loop.transform(X_train_1)
    X_train_2 = mm_scaler_loop.transform(X_train_2)

    missingvalues1 = SimpleImputer(missing_values = np.nan,
    strategy = "constant", fill_value=0)
    missingvalues1 = missingvalues1.fit(X_train_1)
```

```
    X_train_1 = missingvalues1.transform(X_train_1)

    missingvalues2 = SimpleImputer(missing_values = np.nan,
    strategy = "constant", fill_value=0)
    missingvalues2 = missingvalues2.fit(X_train_2)
    X_train_2 = missingvalues2.transform(X_train_2)

    total_los_1 = y_training[y_training<30]
    total_los_2 = y_training[y_training>=30]

    total_los_1 = np.array(total_los_1)
    total_los_2 = np.array(total_los_2)

# Prepare data for classification step
    X_training = mm_scaler_loop.transform(X_training)
    X_validating = mm_scaler_loop.transform(X_validating)

    y_training = np.array(y_training)
    y_validating = np.array(y_validating)

    missingvalues3 = SimpleImputer(missing_values = np.nan,
    strategy = "constant", fill_value=0)
    missingvalues3 = missingvalues3.fit(X_training)
    X_training = missingvalues3.transform(X_training)

    missingvalues4 = SimpleImputer(missing_values = np.nan,
    strategy = "constant", fill_value=0)
    missingvalues4 = missingvalues4.fit(X_validating)
    X_validating = missingvalues4.transform(X_validating)

# Training two regression model on two subset data
    svr1.fit(X_train_1, total_los_1)
    svr2.fit(X_train_2, total_los_2)

# Training classifier
    xgb_classifier_model.fit(X_training,
    np.array(train_label_loop) ,eval_set=[(X_validating,
    validating_label_loop)],early_stopping_rounds=200,
    verbose=5)
```

```
# Validating hybrid model
    y_predict_validate = []
    for i in range(0, X_validating.shape[0]):
        temp_y_validate = y_validating[i].reshape(-1,1)
        temp_X_validate = X_validating[i].reshape(-1,X_validating.shape[1])
        class_patient = svc.predict(temp_X_validate)

        if class_patient[0] == 1:
            y_predicted = svr1.predict(temp_X_validate)
        else:
            y_predicted = svr2.predict(temp_X_validate)

        y_predict_validate.append(y_predicted[0])

    validating_loss = mean_absolute_error(list(y_validating),
    y_predict_validate)
    validation_score.append(validating_loss)
```

Trong đoạn mã trên, ở mỗi vòng lặp trong quy trình kiểm tra chéo, dữ liệu sẽ được xử lí trong vòng lặp này, bao gồm các bước tiền xử lí như chuẩn hóa dữ liệu, điền dữ liệu trống sử dụng hằng số. Các bước xây dựng mô hình phân lớp và và mô hình hồi quy trong mô hình hai giai đoạn cũng sẽ được đánh giá trong quy trình khép kín với đoạn mã trên.

Chương 6

Thí nghiệm và đánh giá

6.1 Các độ đo được sử dụng

Trong chương này, chúng tôi trình bày các độ đo được sử dụng để đánh giá hiệu năng mô hình. Đầu tiên, chúng tôi trình bày ma trận nhầm lẫn (*confusion matrix*) cho các nhãn dự đoán (ma trận nhầm lẫn này sẽ được dùng như một công cụ đánh giá mô hình hai giai đoạn được đề xuất).

Bảng 6.1. Ma trận nhầm lẫn cho các nhãn dữ liệu với bước phân lớp.

		Nhãn thực tế	
		Ngắn ngày	Dài ngày
Nhãn dự đoán	Ngắn ngày	True Positive (TP)	False Positive (FP)
	Dài ngày	False Negative (FN)	True Negative (TN)

Tiếp theo chúng tôi trình bày độ đo **Mean absolute error (MAE)**. Đây là độ đo được dùng để đánh giá hiệu năng các mô hình hồi quy được sử dụng. Nó có ý nghĩa là trung bình sai khác của số ngày thực tế nằm viện của bệnh nhân và số ngày bệnh nhân được dư đoán sẽ nằm viện theo mô hình hồi quy

$$MAE = \frac{1}{N}\sum_{i=1}^{N} \left| y_{i_{predicted}} - y_{i_{real}} \right|$$

Trong đó:

- $\mathbf{y_{i_{predicted}}}$: là số ngày nằm viện của bệnh nhân được dự đoán dựa trên mô hình hồi quy,
- $\mathbf{y_{i_{real}}}$: là số ngày nằm viện thực tế của bệnh nhân.

Cuối cùng, chúng tôi dùng độ đo **Mean validation error (MVE)** để đánh giá mức độ tổng quát hóa của mô hình. Độ đo này được tính bằng cách lấy trung bình độ lỗi **MAE** khi thực hiện đánh giá chéo k-fold cross-validation

$$MVE = \frac{1}{K}\sum_{i=1}^{K} MAE_i$$

6.2 Thiết kế thí nghiệm

Trong mục này, chúng tôi trình bày các bước thực hiện thí nghiệm. Chúng tôi tiến hành thí nghiệm theo các bước như sau:

1. Chia tập dữ liệu thành tập huấn luyện và tập kiểm thử với tỉ lệ 80/20. Tiến hành đánh nhãn cho tập dữ liệu. Việc đánh nhãn này nhằm để sử dụng cho bước phân lớp trong mô hình hai giai đoạn.

2. Thực hiện 10-fold cross-validation trên tập huấn luyện. Chúng tôi sẽ thực hiện quá trình này với nhiều bộ tham số khác nhau để lựa chọn bộ tham số tối ưu với tập dữ liệu của chúng tôi. Bộ tham số tối ưu là bộ tham số cho kết quả **Mean validation error (MVE)** nhỏ nhất.

Bảng 6.2. Các tham số để thực hiện tìm kiếm.

(a) Mô hình Gradient boosting.

Tham số	Tập giá trị
learning_rate	{0.02, 0.03, 0.05, 0.3, 0.7}
max_depth	{3, 5, 6, 9, 10, 12, 14}
gamma	{0.1, 0.2, 0.3, 0.4, 0.5}
n_estimators	{200, 400, 600, 800, 1000}

(b) Mô hình Support vector Regression.

Tham số	Tập giá trị
max_iter	{500, 1000, 1500, 2000}
gamma	{1, 0.1, 0.001, 0.0001}
kernel	{'linear','poly', 'rbf'}

(c) Mô hình Random Forest.

Tham số	Tập giá trị
max_depth	{3, 5, 6, 9, 10, 12, 14}
max_features	{10, 20, 40, 50, 70}

3. Thực hiện kiểm tra hiệu quả của đặc trưng được đề xuất. Đầu tiên, chúng tôi tiến hành chạy mô hình hồi quy kinh điển trên các đặc trưng cơ bản (**baseline-feature**) đã được sử dụng trong các nghiên cứu trước đây. Sau đó, chúng tôi tiến hành chạy các mô hình trên bộ đặc trưng **baseline- feature** cộng với nhóm đặc trưng về chẩn đoán bệnh, gọi là **group-diagnoses** do chúng tôi đề xuất. Bộ đặc trưng sau khi gộp toàn bộ các đặc trưng cơ bản **baseline-feature** và nhóm đặc trưng về chẩn đoán bệnh được gọi là **total-features-set**.

4. Chúng tôi tiến hành thu hẹp các điều kiện thí nghiệm để xem mức độ kết quả được cải thiện. Nhắc lại rằng trong nghiên cứu này chúng tôi thực hiện trên điều kiện rất tổng quát là dự đoán số ngày nằm viện của bệnh nhân nằm viện tối đa đến 90 ngày. Trên thực tế, bệnh viện thường không lên kế hoạch với số ngày lớn như vậy. Họ thường chỉ lên kế hoạch cho bệnh nhân trong giới hạn dưới một tháng. Điều này là hợp lí vì tùy tình hình diễn tiến bệnh của bệnh nhân mà kế hoạch sẽ thay đổi.

5. Chúng tôi tiến hành thí nghiệm với mô hình hai giai đoạn mà chúng tôi đề xuất. Mục tiêu là xem xét tình hiệu quả, ưu nhược điểm của mô hình hai giai đoạn so với các mô hình hồi quy kinh điển.

6.3 Kết quả thí nghiệm

6.3.1 So sánh kết quả các mô hình được đề xuất trên baseline-feature

Đầu tiên chúng tôi tiến hành so sánh kết quả dự đoán của ba mô hình hồi quy được đề xuất với nhóm đặc trưng **baseline-feature**. Để tiện cho việc theo dõi, chúng tôi nhắc lại các đặc trưng thuộc nhóm này:

- Tuổi bệnh nhân,
- Toại bảo hiểm bệnh nhân sử dụng,
- Giới tính của bệnh nhân,
- Loại hình cấp cứu ICU mà bệnh nhân được điều trị,
- Các chỉ số xét nghiệm của bệnh nhân trong 24 giờ ICU đầu tiên.

Bảng 6.3. Kết quả dự đoán trên các mô hình được đề xuất (đơn vị: ngày).

	Mean validation error (10-fold)	**Mean absolute error**
Support Vector Regression	**5.45**	**5.5**
Gradient Boosting	5.57	5.6
Random Forest	5.67	5.75

Nhìn vào kết quả trên, ta thấy mô hình hồi quy Support Vector Regression cho kết quả vượt trội. Kết quả cho thấy mô hình Support Vector Regression vừa cho kết quả tốt khi thực hiện 10-fold cross-validation, vừa cho kết quả tốt trên tập kiểm tra. Điều này chứng tỏ mô hình Support Vector Regression với bộ tham số đã chọn có tính tổng quát hóa tốt. Trong các thí nghiệm tiếp theo, chúng tôi sẽ chỉ sử dụng mô hình này để tiến hành các thí nghiệm.

6.3.2 So sánh kết quả với đặc trưng dữ liệu được đề xuất

Trong phần này, chúng tôi sử dụng mô hình Support Vector Regression để đánh giá mức độ cải thiện kết quả với nhóm đặc trưng về chẩn đoán bệnh, gọi là **group-diagnoses** mà chúng tôi đã đề xuất.

Bảng 6.4. So sánh kết quả với đặc trưng mới được đề xuất (đơn vị: ngày).

	Mean validation error (10-fold)	**Mean absolute error**
Baseline-features	5.45	5.5
Baseline-features + group-diagnoses	**4.85**	**4.9**

Chúng tôi nhận xét việc sử dụng nhóm đặc trưng **group-diagnoses** đã giúp cải thiện kết quả mô hình một cách đáng kể cả khi thực hiện 10-fold cross-validation và khi thực hiện kiểm tra trên tập kiểm thử. Kết quả độ lỗi **Mean absolute error** đều giảm được khoảng 0.6 ngày cả trên tập huấn luyện (kết quả khi thực hiện 10-fold cross-validation) và tập kiểm tra.

6.3.3 Kết quả khi thu hẹp điều kiện thí nghiệm và so sánh với các công trình khác

Nghiên cứu này thực hiện trong điều kiện rất tổng quát là dự đoán số ngày nằm viện của bệnh nhân nằm viện đến tối đa 90 ngày. Trong mục này, chúng tôi tiến hành thu hẹp điều kiện thí nghiệm bằng cách chỉ huấn luyện mô hình và dự đoán với các bệnh nhân nằm viện dưới 90 ngày, dưới 60 ngày, và dưới 30 ngày. Điều này là nhằm kiểm tra tính ứng dụng của mô hình trong thực tế. Vì các bệnh viện thường chỉ lên kế hoạch đối với bệnh nhân trong những khoảng thời gian ngắn hạn. Ở mục này chúng tôi sẽ sử dụng toàn bộ các đặc trưng dữ liệu, bao gồm các đặc trưng cơ bản (**baseline-feature**) và nhóm đặc trưng được chúng tôi đề xuất (group-diagnoses) để tiến hành thí nghiệm. Chúng tôi sử dụng mô hình Support Vector Regression trong thí nghiệm này.

Bảng 6.5. So sánh kết quả khi giới hạn thời gian nằm viện (đơn vị: ngày).

Thời gian nằm viện của bệnh nhân	Mean validation error (10-fold)	Mean absolute error
Dưới 90 ngày	4.85	4.9
Dưới 60 ngày	4.59	4.61
Dưới 30 ngày	**3.71**	**3.65**

Nhìn vào kết quả Bảng 6.5, chúng tôi nhận xét rằng việc thu hẹp điều kiện về số ngày nằm viện của bệnh nhân giúp việc dự đoán được chính xác hơn đáng kể.

Tiếp theo, chúng tôi muốn so sánh kết quả đạt được của mình với các nghiên cứu khác. Chúng tôi lựa chọn kết quả sử dụng mô hình hồi quy Support Vector Regression trong Bảng 6.5 với các bệnh nhân trên tập kiểm thử có thời gian nằm viện dưới 30 ngày để so sánh với kết quả của các công trình khác. Lí giải vì sao chúng tôi lựa chọn kết quả này để so sánh: Các nghiên cứu khác trên bộ dữ MIMIC-III sử dụng mô hình hồi quy để dự đoán thường làm việc với bài toán dự đoán số ngày bệnh nhân nằm trong phòng cấp cứu (ICU). Chúng tôi nhận xét rằng đây là bài toán dễ hơn của chúng tôi. Vì thời gian nằm ICU thường ngắn và thời gian nằm viện tổng cộng luôn lớn hơn thời gian bệnh nhân nằm ICU. Giới hạn điều kiện kiểm thử với những bệnh nhân nằm viện dưới 30 ngày vừa làm kết quả trở nên thực tế, đồng thời điều kiện thí nghiệm vẫn rộng hơn các công trình liên quan để có thể so sánh được.

Chúng tôi muốn bình luận thêm rằng các nghiên cứu về dự đoán số ngày nằm viện trên bộ dữ liệu MIMIC-III phần lớn sử dụng các mô hình phân lớp, rất ít công trình sử dụng các mô hình hồi quy. Hơn nữa, các công trình sử dụng hồi quy phần lớn đều giới hạn loại bệnh của bệnh nhân được xem xét để đơn giản hóa bài toán.

Ở đây, chúng tôi so sánh kết quả trong nghiên cứu của mình với công trình của Harutyunyan [7] và các đồng nghiệp. Đây là một trong những công trình có điều kiện thí nghiệm gần với nghiên cứu của chúng tôi nhất. Trong bài báo này, họ nghiên cứu dự đoán số ngày nằm ICU của bệnh nhân sử dụng hồi quy với các mô hình học sâu (deep learning). Kết quả **Mean absolute error** trên tập kiểm thử của họ đạt 3.91 ngày. Kết quả của chúng tôi là tốt hơn với chỉ số **Mean absolute error** đạt 3.65 ngày. Tuy nhiên chúng tôi nhấn mạnh rằng so sánh này chỉ mang tính

tham khảo vì yêu cầu hai bài toán trong hai nghiên cứu có sự khác nhau và cách trích xuất dữ liệu của hai nghiên cứu cũng không giống nhau.

Cuối cùng chúng tôi muốn so sánh kết quả trong nghiên cứu này với công trình của Rouzbahman và các cộng sự [6]. Bài báo này sử dụng bộ dữ liệu MIMIC-II (là một phiên bản cũ của bộ dữ liệu MIMIC-III). Nghiên cứu của họ tập trung vào việc xây dựng một mô hình hai giai đoạn (giai đoạn phân cụm và giai đoạn xây dựng từng mô hình hồi quy riêng cho mỗi cụm dữ liệu). Kết quả của họ đạt được là 5.07 (ngày) với độ lỗi **Mean absolute error** trên tập kiểm thử. Kết quả của chúng tôi (4.9 (ngày)) tốt hơn của họ ngay cả trong điều kiện tổng quát nhất trong điều kiện thí nghiệm của chúng tôi (LOS $<$ 90 ngày).

6.3.4 Kết quả với mô hình hai giai đoạn

Trong mục này chúng tôi tiến hành kiểm tra kết quả khi sử dụng mô hình hai giai đoạn (gọi là **divide-classify-regress**) do chúng tôi đề xuất.

Bảng 6.6. Kết quả dự đoán của mô hình hai giai đoạn (đơn vị: ngày).

	mean validation error (10-fold)	mean absolute error
Support vector regression	4.85	4.9
Divide-classify-regress	**4.89**	**4.98**

Nhìn vào Bảng 6.6, ta thấy kết quả dự đoán mô hình hai giai đoạn không tốt so với mô hình Support Vector Regression khi thực hiện 10-fold cross-validation trên tập huấn luyện và khi kiểm tra trên tập kiểm thử. Tuy nhiên, trong phần tiếp theo, chúng tôi sẽ phân tích lí do một mô hình hai giai đoạn như thế này vẫn hữu ích trên thực tế. Nhắc lại rằng, nghiên cứu này có mục tiêu dùng các mô hình hồi quy để dự đoán chính xác số ngày nằm viện của bệnh nhân. Tuy nhiên, với các bệnh nhân có số ngày nằm viện dài (trên 30 ngày). Việc dự đoán chính xác số ngày nằm viện là không khả thi và cũng không thực tế, vì thông thường bệnh viện không lên kế hoạch cho từng bệnh nhân với thời gian dài đến vậy. Mô hình hai giai đoạn của chúng tôi với bước phân loại bệnh nhân nằm viện dưới 30 ngày hay trên 30 ngày giúp phần nào giải quyết vấn đề đó. Mô hình này giúp dự đoán đúng thêm một số bệnh nhân sẽ nằm viện dài ngày. Điều này là cực kì có ích với gia đình các bệnh nhân vì giúp họ sẽ có sự chuẩn bị về mặt thời gian và tài chính được tốt hơn.

Bảng 6.7. Ma trận nhầm lẫn cho các nhãn dữ liệu trên tập kiểm thử.

		Nhãn thực tế	
		Ngắn ngày	Dài ngày
Nhãn dự đoán	Ngắn ngày	5767	266
	Dài ngày	30	25

Nhìn vào Bảng 6.7, ta thấy mô hình hai giai đoạn với bước phân loại bệnh nhân giúp dự đoán đúng 25 bệnh nhân sẽ nằm viện dài ngày trên tổng số 291 bệnh nhân nằm viện dài ngày trên tập kiểm thử. Kết quả này tuy còn khiêm tốn nhưng vẫn rất có ích trên thực tế.

Chương 7

Tổng kết

7.1 Kết quả đạt được

Trong luận văn này, chúng tôi đã xây dựng được hệ thống dự đoán chính xác thời gian nằm viện của bệnh nhân sử dụng mô hình hồi quy. Các đặc trưng chúng tôi sử dụng một phần dựa trên các đặc trưng được đề xuất trong nghiên cứu trước đó như các thông tin cơ bản của bệnh nhân (tuổi, giới tính, loại bảo hiểm được sử dụng, loại cấp cứu ICU được sử dụng cho bệnh nhân, các chỉ số xét nghiệm của bệnh nhân). Chúng tôi còn đề xuất đặc trưng về các chẩn đoán bệnh của bác sĩ đối với bệnh nhân. Tuy rằng thông tin chẩn đoán bệnh của bệnh nhân đã được sử dụng trong các nghiên cứu trước đó, tuy nhiên phương pháp của chúng tôi giúp giảm số lượng biến khi sử dụng đặc trưng về các chẩn đoán bệnh một cách đáng kể. Các kết quả thí nghiệm cho thấy đề xuất của chúng tôi giúp cải thiện khả năng dự đoán một cách đáng kể, giảm chỉ số **Mean absolute error (MAE)** trên tập kiểm thử từ 5.5 ngày xuống còn 4.9 ngày.

Kết quả của chúng tôi khi giới hạn điều kiện bài toán gần với thực tế hơn (chỉ dự đoán với các bệnh nhân trên tập kiểm thử có thời gian nằm viện dưới 30 ngày) là 3.65 ngày (**MAE**). Kết quả này là tích cực hơn khi so sánh với công trình của Harutyunyan [7] và các đồng nghiệp. Tuy nhiên, chúng tôi muốn nhấn mạnh rằng sự so sánh trên là tương đối vì các điều kiện thí nghiệm trong hai nghiên cứu không giống nhau hoàn toàn.

Ngoài ra, trong nghiên cứu này, chúng tôi còn đề xuất thêm một mô hình hai giai đoạn. Mô hình của chúng tôi tuy cho kết quả hồi quy không tốt bằng mô hình hồi quy kinh điển Support Vector Regression, tuy nhiên sự chênh lệch trên tập kiểm thử là nhỏ. Bù lại, mô hình hai giai đoạn của chúng tôi có ưu điểm hơn ở khía canh sau: với bước phân loại bệnh nhân, chúng tôi dự đoán đúng được thêm được 25 trên tổng số 291 bệnh nhân sẽ nằm viện trên 30 ngày. Kết quả này là tích cực, vì bình thường rất khó dự đoán chính xác (sử dụng các mô hình hồi quy kinh điển) số ngày nằm viện với các bệnh nhân nằm viện dài ngày đến vậy. Điều này mang lại nhiều lợi ích cho cả bệnh viện cũng như bệnh nhân.

7.2 Hạn chế và hướng phát triển

Trong nghiên cứu này, chúng tôi chưa tận dụng triệt để bản chất chuỗi thời gian của các đặc trưng xét nghiệm (Các đặc trưng xét nghiệm được đo lại thường xuyên sau mỗi giờ đồng hồ). Nếu biểu diễn các thông tin này dưới dạng dữ liệu chuỗi thời gian nhằm thể hiện rõ hơn diễn tiến bệnh của bệnh nhân thì có khả năng sẽ cải thiện kết quả dự đoán.

Ngoài ra, việc số bệnh nhân nằm viện trên 30 ngày quá ít so với số lượng bệnh nhân nằm viện ngắn ngày làm cho mô hình bị thiên lệch, dẫn đến kết quả dự đoán không tốt trên nhóm bệnh nhân nằm viện dài ngày. Trong các nghiên cứu sắp tới, chúng tôi dự định sẽ nghiên cứu kĩ hơn các phương pháp chọn lại mẫu cũng như thay đổi cấu trúc hàm mất mát nhằm giải quyết các vấn đề về mất cân bằng dữ liệu này.

Appendix A

Ý nghĩa các chỉ số xét nghiệm

Trong mục này, chúng tôi trình bày ý nghĩa của các chỉ số xét nghiệm trong bộ dữ liệu MIMIC-III.

- **alanine aminotransferase**: Alanine aminotransferase (ALT) là một loại men được tìm thấy ở trong tế bào gan và có vai trò hỗ trợ phân giải protein để cơ thể có thể dễ dàng hấp thụ. Khi gan có dấu hiệu viêm hoặc tổn thương, lượng ALT trong máu tăng lên, do vậy khi nghi ngờ bệnh về gan, bác sĩ thường sử dụng xét nghiệm này để chẩn đoán bệnh.
- **albumin**: Albumin là một thành phần protein quan trọng nhất của huyết thanh, chiếm đến 58-74% lượng protein toàn phần. Xét nghiệm này dùng để chẩn đoán các bệnh tiểu đường hay huyết áp cao, một số loại bệnh có ảnh hưởng xấu đến chức năng của thận.
- **albumin ascites**: Khi chỉ số albumin trong máu giảm, có thể dẫn đến tình trạng báng bụng, cổ trướng, tức là tình trạng ứ đọng dịch ở khoang phúc mạc. Đây là dấu hiệu của bệnh xơ gan.
- **albumin pleural**: Khi lượng albumin trong máu giảm, có thể là một trong những nguyên nhân dẫn đến tràn dịch màn phổi.
- **albumin urine**: Albumin urine là một xét nghiệm được thực hiện để đo lượng albumin trong nước tiểu. Một lượng nhỏ albumin trong nước tiểu là dâu hiệu cảnh báo sớm bệnh nhân có nguy cơ tổn thương thận.
- **Alkaline Phosphatase**: Xét nghiệm Alkaline Phosphatase được thực hiện để đo hoạt độ enzym phosphatase kiềm trong máu ở những bệnh nhân nghi ngờ mắc các bệnh về gan hoặc xương.
- **anion gap**: Khoảng trống Anion (AG) là sự chênh lệch giữa các cation và anion ở phần ngoại bào. Những tính toán này cho phép bác sĩ xác định được nguyên nhân bạn bị nhiễm toan chuyển hóa ví dụ như do sự tích tụ của acid lactic (biến chứng của tình trạng sốc do mất máu quá nhiều hay khó thở) hay do sự tích tụ của các ceton trong máu (biến chứng của bệnh đái tháo đường) và xét nghiệm cũng cho biết lượng bicarbonate để trung hòa chúng và duy trì pH trong máu.

- **asparate aminotransferase**: Asparate aminotransferase (AST) là loại enzyme được tìm thấy chủ yếu ở các tế bào của gan và thận, một lượng nhỏ hơn của nó cũng được tìm thấy trong cơ tim và cơ xương. Chỉ số AST cao có thể báo hiệu tổn thương tế bào gan, cũng có thể là dấu hiệu tổn thương các cơ quan khác như tim hay thận. Do đó, bác sĩ thường chỉ định xét nghiệm AST cùng các xét nghiệm đánh giá chức năng gan khác.

- **basophils**: Chỉ số này còn được gọi là lượng Bạch cầu ái kiềm. Chỉ số này có vai trò quan trọng trong các phản ứng dị ứng. Chỉ số basophils tăng trong bệnh leukemia mạn tính, sau phẫu thuật cắt lách, bệnh đa hồng cầu, nhiễm đọc, các phản ứng dị ứng.... giảm do tổn thương tủy xương, stress, quá mẫn.

- **bicarbonate**: bicarbonate là một ion mang điện tích âm được bài tiết và hấp thụ bởi thận. Nó được cơ thể sử dụng để duy trì lương acids-base của cơ theer(pH), và cùng với natri, kali, clorua duy trì cân bằng điện ở tế bào. Xét nghiệm bicarbonate thường không được làm một mình, mà thường được chỉ định làm kèm với việc đo các chỉ số natri, clorua, kali để có thể hình thành bảng chất điện điện giải. Xét nghiệm này nhằm chẩn đoạn việc mất cân bằng điện giải.

- **bilirubin**: bilirubin là sắc tố mật chính hình thành từ sự thoái giáng của heme trong tế bào hồng cầu. Xét nghiệm bilirubin trong máu là xét nghiệm đặc biệt cần thiết để đánh giá tình trạng sức khỏe của con người. Lượng bilirubin tăng cao (cả trực tiếp hoặc gián tiếp) hơn lượng bình thường sẽ có khả năng nhiễm các bệnh về gan cao hơn. Thông thường lượng bilirubin cao sẽ cho thấy tỷ lệ hủy hoại tế bào máu đỏ sẽ ngày càng tăng cao hơn. Đối với trẻ sơ sinh, việc xác định nhanh chóng nồng độ bilirubin trong máu cũng là một phương pháp quan trọng. Xét nghiệm kịp thời trước khi bilirubin gián tiếp bị dư thừa gây tổn thương tế bào não của trẻ. Hậu quả của tổn thương sẽ làm trẻ chậm phát triển trí tuệ, suy giảm khả năng học tập và phát triển. Ngoài ra còn khiến trẻ bị mất thính lực, rối loạn vận động mắt hoặc nặng hơn là tử vong.

- **blood urea nitrogen**: blood urea nitrogen (BUN), có nghĩa là lượng nitơ có trong ure. Ý nghĩa xét nghiệm BUN sẽ cho biết nồng độ urea nitrogen trong máu đang ở mức bình thường hay bất thường. Nếu như nồng độ urea nitrogen quá cao hoặc quá thấp thì có thể là dấu hiệu cảnh báo thận hay gan đang gặp vấn đề, cũng như một số tình trạng sức khỏe khác.

- **calcium**: Xét nghiệm calcium đo lượng canxi trong máu hoặc nước tiểu. Canxi là một trong những khoáng chất quan trọng nhất trong cơ thể. Xét nghiệm này được dùng để chẩn đoán bệnh sỏi thận, rối loạn thần kinh, các bệnh về xương.

- **calcium ionized**: Định lượng Calci ion hoá còn giúp chẩn đoán bệnh suy thận, ghép thận, cường cận giáp và các bệnh ác tính khác.

- **calcium urine**: Xét nghiệm canxi trong nước tiểu thực hiện nhằm mục đích đo lượng canxi được đào thải ra khỏi cơ thể qua nước tiểu. Xét nghiệm này còn được gọi là xét nghiệm

Ca2+ niệu. Nếu nồng độ canxi trong nước tiểu cao bất thường, đó có thể là dấu hiệu của một số bệnh như: bệnh cường tuyến giáp, suy thận, nhiễm đọc vitamin D.

- **cardiac index**: Đây là một chỉ số để đo tình trạng hoạt động của tim.
- **cardiac output fick**: Đây là một chỉ số để đo tình trạng hoạt động của tim.
- **cardiac output thermodilution**: Đây là một chỉ số để đo tình trạng hoạt động của tim.
- **central venous pressure** Chỉ số central venous pressure (CVP) thể hiện khối lượng tuần hoàn (thể tích) trong lòng mạch máu và khả năng làm việc của tim. Chỉ số bình thường của CVP là 4 - 10 cmH2O. Khi CVP lên cao trên 10 cmH2O có thể do sự giảm co bóp của tim hoặc do truyền dịch quá nhiều. Khi CVP thấp hơn 4cmH2O thường do thiếu khối lượng tuần hoàn.
- **chloride**: Xét nghiệm này dùng để đo nồng độ clo máu: định lượng các ion chính có trong huyết tương (điện giải đồ) để đánh giá trình trạng cân bằng nước trong cơ thể và để đánh giá cân bằng toan-kiềm.
- **chloride urine**: Xét nghiệm này dùng để đo nồng độ clo trong nước tiểu: để đánh giá tình trạng thể tích, khẩu phần muối và nguyên nhân gây hạ kali máu. Được chỉ định trong đánh giá chẩn đoán tình trạng nhiễm toan do ống thận. Ngoài ra nó còn được dùng để đánh giá các thành phần điện giải của nước tiểu và thăm dò thăng bằng toan-kiềm.
- **cholesterol**: Mỡ trong máu là tên gọi chung cho các loại mỡ tồn tại trong huyết dịch, bao gồm rất nhiều thành phần khác nhau. Trong mỡ máu, Cholesterol là thành phần quan trọng nhất, có mặt trong tât ca cac mô tô chưc cua cơ thê, tham gia vào quá trình xây dựng cấu trúc tế bào, vận hành chức năng não bộ, sản xuất hormone hay dự trữ vitamin. Cholesterol chỉ trở nên có hại khi rối loạn cholesterol xảy ra.
- **cholesterol hdl**: Xét nghiệm này thường được chỉ định cho bệnh nhân rối loạn mỡ máu, xơ vữa động mạch, tăng huyết áp hoặc kiểm tra sức khỏe định kỳ cho những người trên 40 tuổi. Cholesterol tăng: ít nguy cơ gây xơ vữa động mạch. Cholesterol giảm: dễ có nguy cơ gây xơ vữa động mạch, hay gặp trong các trường hợp rối loạn mỡ máu, xơ vữa động mạch, tăng huyết áp, cơn đau thắt ngực.
- **cholesterol ldl**: Xét nghiệm này chỉ định dành cho bệnh nhân rối loạn mỡ máu, xơ vữa động mạch, tăng huyết áp, bệnh mạch vành, đái tháo đường. Chỉ số cholesterol ldl càng cao, nguy cơ bị xơ vữa động mạch càng lớn. Chỉ số này giảm trong các trường hợp: xơ gan, hội chứng kém hấp thu, suy kiệt, cường tuyến giáp.
- **co2**: Xét nghiệm co2 là xét nghiệm máu đơn giản để đo lượng carbon dioxide trong máu. Đây là một phần của xét nghiệm điện giải. Và vì thận và phổi làm nhiệm vụ duy trì nồng độ co2 trong máu nên trong trường hợp nồng độ co2 trong máu cao hơn mức bình thường, bệnh nhân có thể được chỉ định kiểm tra chức năng thận và phổi.

- **creatinine**: Creatinine là sản phẩm của sự thoái hóa creatin trong các cơ, được đào thải qua thận và là chỉ số phản ánh chính xác chức năng của thận. Creatin đóng vai trò quan trọng cho việc sinh ra nguồn năng lượng cho các cơ hoạt động, creatin bị thoái dáng trong các cơ sẽ tạo thành creatinin và được lọc qua cầu thận. Trường hợp khi chúng không được cơ thể tái hấp thu ở ống thận thì sẽ phản ảnh chính xác chức năng lọc của thận. Khi nồng độ creatinine tăng cao đồng nghĩa với có rối loạn chức năng thận. Vì vậy khi chức năng thận bị suy chức năng thì khả năng lọc creatinine bị giảm dẫn tới nồng độ creatinine trong máu sẽ tăng cao hơn bình thường.

- **creatinine pleural**: Chỉ số này dùng để đánh giá lượng creatinine trong màng phổi so với trong máu. Đây là chỉ số để chẩn đoán bệnh viêm màng phổi (khi creatinine trong màng phổi tăng nhưng thấp hơn creatinine trong máu).

- **creatinine urine**: Chỉ số này dùng để đánh giá bệnh tràn dịch màng phổi (khi creatinine trong màng phổi tăng cao hơn creatinine trong máu).

- **diastolic blood pressure**: Huyết áp tâm trương là lực tác động của máu lên thành động mạch ở thì tâm trương (khi cơ tim được thả lỏng), đây là mức huyết áp thấp nhất trong mạch máu. Khi huyết áp tâm trương cao, mạch máu sẽ trở nên ít đàn hồi, cứng lại và xơ vữa. Huyết áp tâm trương bình thường thường dao động từ 60 - 80mmHg. Nếu huyết áp tâm trương của là 80 - 89 mmHg, cần chú ý đặc biệt vì bạn đã có tiền tăng huyết áp. Áp suất tâm trương thường thay đổi trong suốt cả ngày. Người bệnh nên kiểm tra huyết áp vài lần một ngày để có được con số trung bình. Các yếu tố gây ra sự dao động áp lực tâm trương bao gồm: sử dụng nicotine, mức độ căng thẳng.

- **eosinophils**: Tăng bạch cầu ái toan (eosinophils) là tình trạng các tế bào bạch cầu ái toan trong máu, trong mô hoặc một số tạng tăng lên một cách bất bình thường. Tình trạng này cũng có thể là quá trình hình thành bạch cầu ái toan bị rối loạn, hoặc là tích tụ bất thường hay thiếu hụt một loại bạch cầu nào đó. Khi loại bạch cầu này tăng lên thường liên quan đến đáp ứng điều hòa miễn dịch, xảy ra ở nhiều quá trình bệnh lý, bao gồm phản ứng viêm, dị ứng, ung thư và nhiễm ký sinh trùng.

- **fibrinogen**: Xét nghiệm fibrinogen đánh giá nồng độ protein này trong huyết tương, được sử dụng trong một số trường hợp: phát hiện một tình trạng viêm nhiễm, xem xét chức năng đông máu đường chung của cơ thể, theo dõi tình trạng tiến triển của người bệnh mắc các bệnh về gan.

- **fraction inspired oxygen**: Đây là thuật ngữ y học chỉ nồng độ oxy trong hỗn hợp khí thở vào.

- **glascow coma scale total**: Chỉ số này dùng để đánh giá tình trạng ý thức của người bệnh một cách lượng hóa. Nó được thiết lập để lượng giá độ hôn mê của nạn nhân bị chấn thương đầu.

- **glucose**: Chỉ số này dùng để chẩn đoán cho bệnh nhân bị bệnh đái tháo đường.
- **heart rate**: Xét nghiệm này dùng để đo nhịp tim.
- **height**: Chỉ số cân nặng của cơ thể.
- **hematocrit**: Đây là tỷ lệ thể tích hồng cầu trên thể tích máu toàn phần. Chỉ số này giảm khi mất máu, thiếu máu, xuất huyết. Chỉ số này tăng khi mắc bệnh phổi, bệnh tim mạch, mất nước, chứng tăng hồng cầu.
- **hemoglobin**: Chỉ số này cho biết lượng huyết sắc tố trong một thể tích máu. Chỉ số này tăng khi mất nước, bệnh tim mạch, bỏng. Chỉ số này giảm khi thiếu máu, xuất huyết, tán huyết.
- **lactate**: Xét nghiệm lactate trong máu là phương pháp xét nghiệm được tiến hành nhằm đo lường lượng lactate có trong máu người bệnh, một số ít các trường hợp là trong dịch não tủy. Lactate là sản phẩm được tạo ra trong quá trình chuyển hóa tế tế bào. Đôi khi lactate có thể xuất hiện dưới dạng axit lactic tùy thuộc vào độ pH. Tuy nhiên, cơ thể con người có độ pH trung tính nên thường xuất hiện trong máu dưới dạng lactate. Ở người bình thường, nồng độ lactate trong máu và trong dịch não tủy thường thấp. Chỉ khi các tế bào không có đủ oxy hoặc khi con đường sản xuất năng lượng trong các tế bào bị đứt đoạn, khi đó các tế bào hồng cầu, tế bào cơ, não và các mô khác sẽ sản xuất lượng lactate vượt quá mức. Khi cơ thể một người xuất hiện tình trạng tăng đáng kể nồng độ acid lactic trong máu (hay tăng lactate) có thể tiến triển thành nhiễm acid lactic. Tình trạng nhiễm acid lactic nghiêm trọng có thể dẫn đến sự phá vỡ cân bằng của axit và bazo, gây ra các triệu chứng như buồn nôn, nôn, đổ mồ hôi, thậm chí hôn mê.
- **lactate dehydrogenase**: Khi bệnh nhân mắc các bệnh gây tổn thương lên tế bào các cơ quan và mô trong cơ thể như gan, tim, thận thì việc tiến hành xét nghiệm nhằm khảo sát tình trạng tổn thương của các tế bào rất cần thiết. Xét nghiệm định lượng lactate dehydrogenase là một xét nghiệm chuyên biệt, hỗ trợ đáng kể cho công tác chẩn đoán và điều trị.
- **lactate dehydrogenase pleural**: Xét nghiệm này dùng để chẩn đoán hiện tượng tràn dịch màn phổi.
- **lactic acid**: Xét nghiệm axit lactic, thường được thực hiện trên mẫu máu lấy từ tĩnh mạch ở cánh tay, nhưng cũng có thể được thực hiện trên mẫu máu lấy từ động mạch. Xét nghiệm axit lactic là xét nghiệm máu đo mức axit lactic được tạo ra trong cơ thể. Hầu hết nó được tạo ra bởi mô cơ và tế bào hồng cầu. Khi nồng độ oxy trong cơ thể bình thường, carbohydrate sẽ phân hủy thành nước và carbon dioxide. Khi mức oxy thấp, carbohydrate bị phá vỡ để lấy năng lượng và tạo ra axit lactic. Nồng độ axit lactic rất cao gây ra một tình trạng nghiêm trọng, đôi khi đe dọa đến tính mạng được gọi là nhiễm axit lactic. Nhiễm axit lactic cũng có thể xảy ra ở một người dùng metformin (Glucophage) để kiểm soát bệnh tiểu đường khi bị suy tim hoặc suy thận hoặc nhiễm trùng nặng..

- **lymphocytes**: Lymphocyte là các tế bào có khả năng miễn dịch, gồm lympho T và lympho B. Lymphocyte tăng trong trường hợp nhiễm khuẩn, bệnh bạch cầu dòng lympho, suy tuyến thượng thận,..Giảm trong nhiễm HIV/AIDS, lao, ung thư, thương hàn nặng, sốt rét.

- **lymphocytes ascites**: Xét nghiệm này được dùng kết hợp với một số xét nghiệm khác. Xét nghiệm tìm thấy nhiều tế bào biểu mô, lymphocyte và một ít hồng cầu: nguyên nhân do bệnh tim, giang mai hoặc lao. Xét nghiệm thấy nhiều tế bào biểu mô, một ít hồng cầu và tế bào lymphocyte: nguyên nhân do bệnh thận hoặc tim.

- **lymphocytes atypical**: Lymphocyte không điển hình, đặc trưng thường thấy khi nhiễm siêu vi.

- **lymphocytes percent**: Khi xét nghiệm tỷ lệ phần trăm bạch cầu lympho tăng: số lượng bạch cầu bình thường, tỷ lệ phần trăm bạch cầu lympho tăng, số lượng tuyệt đối bạch cầu lympho tăng nhẹ có thể lưu ý tới một số trường hợp như: lao, nhiễm virus: thủy đậu, sởi, ho gà, rubella, các bệnh lý có tính chất mạn tính về hệ tiêu hóa, hô hấp.

- **lymphocytes pleural**: Xét nghiệm này dùng để chẩn đoán bệnh lao màng phổi.

- **magnesium**: Xét nghiệm cho magie được thực hiện để: Tìm một nguyên nhân cho các vấn đề về thần kinh và cơ, chẳng hạn như co giật cơ, khó chịu và yếu cơ. Tìm nguyên nhân của các triệu chứng như huyết áp thấp, buồn nôn, nôn, tiêu chảy, chóng mặt, yếu cơ và nói chậm. Theo dõi chức năng thận.

- **mean blood pressure**: Chỉ số huyết áp trung bình.

- **mean corpuscular hemoglobin**: Mean Corpuscular Hemoglobin là lượng huyết sắc tố trung bình có trong một hồng cầu. Giá trị bình thường: 24- 33pg. Tăng: thiếu máu đa sắc hồng cầu bình thường, chứng hồng cầu hình tròn di truyền nặng, sự có mặt của các yếu tố ngưng kết lạnh. Giảm: bắt đầu thiếu máu thiếu sắt, thiếu máu nói chung, thiếu máu đang tái tạo.

- **mean corpuscular hemoglobin concentration**: Mean Corpuscular Hemoglobin Concentration là nồng độ huyết sắc tố trung bình trong một thể tích máu).Giá trị bình thường: 316 – 372 g/L. Tăng: thiếu máu đa sắc hồng cầu bình thường, chứng hồng cầu hình tròn di truyền nặng, sự có mặt của các yếu tố ngưng kết lạnh. Trong thiếu máu đang tái tạo, có thể bình thường hoặc giảm do giảm folate hoặc vitamin B12.

- **mean corpuscular volume**: Mean corpuscular volumelà thể tích trung bình của một hồng cầu. Tăng trong thiếu máu hồng cầu to do thiếu hụt vitamin B12, thiếu acid folic, bệnh gan, chứng tăng hồng cầu. Giảm trong thiếu máu thiếu sắt, thalassemia, thiếu máu do các bệnh mạn tính.

- **monocyte**: Monocyte là một dạng tế bào bạch cầu trong cơ thể chúng ta. Những bạch cầu Monocyte này có tác dụng bảo vệ cơ thể khỏi những tác nhân gây bệnh. Chúng có chức

năng thực hiện quá trình thực bào. Bạch cầu Monocyte có thời gian lưu hành trong máu thường ngắn, thông thường dưới 20 giờ. Trong máu, chúng sẽ xuyên mạch và tăng kích thước trở thành các đại thực bào tổ chức. Khi đã ở dạng này, bạch cầu Monocyte có thể sống trong thời gian rất dài và có khả năng chống lại tác nhân gây bệnh, đặc biệt là chống lại tình trạng nhiễm trùng. Trong xét nghiệm máu, chỉ số bạch cầu Monocyte thường thay đổi theo từng loại bệnh. Và chúng có thể bị ảnh hưởng bởi việc sử dụng thuốc và một số bệnh lý khác trước khi thực hiện xét nghiệm. Tùy thuộc vào từng trường hợp mà các bác sĩ sẽ đưa ra kết luận chính xác rằng chỉ số này tăng do nguyên nhân nào.

- **neutrophils**: Chỉ số neutrophils thường tăng trong trường hợp người bệnh bị nhiễm trùng cấp. Ngoài ra, chỉ số này còn tăng trong trường hợp người bệnh bị nhồi máu cơ tim, nhiễm khuẩn cấp, bệnh bạch cầu dòng tủy và giảm trong trường hợp bệnh nhân bị nhiễm độc kim loại nặng, những người sử dụng các loại thuốc ức chế miễn dịch, bệnh bạch cầu dòng tủy.
- **oxygen saturation**: Độ bão hào oxy trong máu được xem là một trong những dấu hiệu sinh tồn của cơ thể, bên cạnh các dấu hiệu như: nhiệt độ, mạch, nhịp thở và huyết áp. Khi bị thiếu oxy máu, các cơ quan như tim, gan, não... sẽ chịu tác động tiêu cực rất nhanh. Vì vậy, cần theo dõi chỉ số độ bão hào oxy trong máu thường xuyên để kịp thời can thiệp nếu xảy ra tình trạng nguy hiểm.
- **partial pressure of carbon dioxide**: Đây là chỉ số đo lượng carbon dioxide trong động mạch hay tĩnh mạch.
- **partial thromboplastin time**: Đây là xét nghiệm huyết học được thực hiện trong quá trình khám sức khỏe định kỳ, thăm khám chức năng gan, bệnh lý đông cầm máu hoặc xét nghiệm bắt buộc trước khi thực hiện phẫu thuật.
- **peak inspiratory pressure**: Áp lực hô hấp tối đa là mức áp suất cao nhất áp dụng cho phổi trong quá trình hít vào.
- **pH**: Chỉ số pH trong cơ thể.
- **pH urine**: Chỉ số pH trong nước tiểu.
- **phosphorous**: Xét nghiệm này dùng để chẩn đoán việc mất cân bằng canxi và phôt pho trong cơ thể.
- **plateau pressure**: Đây là áp suất trung bình giữa đường thở và phế nang.
- **platelets**: Lượng tiểu cầu thấp có thể có nguyên nhân không phải do bệnh lý có từ trước. Ví dụ như mang thai, độ cao hoặc tác dụng phụ của thuốc.
- **positive end expiratory pressure**: Positive end expiratory pressure là người bệnh có áp lực khí cuối thì thở ra luôn dương, không do thầy thuốc đặt trên máy (bình thường áp lực cuối thì thở ra của máy phải bằng 0). áp lực này cao làm giảm thông khí phế nang chấn thương do áp lực gây vỡ phế nang.

- **post void residual**: Nước tiểu tồn dư sau khi tiểu tiện.
- **potassium**: Chỉ số Kali máu giúp duy trì cân bằng nội môi của cơ thể, chính vì vậy mà thiếu kali máu hay tăng kali máu đều có thể gây nguy hiểm cho sức khỏe. Kali máu giảm có thể gây ra triệu chứng như mệt mỏi rã rời cơ thần kinh, táo bón dai dẳng và chướng bụng. Nếu nặng hơn có thể gây rối loạn nhịp tim, cơn nhịp nhanh thất, rung thất dẫn tới tử vong. Kali máu tăng khi có nồng độ kali máu > 5 mmol/l. Khi tỷ lệ này lớn hơn 7 mmol/l sẽ gây nguy hiểm đến tính mạng.
- **prothrombin time inr**: Thời gian prothrombin (PT) là xét nghiệm máu đo thời gian để đông máu. Xét nghiệm thời gian prothrombin có thể được sử dụng để kiểm tra các vấn đề chảy máu. PT cũng được sử dụng để kiểm tra xem thuốc để ngăn ngừa cục máu đông có hoạt động hay không.
- **pulmonary artery pressure**: Áp lực động mạch phổi. Đây là một đại lượng được dùng nhiều trên thực hành lâm sàng trong chẩn đoán, tiên lượng, chỉ định điều trị và đánh giá kết quả điều trị đối với nhiều bệnh tim mạch.
- **pulmonary artery pressure systolic**: Chỉ số đo áp lực động mạch phổi.
- **pulmonary capillary wedge pressure**: Chỉ số đo áp lực động mạch phổi bít.
- **red blood cell count**: Số lượng hồng cầu có trong một đơn vị máu. Chỉ số này tăng trong các trường hợp: cô đặc máu, bệnh đa hồng cầu nguyên phát, thiếu oxy kéo dài. Chỉ số này giảm khi thiếu máu, mất máu, suy tỷ.
- **red blood cell count urine**: Nếu xét nghiệm nước tiểu phát hiện trong nước tiểu có lượng hồng cầu cao thì bệnh nhân có thể đã mắc các bệnh về thận hoặc hệ tiết niệu, như nhiễm trùng đường tiểu, sỏi thận, bướu thận, xuất huyết bàng quang.
- **respiratory rate**: Tần số hô hấp điển hình cho một người lớn khỏe mạnh là 12-20 lần mỗi phút. Trung tâm hô hấp điều hòa và kiểm soát nhịp thở được cố định khoảng hai giây cho một lần hít vào và ba giây thở ra. Cho phép tần số hô hấp trung bình hạ mức trung bình 12 lần mỗi phút
- **sodium**: Natri trong máu mức bình thường là 135-145 mmol/l. Natri là cation chủ yếu ở dịch ngoại bào, cùng với Clo, Bicarbonat...đóng vai trò quan trọng trong điều hòa cân bằng nước và duy trì áp suất thẩm thấu cho dịch ngoại bào.
- **systemic vascular resistance**: Kháng trở mạch máu hệ thống.
- **systolic blood pressure**: Chỉ số cao huyết áp.
- **temperature**: Nhiệt độ của cơ thể.
- **tidal volume observed**: Đây là thể tích khí được tính bằng ml, do máy thở đưa vào phổi người bệnh một lần khi hít vào. Thể tích này được tính theo cân nặng của người bệnh.

- **tidal volume spontaneous** Thể tích khí khi hít vào do bệnh nhân tự thở không có sự hổ trợ của máy thở.
- **total protetin**: Protein máu là những protein có trong huyết tương có chức năng vô cùng quan trọng được đánh giá là chỉ số quan trọng của cơ thể. Khi chỉ số protein máu bị thay đổi nó kéo theo nhiều bệnh lý nguy hiểm.
- **protein urine**: Protein niệu chỉ sự có mặt của protein trong nước tiểu. Bình thường trong nước tiểu không có hoặc có rất ít protein do cơ chế tái hấp thu protein ở thận.
- **troponin-i**: Chỉ số này có vai trò trong chẩn đoán theo dõi nhồi máu cơ tim cấp (AMI) và phân tầng nguy cơ bệnh tim mạch.
- **troponin-t** Chỉ số này chỉ định ở các bệnh nhân có nguy cơ hội chứng mạch vành tim cấp hoặc nhồi máu cơ tim thấp.
- **venous pvo2**: Chỉ số này phản ánh khả năng oxy hóa máu của phổi.
- **white blood cell count**: Đây là xét nghiệm dùng để để đo số lượng tế bào bạch cầu (WBC) có trong máu.
- **white blood cell count urine**: Xét nghiệm này dùng để đo số lượng bạch cầu có trong nước tiểu.

Tài liệu tham khảo

[1] W. H. Organization *et al.*, "Global spending on health: a world in transition," tech. rep., World Health Organization, 2019.

[2] H. Baek, M. Cho, S. Kim, H. Hwang, M. Song, and S. Yoo, "Analysis of length of hospital stay using electronic health records: A statistical and data mining approach," *PloS one*, vol. 13, no. 4, p. e0195901, 2018.

[3] W. E. Muhlestein, D. S. Akagi, J. M. Davies, and L. B. Chambless, "Predicting inpatient length of stay after brain tumor surgery: developing machine learning ensembles to improve predictive performance," *Neurosurgery*, vol. 85, no. 3, pp. 384–393, 2019.

[4] D. J. Whellan, X. Zhao, A. F. Hernandez, L. Liang, E. D. Peterson, D. L. Bhatt, P. A. Heidenreich, L. H. Schwamm, and G. C. Fonarow, "Predictors of hospital length of stay in heart failure: findings from get with the guidelines," *Journal of cardiac failure*, vol. 17, no. 8, pp. 649–656, 2011.

[5] T. Gentimis, A. Ala'J, A. Durante, K. Cook, and R. Steele, "Predicting hospital length of stay using neural networks on mimic iii data," in *2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech)*, pp. 1194–1201, IEEE, 2017.

[6] M. Rouzbahman, A. Jovicic, and M. Chignell, "Can cluster-boosted regression improve prediction of death and length of stay in the icu?," *IEEE journal of biomedical and health informatics*, vol. 21, no. 3, pp. 851–858, 2016.

[7] H. Harutyunyan, H. Khachatrian, D. C. Kale, G. Ver Steeg, and A. Galstyan, "Multitask learning and benchmarking with clinical time series data," *Scientific data*, vol. 6, no. 1, pp. 1–18, 2019.

[8] C. Bishop, "Pattern recognition and machine learning (information science and statistics), 1st edn. 2006. corr. 2nd printing edn," *Springer, New York*, 2007.

[9] S. Tong and D. Koller, "Restricted bayes optimal classifiers," in *AAAI/IAAI*, pp. 658–664, 2000.

[10] L. Breiman, "Bagging predictors," *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.

[11] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.

[12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[13] S. v. d. Walt, S. C. Colbert, and G. Varoquaux, "The NumPy array: a structure for efficient numerical computation," *Computing in Science & Engineering*, vol. 13, no. 2, pp. 22–30, 2011.

[14] W. McKinney *et al.*, "Data structures for statistical computing in python," in *Proceedings of the 9th Python in Science Conference*, vol. 445, pp. 51–56, SciPy Austin, TX, 2010.

[15] A. E. Johnson, T. J. Pollard, L. Shen, H. L. Li-Wei, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. A. Celi, and R. G. Mark, "Mimic-iii, a freely accessible critical care database," *Scientific data*, vol. 3, no. 1, pp. 1–9, 2016.

[16] A. E. Johnson, T. J. Pollard, and R. G. Mark, "Reproducibility in critical care: a mortality prediction case study," in *Machine Learning for Healthcare Conference*, pp. 361–376, 2017.

[17] K. Lin, Y. Hu, and G. Kong, "Predicting in-hospital mortality of patients with acute kidney injury in the icu using random forest model," *International journal of medical informatics*, vol. 125, pp. 55–61, 2019.

[18] S. Wang, M. B. McDermott, G. Chauhan, M. Ghassemi, M. C. Hughes, and T. Naumann, "Mimic-extract: A data extraction, preprocessing, and representation pipeline for mimic-iii," in *Proceedings of the ACM Conference on Health, Inference, and Learning*, pp. 222–235, 2020.