LITERATURE REVIEW

TIẾN BỘ VÀ THÁCH THỨC CỦA LĨNH VỰC HỌC MÁY TRONG HOÁ TIN

Lê Thị Thuỳ Hương[1,3], Trần Văn Lăng[2,*], Phạm Minh Quân[1]

[1]Viện Hoá học Các Hợp chất Thiên nhiên, VAST
[2] Trường Đại học Ngoại ngữ -Tin học TP.HCM
[3]Học viện Khoa học và Công nghệ, VAST

thuyhuong0102sp2@gmail.com, langtv@huflit.edu.vn, pham-minh.quan@inpc.vast.vn

TÓM TẮT— Học máy (Machine Leaning - ML) đã trở thành một trong những kỹ thuật mạnh mẽ trong Hóa tin hay Hoá tin học (Cheminformatics) còn gọi là hoá học tính toán, nó đã được ứng dụng trong nhiều bài toán khác nhau. Chẳng hạn, trong hoá học, học máy được dùng trong việc khám phá thuốc, dự đoán độc tính và thiết kế vật liệu. Trong bài báo này, nhằm mục đích cung cấp một khảo sát chung về ML trong hóa tin, chúng tôi bắt đầu bằng cách thảo luận về các khái niệm cơ bản của ML, sau đó xem xét các loại thuật toán ML khác nhau đã được áp dụng cho các bài toán Hóa tin. Qua đó cung cấp cho các nhà nghiên cứu và những người thực hành trong ngành Hoá tin hiểu biết thấu đáo về việc áp dụng những kỹ thuật, phương pháp của tin học; đồng thời đưa ra một số thách thức cũng như cơ hội để nghiên cứu phát triển. Phần cuối cùng trình bày một nghiên cứu thử nghiệm qua việc xác định hoạt tính dựa trên tập mẫu chứa các xét nghiệm sàng lọc do tổ chức Burnham Center for Chemical Genomics thực hiện, nhằm ức chế biểu hiện bề mặt tế bào VCAM-1 do gen TNFa gây ra. Đây là một tập mẫu có tỷ lệ chênh lệch rất cao giữa số lượng mẫu của hợp chấp có hoạt tính và không có hoạt tính. Kết quả cho thấy mô hình phân loại được lựa chọn tương đối phù hợp thông qua thước đo AUC và G-mean.

Từ khóa— Hoá tin, học phối hợp, mất cân bằng dữ liệu, học máy, hoạt tính sinh học.

I. GIỚI THIỆU

Hóa tin hay Hoá tin học (Cheminformatics) còn được gọi là Hoá học tính toán, đây là một ngành nổi lên như một ngành khoa học quan trọng, liên quan đến việc phát triển và ứng dụng các phương pháp tính toán để phân tích, lưu trữ và truy xuất thông tin hóa học. Thành tựu về Hóa tin xuất hiện ở nhiều lĩnh vực cũng như ngành khác. Chẳng hạn, (1) lĩnh vực khám phá thuốc, nhằm xác định các loại thuốc tiềm năng mới bằng cách sàng lọc cơ sở dữ liệu lớn về các hợp chất hóa học để tìm các hợp chất có đặc tính mong muốn; (2) ngành khoa học vật liệu, nhằm thiết kế các vật liệu mới với các đặc tính mong muốn bằng cách tìm kiếm các mẫu trong tập dữ liệu lớn về dữ liệu vật liệu; (3) ngành khoa học môi trường nhằm xác định và theo dõi các chất gây ô nhiễm môi trường bằng cách phân tích các tập dữ liệu lớn về các mẫu môi trường. Thực chất đây là một ngành mang tính đa ngành, bản thân nó nằm ở khu vực giao điểm của hóa học, khoa học máy tính, khoa học dữ liệu và công nghệ thông tin.

Đối với ngành Hoá học, những tiếp cận của Hoá tin có thể được sử dụng để giải quyết một số bài toán cụ thể như biểu diễn và phân tích nhiều loại thông tin hóa học như:

- Cấu tạo hoá học: Cấu tạo hoá học để biểu diễn các cấu trúc hoá học, qua đó thể hiện sự sắp xếp các nguyên tử trong phân tử. Cấu trúc hóa học có thể được thể hiện theo nhiều cách khác nhau, bao gồm chuỗi SMILES, mã InChI và bản vẽ 2D và 3D.
- Tính chất vật lý: Tính chất vật lý là tính chất của các phân tử có thể đo được, chẳng hạn như khối lượng phân tử, điểm nóng chảy và điểm sôi. Tính chất vật lý có thể được sử dụng để mô tả các phân tử và dự đoán hành vi của chúng.
- Hoạt tính sinh học: Hoạt tính sinh học là khả năng tương tác của một phân tử với các hệ thống sinh học, chẳng hạn như protein và tế bào. Hoạt động sinh học có thể được sử dụng để xác định các loại thuốc tiềm năng và thiết kế các vật liệu mới với các đặc tính mong muốn.

Từ đó có thể nói các thành tựu của Hoá tin cho phép các nhà nghiên cứu điều hướng trong không gian hóa học rộng lớn, có thể dự đoán các đặc tính phân tử và thiết kế các hợp chất nhằm mục tiêu ứng dụng cho những vấn đề cụ thể của hoá học đặt ra.

Bằng cách khai thác sức mạnh của các phương pháp tiếp cận dựa trên dữ liệu và mô hình tính toán, Hóa tin đóng vai trò then chốt trong việc hướng dẫn các quy trình ra quyết định, giảm chi phí và giảm thiểu thời gian cần thiết cho các thí nghiệm trong phòng thí nghiệm truyền thống. Đặc biệt, với sự tăng trưởng theo cấp số nhân của dữ liệu có sẵn và nhu cầu ngày càng tăng đối với các quy trình phát triển thuốc hiệu quả, hóa học đã trở nên không thể thiếu trong việc thúc đẩy khám phá và tối ưu hóa các phương pháp trị liệu mới.

Một số thành tựu của khoa học máy tính, khoa học dữ liệu, cũng như công nghệ thông tin. Đặc biệt trong số đó, phổ biến nhất là 2 lĩnh vực: học máy và khai thác dữ liệu đã được sử dụng nhiều trong Hoá tin.

* Corresponding Author

- Học máy (Machine Learning) là một phương pháp để tạo ra sản phẩm từ trí tuệ nhân tạo (không phải từ trí tuệ con người) thông qua việc xây dựng chương trình máy tính học từ dữ liệu mà con người thu thập được. Học máy được sử dụng trong hóa học nhằm khai thác thông tin từ dữ liệu rồi tự đề xuất mô hình – bản chất là thuật toán - để có thể dự đoán các đặc tính của phân tử, xác định các loại thuốc tiềm năng cũng như thiết kế các hợp chất mới.
- Khai thác dữ liệu (Data Mining) là một quá trình trích xuất các mẫu, các quy luật từ những tập dữ liệu lớn. Các phương pháp trong khai thác dữ liệu được sử dụng trong hóa học để xác định các mẫu trong dữ liệu hóa học có thể được sử dụng để cải thiện hiệu quả khám phá thuốc, khoa học vật liệu và khoa học môi trường.

Trong lĩnh vực học máy đã có những thành tựu đáng kể về mặt học thuật, và cũng không dừng lại ở đó, học máy đã xâm nhập vào rất nhiều lĩnh vực của thực tiễn đặt ra. Các thuật toán học máy có thể được phân loại một cách cơ bản thành hai loại: học có giám sát và học không giám sát. Các thuật toán học có giám sát nhằm học từ dữ liệu đã được gắn nhãn với đầu ra mong muốn. Ví dụ, một thuật toán học có giám sát có thể được huấn luyện để dự đoán độc tính của một hợp chất hóa học dựa trên cấu trúc của nó. Thuật toán học không giám sát nhằm học từ dữ liệu không được gắn nhãn. Ví dụ, một thuật toán học không giám sát có thể được sử dụng để phân cụm các hợp chất hóa học dựa trên sự giống nhau về cấu trúc của chúng.

Có nhiều thuật toán học máy khác nhau, mỗi thuật toán có điểm mạnh và điểm yếu riêng. Một số thuật toán học máy phổ biến nhất được sử dụng trong Hóa tin bao gồm:

- Máy vectơ hỗ trợ (SVM) là một thuật toán thuộc loại học có giám sát, có thể được sử dụng cho các nhiệm vụ phân loại hoặc hồi quy. SVM được biết đến với độ chính xác cao, nhưng chúng có thể mất thời gian về mặt tính toán để đào tạo.
- Mạng neural (Artificial Neural Network) cũng là thuật toán học có giám sát, được lấy cảm hứng từ bộ não con người. Mạng lưới thần kinh có thể được sử dụng cho nhiều nhiệm vụ, bao gồm phân loại, hồi quy và dự báo. Mạng lưới thần kinh có thể rất mạnh, nhưng chúng cũng có thể khó đào tạo và giải thích.
- Học phối hợp (Ensemble Learning) là một phương pháp trong đó nhiều mô hình học máy được kết hợp lại để tạo thành một mô hình mạnh hơn, thay vì sử dụng một mô hình đơn lẻ để dự đoán. Cơ chế hoạt động như vậy giúp mô hình phối hợp có khả năng kháng nhiễu, ổn định và có khả năng tổng quát hóa tốt hơn. Có nhiều phương pháp học phối hợp phổ biến, bao gồm:
    - Bagging: Kỹ thuật này sử dụng nhiều mô hình độc lập được huấn luyện trên các tập dữ liệu con được lấy mẫu ngẫu nhiên từ tập huấn luyện ban đầu. Kết quả dự đoán của các mô hình con được kết hợp để đưa ra dự đoán cuối cùng.
    - Boosting: Phương pháp này tạo ra một chuỗi các mô hình theo tuần tự, trong đó mỗi mô hình tiếp theo được huấn luyện để cố gắng sửa các lỗi dự đoán của mô hình trước đó. Các dự đoán của các mô hình con được kết hợp để đưa ra dự đoán tổng quát.
    - Random Forest: Đây là một phương pháp kết hợp của Bagging và Decision Tree. Random Forest sử dụng nhiều cây quyết định độc lập và kết hợp kết quả dự đoán của từng cây để đưa ra kết quả cuối cùng.
    - Stacking: Kỹ thuật này kết hợp dự đoán của nhiều mô hình con bằng cách sử dụng một mô hình meta-learner để học cách kết hợp các dự đoán này.

Các kỹ thuật trong học máy đã cách mạng hóa ngành Hóa tin bằng cách cung cấp các giải pháp mạnh mẽ và hiệu quả để phân tích dữ liệu, lựa chọn tính năng và lập mô hình dự đoán. Một số thuật toán học có giám sát (supervisor) phổ biến như:

- Linear Regression để dự đoán giá trị liên tục dựa trên các biến đầu vào, bằng cách tìm ra một mô hình tuyến tính tốt nhất để mô tả mối quan hệ giữa biến đầu vào và đầu ra.
- Logistic Regression sử dụng để dự đoán xác suất xảy ra của một biến phụ thuộc dựa trên các biến đầu vào. Thuật toán được sử dụng phổ biến trong bài toán phân loại nhị phân.
- Decision Trees là một kỹ thuật học giám sát sử dụng cấu trúc cây quyết định để phân loại hoặc dự đoán dựa trên các quyết định tại các nút trong cây.
- Random Forest dựa trên việc kết hợp nhiều cây quyết định thành một mô hình dự đoán. Từ đó tạo ra một tập hợp các cây quyết định độc lập và kết hợp kết quả từ các cây này để đưa ra dự đoán cuối cùng.
- Support Vector Machines (SVM) là một kỹ thuật học giám sát sử dụng để phân loại hoặc hồi quy, bằng cách tạo ra một siêu phẳng tối ưu để phân tách các lớp hoặc dự đoán giá trị đầu ra.
- Neural Networks hay còn gọi là mạng thần kinh nhân tạo (ANN – Artificial Neural Networks) là một kỹ thuật học giám sát dựa trên mô phỏng cấu trúc của não người.
- Gradient Boosting sử dụng để tạo ra một mô hình dự đoán mạnh từ các mô hình dự đoán yếu.

Các thuật toán này cho phép xây dựng các mô hình dự đoán liên quan đến các mô tả hóa học với các thuộc tính hoặc các hoạt tính cụ thể. Một số thuật toán học không giám sát (unsupervisor) như :

- K-means Clustering nhằm phân cụm hay phần chùm dữ liệu bằng cách tìm kiếm các tâm cụm (cluster centroids) và gán các điểm dữ liệu vào các cụm dựa trên sự tương đồng.
- DBSCAN (Density-Based Spatial Clustering of Applications with Noise) phân cụm dữ liệu dựa trên mật độ, xác định các vùng có mật độ cao và phân chia các điểm dữ liệu vào các cụm.
- Gaussian Mixture Models (GMM) dùng để mô hình hóa dữ liệu bằng phân phối Gaussian hỗn hợp, ước tính các tham số để tìm kiếm các cụm dựa trên phân phối này. Bản chất là thuật toán phân loại không giám sát.
- Hierarchical Clustering để xây dựng cấu trúc phân cấp của dữ liệu bằng cách liên kết các điểm dữ liệu gần nhau, từ việc hợp nhất các điểm dữ liệu thành các cụm lớn hơn cho đến khi có một cụm lớn duy nhất.
- Spectral Clustering nhằm phân tích ma trận đồng dạng để phân cụm.
- Mean Shift Clustering, thuật toán này dựa trên việc di chuyển các điểm dữ liệu trong không gian đến các vị trí có mật độ dữ liệu cao nhất.

Việc tích hợp học máy trong Hóa tin đã cho phép tạo ra một loạt các ứng dụng. Chẳng hạn:

- Sử dụng mô hình Quan hệ cấu trúc-hoạt động định lượng (Quantitative Structure-Activity Relationship - QSAR) để thiết lập mối tương quan định lượng giữa cấu trúc hóa học và hoạt tính sinh học, từ đó hỗ trợ xác định các ứng cử viên thuốc tiềm năng.
- Các phương pháp sàng lọc ảo (virtual screening) sử dụng thuật toán trong học máy để sàng lọc ban đầu các hợp chất lớn, từ đó xác định các phân tử có thuộc tính mong muốn để nghiên cứu thêm.
- Mô hình ADMET (Absorption – Hấp thụ, Distribution – Phân phối, Metabolism – Chuyển hoá, Excretion – Bài tiết, và Toxicity - Độc tính) sử dụng học máy để đánh giá hồ sơ an toàn và hiệu quả của các hợp chất hóa học.

Cũng chính vì vậy mà ngoài các thuật ngữ thông dụng, người ta còn có thuật ngữ "in silico" đề cập đến việc sử dụng máy tính để thực hiện các mô phỏng và phân tích dựa trên dữ liệu và thông tin được thu thập từ nhiều nguồn, bao gồm cơ sở dữ liệu, mô hình toán học và phương pháp tính toán. Thuật ngữ "in silico" này được tạo ra từ việc tương ứng với thuật ngữ "in vivo" (trong cơ thể sống) và "in vitro" (ngoài cơ thể sống) **[1]**.

Mặc dù có những tiến bộ đáng kể, việc sử dụng học máy trong Hóa tin cũng đặt ra một số thách thức. Khả năng diễn giải và khả năng giải thích của các mô hình học máy vẫn là một mối quan tâm hàng đầu, vì mối quan hệ phức tạp giữa các đặc tính và mô tả hóa học có thể không phải lúc nào cũng dễ dàng diễn giải. Ngoài ra, việc xử lý các bộ dữ liệu hóa học quy mô lớn và nhiều chiều yêu cầu các thuật toán hiệu quả và có thể mở rộng. Sự khan hiếm dữ liệu là một thách thức khác, đặc biệt là trong bối cảnh các sự kiện hiếm gặp trong tự nhiên.

Một số câu hỏi mang tính cơ bản trong ngành Hóa tin. Đó là (1) các nguồn dữ liệu chính và phương pháp được sử dụng để thu thập dữ liệu hóa học và sinh học là gì; (2) làm thế nào các cấu trúc hóa học có thể được biểu diễn và mã hóa một cách hiệu quả để phân tích và tính toán; (3) những phương pháp và thuật toán tính toán nào được sử dụng phù hợp cho mô hình dự đoán, sàng lọc ảo và thiết kế hợp chất phức tạp; (4) đồng thời, cũng phải đưa ra một số sản phẩm của Hóa tin, bao gồm các công cụ phần mềm như khám phá thuốc, hóa gen học và nhận dạng sản phẩm tự nhiên.

Phần II tiếp theo của bài báo sẽ trình bày về nguồn dữ liệu, một số công trình thu thập được gần đây liên quan đến nội dung bài viết, cũng như thách thức và cơ hội của ngành Hoá tin. Phần thứ III trình bày một vài phương pháp giải quyết cụ thể khi giải quyết bài toán Hoá tin đặt ra, trong đó trình bày phương pháp tiếp cận để giải quyết một bài toán cụ thể của Hoá tin đó là xác định hoạt tính của một hợp chất; phần tiếp theo sẽ trình bày một số kết quả thử nghiệm của bài toán đặt ra trong Phần IV. Cuối cùng một vài kết luận cũng như hướng đề xuất được trình bày trong phần V.

II. MỘT VÀI KHẢO SÁT LIÊN QUAN

A. NGUỒN DỮ LIỆU

Hiện nay có một số mẫu hay tập dữ liệu (dataset) phổ biến được sử dụng để huấn luyện và đánh giá các mô hình trong học máy cũng như trong khai thác dữ liệu dành cho Hoá tin. Một số ngân hàng dữ liệu phổ biến cung cấp các dataset này như:

- **PubChem** là một nguồn tài nguyên lớn chứa thông tin về hàng triệu hợp chất hóa học và hoạt tính sinh học tương ứng. PubChem cung cấp nhiều tập dữ liệu có thể được sử dụng trong các bài toán khác nhau của ngành Hóa tin, chẳng hạn cung cấp dữ liệu về cấu trúc hóa học, thuộc tính và hoạt tính của các hợp chất; bao gồm cả thông tin về tác nhân chống ung thư, chống viêm, chống nhiễm trùng và nhiều loại hoạt tính khác. Có thể tải xuống (download) tập dữ liệu liên quan tại địa chỉ https://pubchem.ncbi.nlm.nih.gov/docs/downloads.

- **DrugBank** là một ngân hàng dữ liệu chứa thông tin chi tiết về các dược phẩm đã được chấp thuận, bao gồm cả thông tin về cấu trúc hóa học, cơ chế tác động, liên kết với mục tiêu sinh học, và các thuộc tính khác. Để tải xuống bộ dữ liệu DrugBank phi thương mại này, cần tạo một tài khoản DrugBank miễn phí qua email tại địa chỉ https://go.drugbank.com/releases/latest.
- **ChEBI** chứa thông tin về các thực thể hóa học có liên quan đến sinh học. Trong đó bao gồm các thuộc tính, cấu trúc và quan hệ giữa các thực thể hóa học khác nhau. Có thể tìm thấy một số dataset liên quan tại địa chỉ https://www.ebi.ac.uk/chebi/; sau đó có thể dùng phần mềm Avogradro (https://avogadro.cc) để hiển thị cũng như chỉnh sửa các hợp chất. Chẳng hạn, tập tin SDF biểu diễn từng phân tử bao gồm các thông tin như tên phân tử, công thức hóa học, trạng thái vật lý, hoạt tính sinh học và các thuộc tính khác liên quan. Dataset này có tại địa chỉ https://ftp.ebi.ac.uk/pub/databases/chebi/SDF/.
- **ChEMBL** là một ngân hàng dữ liệu lưu trữ thông tin về hoạt tính sinh học và thông tin liên quan đến dược phẩm. ChEMBL chứa dữ liệu từ nhiều nguồn, bao gồm cả dữ liệu về hoạt tính và tương tác với các mục tiêu sinh học như enzyme và receptor. Cơ sở dữ liệu này cung cấp thông tin quan trọng để phân tích mối quan hệ cấu trúc-hoạt tính (SAR) qua đó để phát triển mô hình dự đoán hoạt tính.
- **ZINC database**: Đây là một nguồn tài nguyên chứa thông tin về hàng triệu hợp chất hóa học. Cơ sở dữ liệu ZINC cung cấp dữ liệu về cấu trúc hóa học, tính chất và một số đặc tính sẵn có của các hợp chất. Tập dữ liệu này có thể được sử dụng để nghiên cứu khám phá thuốc, tìm kiếm các hợp chất tiềm năng và phát triển mô hình dự đoán.
- **MoleculeNet** là một bộ sưu tập các tập dữ liệu Hoá tin phổ biến được tạo ra bởi đội ngũ nghiên cứu và phát triển học máy Stanford's Medical AI Lab tại Trường Đại học Stanford. Dataset này bao gồm nhiều tập dữ liệu đa dạng như dự đoán hoạt tính, dự đoán tính hoà tan, dự đoán độ bền với enzyme, và nhiều tác vụ khác.
- **QM9** là dataset chứa thông tin về khoảng 134 ngàn phân tử hữu cơ nhỏ gọn được tạo ra từ các phân tử lớn hơn bằng cách áp dụng các phép biến đổi hóa học tiêu chuẩn. Trong đó bao gồm các đặc trưng hóa học, năng lượng, cấu trúc phân tử và các thuộc tính liên quan.
- **Tox21** là một tập dữ liệu chứa thông tin về sự độc tính của hàng ngàn hợp chất hóa học đối với một số mục tiêu sinh học. Đây là một tập dữ liệu phổ biến trong lĩnh vực dự đoán độc tính của các chất hoá học và đánh giá tính an toàn của chúng.

B. CÔNG TRÌNH LIÊN QUAN

Trong Hoá tin nói chung có rất nhiều công trình có chỉ số trích dẫn cao. Chẳng hạn công trình "*Chemoinformatics in Drug Discovery*" của Jun Xu và Arnold Hagler (2002) **[2]** trình bày những thành tựu trong ngành Hóa tin và tác động của chúng đối với các quy trình khám phá thuốc. Một số phương pháp khai thác dữ liệu chính được sử dụng như tính toán mô tả, tính toán ma trận tương đồng cấu trúc và thuật toán phân loại. Một số ứng dụng của Hóa tin trong khám phá thuốc; chẳng hạn như chọn hợp chất, sàng lọc ảo, khai thác dữ liệu HTS và ADMET in silico cũng được trình bày.

Hai tác giả G. M. Downs, J. M. Barnard đến từ công ty Barnard Chemical Information Ltd, thuộc nước Anh đã công bố công trình "*Clustering Methods and Their Uses in Computational Chemistry*" (2003) **[3]**. Bài trình bày này góp phần tăng cường sự quan tâm của các nhà hoá học vào sử dụng các ứng dụng của công nghệ thông tin trong nghiên cứu hoá học của các phân tử hữu cơ, họ đưa ra giải pháp dùng phân cụm để phân tích hoá học, bài báo nêu lên những mặt mạnh của việc phân cụm ứng dụng trong phân tích hoá học.

Công trình "*From Big Data to Artificial Intelligence: chemoinformatics meets new challenges*" của I.V. Tetko và O. Engkvist (2020) **[4]**, đề cập đến những thách thức và cơ hội liên quan đến dữ liệu lớn trong Hóa tin. Bài viết này trình bày việc tích hợp dữ liệu hóa học và sinh học quy mô lớn, cũng như việc sử dụng các phương pháp phân tích dữ liệu lớn và học máy để khai thác và trích xuất những hiểu biết có giá trị. Bài viết cũng nhấn mạnh tác động của trí tuệ nhân tạo và học máy, đặc biệt là mạng lưới thần kinh, ngày càng được sử dụng nhiều hơn trong ngành công nghiệp hóa chất.

Trong việc hiển thị, công cụ RDKit đã khá chuẩn mực, tuy nhiên việc trực quan hoá liên kết hoá học trong một hợp chất dưới dạng 3D cũng là một thách thức. Bài viết "*Visualizing chemical space networks with RDKit and NetworkX*" **[5]** của các tác giả V.F. Scalfani, V.D. Patel và A.M. Fernandez đăng trên Journal of Cheminformatics vào năm 2022 trình bày cách tạo Mạng không gian hóa học (Chemical Space Networks - CSN) bằng cách sử dụng workflow của RDKit và NetworkX. Qua đó trực quan hóa mô tả các hợp chất như các nút được kết nối bởi các cạnh tương tự giá trị của dấu vân tay 2D. Để biểu diễn cấu trúc hoá học thành một chuỗi ký tự đã có quy ước SMILES. Tuy nhiên, quy ước này cũng chưa được nhất quán vì một hợp chất có thể viết dưới dạng khác nhau với cùng quy ước. Công trình "*Improving the quality of chemical language model outcomes with atom-in-SMILES tokenization*" của các tác giả U.V. Ucak, I. Ashyrmamatov và J. Lee cũng đã tìm cách cách tiến để hợp lý hơn. Công trình này đăng trên Journal of Cheminformatics vào 2023 **[6]**. Mã SMILES là một bước tiền xử lý quan trọng trong xử lý ngôn ngữ tự nhiên có thể có ảnh hưởng đáng kể đến chất lượng dự đoán. Nghiên cứu này cho thấy

mã thông báo SMILES truyền thống có một hạn chế nhất định dẫn đến việc mã thông báo không phản ánh bản chất thực sự của các phân tử. Để giải quyết vấn đề này, công trình này đã phát triển sơ đồ mã hóa atom-in-SMILES giúp loại bỏ sự mơ hồ trong bản chất chung của mã thông báo SMILES.

Trong việc khám phá thuốc bằng các phương pháp học máy, có Công trình "*Virtual Screening Algorithms in Drug Discovery: A Review Focused on Machine and Deep Learning Methods*" của tác giả Tiago Alves de Oliveira, Michel Pires da Silva và cộng sự (2023) **[7]** đã liệt kê hầu hết những thuật toán trong học máy được dùng trong việc sàng lọc ảo để hỗ trợ điều chế thuốc. Để xử lý dữ liệu nhằm hạn chế bớt việc học qúa dư thừa (overfitting) trong quá trình huấn luyện có công trình "*Large-scale evaluation of k-fold cross-validation ensembles for uncertainty estimation*" **[8]** đăng trên Journal of Cheminformatics năm 2023. Nhóm tác giả tập trung để phối hợp kỹ thuật kiểm tra chéo để tăng sự hiệu quả. Công trình "*Predicting the reproductive toxicity of chemicals using ensemble learning methods and molecular fingerprints*" **[9]** tập trung vào độc tính sinh sản, một vấn đề quan trọng về an toàn trong việc đánh giá tác động tiêu cực của các chất hoá học trong khám phá thuốc. Các mô hình tính toán có khả năng dự đoán chính xác tiềm năng độc hại của một chất hoá học. Công trình này sử dụng mô hình học phối hợp (Ensemble Learning) đã được xây dựng để dự đoán độc tính sinh sản của các hợp chất, bên cạnh đó còn sử dụng thuật toán SVM, Random Forest và Gradient Boosting.

C. NHỮNG THÁCH THỨC VÀ CƠ HỘI CỦA HỌC MÁY TRONG HÓA TIN

Cũng như mọi ngành, khi sử dụng những thành tựu của máy tính và công nghệ thông tin hầu như đều có được những thuận lợi để giải quyết, tuy nhiên cũng không phải không có những thách thức đặt ra.

1. THÁCH THỨC

Có một số thách thức bao gồm:

- Tính sẵn có của dữ liệu (Data availability): Một trong những thách thức lớn nhất trong học máy là tính chất sẵn có của dữ liệu. Trong Hóa tin, thường thiếu dữ liệu được gắn nhãn, dữ liệu này cần thiết để huấn luyện các thuật toán ML.
- Chất lượng dữ liệu (Data quality): Ngay cả khi có sẵn dữ liệu được dán nhãn, điều quan trọng là phải đảm bảo rằng dữ liệu có chất lượng cao. Chất lượng dữ liệu có thể bị ảnh hưởng bởi một số yếu tố, chẳng hạn như lỗi thử nghiệm và dữ liệu không đầy đủ. Ngoài ra, có một số dữ liệu việc mất cân bằng thường xãy ra, vì trong tự nhiên việc có đặc tính nào đó thường ít hơn rất nhiều so với việc không có.
- Khả năng diễn giải mô hình (Model interpretability): Một trong những thách thức khi sử dụng các mô hình học máy là chúng có thể khó diễn giải. Điều này có thể gây khó khăn cho việc hiểu mô hình hoạt động như thế nào và khó đưa ra dự đoán dựa trên mô hình. Ngoài ra, việc lựa chọn mô hình phù hợp với bài toán cũng là vấn đề cần đặt ra.

2. CƠ HỘI

Bất chấp những thách thức này, học máy có tiềm năng như một sự cách mạng hóa những vấn đề của hóa học. Chẳng hạn,

- Tăng độ chính xác: Các thuật toán học máy thường có kết quả tính toán vượt trội so với các phương pháp truyền thống đối với nhiều bài toán khác nhau, chẳng hạn như khám phá thuốc và dự đoán độc tính.
- Giảm thời gian và chi phí: Khi áp dụng thuật toán học máy có thể giúp giảm thời gian và chi phí cho việc giải bài toán khám phá thuốc và các bài toán hóa học phức tạp khác.
- Thông tin chi tiết mới (New insights): Học máy có thể giúp cung cấp thông tin chi tiết mới về các quy trình và hệ thống hóa học.

III. PHƯƠNG PHÁP GIẢI QUYẾT VẤN ĐỀ

A. KHAI THÁC TẬP TIN TRONG DATASET

1. THƯ VIỆN RDKIT

Để xem nội dung các tập tin trong phần nguồn dữ liệu (II.A), có thể sử dụng thư viện rdkit-pypi (https://pypi.org/project/rdkit-pypi/). Đây là thư viện để dùng trong một chương trình được viết bằng ngôn ngữ Python. Đây là một bộ công cụ mã nguồn mở dùng trong Hoá tin, hóa học dược phẩm (Drug Discovery), hoá học các hợp chất tự nhiên và khoa học vật liệu. Thư viện rdkit-pypi là một bộ phận trong RDKit. Bộ công cụ này được phát triển bởi một nhóm nghiên cứu hóa học tính toán tại Scripps Research Institute ở La Jolla, California. Nhóm dưới sự lãnh đạo của GS. David Weininger. RDKit cung cấp một tập hợp các công cụ và chức năng để xử lý và phân tích dữ liệu hóa học như tạo và xử lý cấu trúc hóa học, tính toán các thuộc tính hóa học, đồ thị hóa học, phân tích phân tử, v.v… Hiện nay, RDKit hiện được duy trì bởi một nhóm các nhà phát triển từ khắp nơi trên thế giới. Dự án được tài trợ bởi nhiều nguồn khác nhau, bao gồm Viện Y tế Quốc gia Hoa Kỳ (National Institutes of Health), Quỹ Khoa học Quốc gia Hoa Kỳ (National Science Foundation) và Liên minh Châu Âu (European Union).

Một số đặc điểm và tính năng của RDKit:

- Tạo và xử lý cấu trúc hóa học: RDKit hỗ trợ tạo và xử lý các cấu trúc hóa học dưới dạng SMILES, InChI và các định dạng hóa học khác. Đồng thời cho phép người dùng viết và xử lý các dữ liệu hóa học từ các nguồn khác nhau.
- Tính toán thuộc tính hóa học: RDKit cung cấp các công cụ để tính toán và phân tích các thuộc tính hóa học như khối lượng phân tử, độ phân cực, độ kiềm, độ acid, và nhiều thuộc tính khác. Điều này giúp phân tích và đánh giá tính chất hóa học của các phân tử.
- Xử lý đồ thị hóa học: RDKit cung cấp các công cụ để xây dựng, xử lý và phân tích đồ thị hóa học, bao gồm phân tích cấu trúc liên kết, tìm kiếm mô hình phân tử, tìm kiếm đồ thị con và nhiều tính năng khác.
- Tích hợp với các công cụ và thư viện khác: RDKit có khả năng tích hợp với các thư viện và công cụ phổ biến khác trong lĩnh vực hóa học và học máy như numpy, pandas, scikit-learn, TensorFlow. Điều này giúp mở rộng khả năng của RDKit và kết hợp nó với các công cụ khác để thực hiện các nhiệm vụ phức tạp.

RDKit hỗ trợ nhiều ngôn ngữ lập trình, như Python và C++, nhưng trong môi trường Python, RDKit cung cấp cú pháp đơn giản và dễ sử dụng, giúp các nhà phát triển cũng như nghiên cứu dễ dàng tạo ra sản phẩm cho riêng mình.

2. CẤU TRÚC TẬP TIN SDF

Tập tin SDF (Structure-Data File) là một định dạng tập tin thông dụng trong Hoá tin, nhằm để lưu trữ thông tin về các cấu trúc hóa học và dữ liệu liên quan. Tập tin này được sử dụng để lưu trữ và chia sẻ thông tin về các phân tử, bao gồm cả thông tin cấu trúc và thuộc tính hóa học. Cấu trúc của tập tin SDF được tổ chức dưới dạng các mục (entries), mỗi mục biểu diễn một phân tử. Mỗi mục bao gồm hai phần chính: phần khối (block) và phần thuộc tính (property). Trong đó:

- Phần khối chứa thông tin về cấu trúc hóa học của phân tử với các trường như tên, SMILES, InChI, đồ thị hóa học (graph), toạ độ nguyên tử, liên kết hóa học, và các thông tin khác về cấu trúc hóa học của phân tử.
- Phần thuộc tính chứa thông tin về các thuộc tính hóa học khác của phân tử. Các thuộc tính này có thể bao gồm khối lượng phân tử, độ phân cực, độ kiềm, độ acid, hoạt tính sinh học, và các thuộc tính khác.

Mỗi mục trong tập tin SDF được phân tách bằng dấu "##$$" hoặc dấu "$$$$" để đánh dấu sự kết thúc của một mục và sự bắt đầu của mục mới. Một ví dụ cấu trúc tập tin SDF như Hình 1:

```
Molecule 1
  SMILES: C1=CC=CC=C1
  MW: 78.11
  LogP: 1.2
  ...
$$$$
Molecule 2
  SMILES: CC(=O)OC
  MW: 88.15
  LogP: -0.5
  ...
$$$$
...
```

Hình 1. *Cấu trúc tập tin SDF*

Trong tập tin trên, có hai mục, mỗi mục biểu diễn một phân tử. Trong từng mục có các trường như SMILES, khối lượng phân tử (MW), LogP và các thuộc tính khác.

Một tập tin SDF có thể chứa nhiều mục phân tử, cho phép lưu trữ và truy xuất thông tin về nhiều phân tử trong một tập tin duy nhất.

Chẳng hạn, từ thư viện ChEBI có thể download tập tin https://ftp.ebi.ac.uk/pub/databases/chebi/SDF/ChEBI_lite_3star.sdf.gz; sau đó hiển thị một vài mục đầu tiên ra màn hình. Chương trình bằng Python được viết tổng thể như sau.

```
########################################
# Print Molecules from SDF File        #
# @author: A.Prof. Tran Van Lang, PhD #
# File: displaySDF.py                  #
########################################

from rdkit import Chem

sdf_file = 'data/ChEBI_lite_3star.sdf'
```

```
# Đọc dữ liệu từ file SDF và lưu trữ các phân tử vào danh sách molecules
listmolecules = []
with open(sdf_file, 'rb') as file:
    supplier = Chem.ForwardSDMolSupplier(file)
    for molecule in supplier:
        if molecule is not None:
            listmolecules.append(molecule)
            if len(listmolecules) == 3:
                break

# Xuất thông tin các phân tử ra màn hình
index = 1
for molecule in listmolecules:
    print( '\nMolecule #%.d' %index )
    index += 1
    molblock = Chem.MolToMolBlock(molecule)

    # Tạo đối tượng Mol từ MolBlock
    mol = Chem.MolFromMolBlock(molblock)
    print('- SMILES:',Chem.MolToSmiles(molecule))

    # Kiểm tra xem thuộc tính 'NAME' có tồn tại trong phân tử hay không
    name = mol.GetProp('NAME') if 'NAME' in mol.GetPropsAsDict() else 'N/A'
    print( '- Name:', name )
    print( '- Number of atoms:', mol.GetNumAtoms())
    print( '- Number of bonds:', mol.GetNumBonds())
    print( '- Atoms: ',end='')
    for atom in mol.GetAtoms():
        print(atom.GetSymbol(), end=',')
    print( '\n- Bonds: ',end='')
    for bond in mol.GetBonds():
        print(bond.GetBeginAtomIdx(), bond.GetEndAtomIdx(), bond.GetBondType(),end=';')
    print( '\n----------------------')
```

Kết quả hiển thị như Hình 2.

```
Molecule #1
- SMILES: Oc1cc(O)c2c(c1)O[C@H](c1ccc(O)c(O)c1)[C@H](O)C2
- Name: N/A
- Number of atoms: 21
- Number of bonds: 23
- Atoms: C,C,C,C,C,C,C,C,O,C,C,C,C,C,C,O,O,O,O,O,
- Bonds: 3 0 AROMATIC;0 20 SINGLE;0 1 AROMATIC;1 2 AROMATIC;2 19 SINGLE;2 5 AROMATIC;4 3 AROMATIC;4 9 SINGLE;4 5 AR
OMATIC;5 6 SINGLE;6 7 SINGLE;7 18 SINGLE;7 8 SINGLE;8 9 SINGLE;8 10 SINGLE;15 10 AROMATIC;10 11 AROMATIC;11 12 AROM
ATIC;12 16 SINGLE;12 13 AROMATIC;13 17 SINGLE;13 14 AROMATIC;14 15 AROMATIC;
----------------------

Molecule #2
- SMILES: CC1(C)C(=O)[C@@]2(C)CC[C@@H]1C2
- Name: N/A
- Number of atoms: 11
- Number of bonds: 12
- Atoms: C,C,C,C,C,C,C,C,C,O,
- Bonds: 4 9 SINGLE;4 2 SINGLE;9 3 SINGLE;3 0 SINGLE;0 1 SINGLE;1 2 SINGLE;4 5 SINGLE;0 5 SINGLE;4 6 SINGLE;3 7 SIN
GLE;3 8 SINGLE;9 10 DOUBLE;
----------------------

Molecule #3
- SMILES: *C(=O)OC(CO)CO[1*]
- Name: N/A
- Number of atoms: 10
- Number of bonds: 9
- Atoms: R1,O,C,O,C,C,O,C,O,R,
- Bonds: 0 1 SINGLE;3 2 SINGLE;4 2 SINGLE;4 6 SINGLE;4 5 SINGLE;1 5 SINGLE;6 7 SINGLE;7 8 DOUBLE;7 9 SINGLE;
----------------------
```

Hình 2. *Hiển thị 3 phân tử đầu tiên có trong file ChEBI_lite_3star.sdf*

3. CHUỖI SMILES

Chuỗi SMILES (Simplified Molecular Input Line Entry System) như đã đề cập ở trên, đó là một định dạng chuỗi được sử dụng để biểu diễn cấu trúc hóa học của các phân tử. Đây một phương tiện đơn giản và nhất quán để biểu diễn cấu trúc hóa học của phân tử. SMILES đã được sử dụng trong nhiều ứng dụng, bao gồm lưu trữ dữ liệu hóa học, tìm kiếm phân tử, xây dựng và mô phỏng phân tử, và phân tích hóa học dựa trên cấu trúc.

Cấu trúc hóa học của một phân tử được biểu diễn bằng cách sử dụng các ký hiệu và quy tắc đơn giản. Các nguyên tố hóa học được biểu diễn bằng các ký hiệu viết tắt, ví dụ: C (carbon), H (hydrogen), O (oxygen), N (nitrogen), và các nguyên tố khác. Các liên kết hóa học giữa các nguyên tử được biểu diễn bằng các ký hiệu gồm "-" (liên kết đơn), "=" (liên kết đôi), "#" (liên kết ba), và "." (không có liên kết).

SMILES cũng có thể biểu diễn các nhóm chức năng, vòng và cấu trúc phức tạp hơn của phân tử. Để biểu diễn nhóm chức năng, các ký hiệu như "()" và "[]" được sử dụng. Chẳng hạn, nhóm amino (-NH2) có thể được biểu diễn bằng cách sử dụng "N" và "H" trong ngoặc đơn "(NH2)". Các vòng có thể được biểu diễn bằng cách sử dụng số chỉ mục cho các nguyên tử trong vòng. Chẳng hạn, một vòng benzene (Hình 3) có thể được biểu diễn bằng "C1=CC=CC=C1".

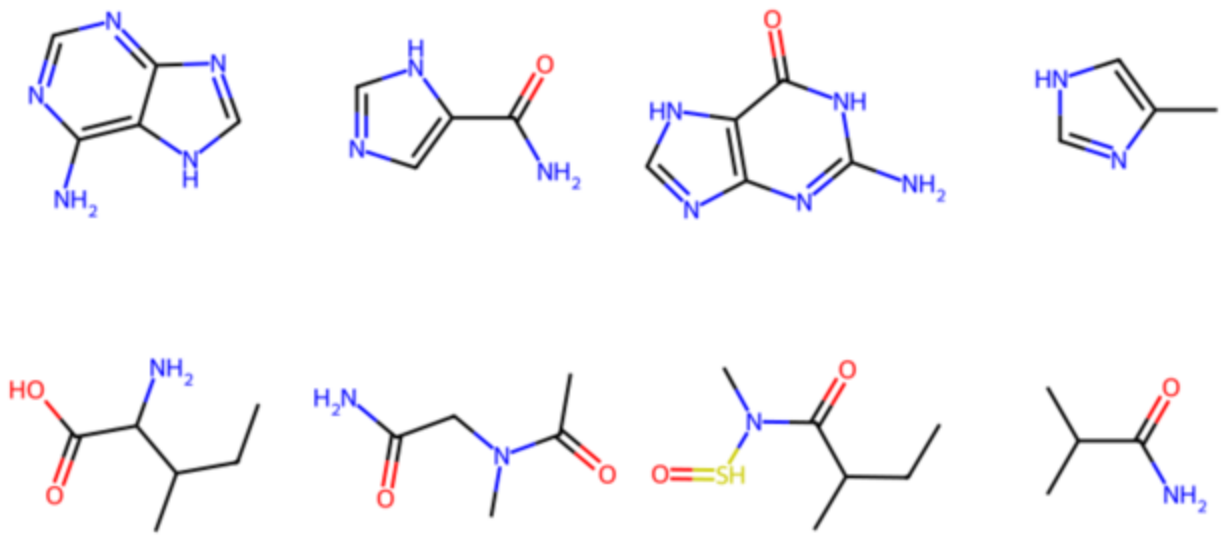
Hình 3. *Vòng Benzene*

Để hiển thị một chuỗi SMILES, ta có thể viết chương trình như sau dùng thư viện RDKit:

```
################################################
# Display Chemical Structure from SMILES String #
# @author: A.Prof. Tran Van Lang, PhD          #
# File: displaySMILES.py                       #
################################################

from rdkit import Chem
from rdkit.Chem import Draw

smiles =[
   'C1=NC2=NC=NC(=C2N1)N',      ### Adenine
   'C1=C(NC=N1)C(=O)N',         ### Cytosine
   'C1=NC2=C(N1)C(=O)NC(=N2)N', ### Guanine
   'CC1=CNC=N1',                ### Thymine
   'CCC(C)C(N)C(=O)O',          ### Isoleucine
   'CC(=O)N(C)CC(=O)N',         ### Lysine
   'CCC(C)C(=O)N(C)S(=O)',      ### Methionine:
   'C[C@@H](C)C(=O)N'           ### Valine
      ]

# Chuyển mảng các chuỗi SMILES thành mảng các cấu trúc hoá học
mols = [Chem.MolFromSmiles(smile) for smile in smiles]

# Xem cấu trúc hoá học của Adenine
Draw.MolToImage(mols[0])
```

Trong trường hợp muốn xuất tất các các cấu trúc hoá với mỗi dòng có 4 cấu trúc như Hình 4, dùng câu lệnh sau để hiển thị nhiều cấu trúc hóa học trên cùng một hình ảnh.

```
img = Draw.MolsToGridImage(mols, molsPerRow=4)
display(img)
```



Hình 4. *Biểu diễn cấu trúc hoá học của 4 nucleotide và 4 trong số 9 amino acid thiết yếu*

Ngoài ra, cũng có thể xuất các cấu trúc này thành những tập tin hình ảnh dạng PNG riêng bằng lệnh:

```
Draw.MolToImage(mols[2]).show()
```

4. CHUỖI INCH

Chuỗi InChI (International Chemical Identifier) là một chuỗi ký tự tiêu chuẩn được sử dụng để mô tả đặc điểm cấu trúc và hóa học của một phân tử. Qua đó cung cấp phương pháp độc lập ngôn ngữ để đại diện cho một phân tử và cho phép truyền tải thông tin về cấu trúc hoá học một cách chính xác và duy nhất. Tính duy nhất của chuỗi InChI là một đặc điểm quan trọng, tức là cùng một cấu trúc hóa học chỉ có một InChI, đảm bảo tính nhất quán và phù hợp trong việc truyền tải thông tin hóa học giữa các nguồn dữ liệu và ứng dụng khác nhau.

Chuỗi InChI được chia thành các phần (layers) khác nhau, mỗi phần đại diện cho một khía cạnh cụ thể của cấu trúc hoá học. Cấu trúc cơ bản của chuỗi InChI bao gồm các phần sau:

- Layer 1: Định danh hóa học (chemical formula), đại diện cho công thức hóa học của phân tử, bao gồm các nguyên tố và số lượng nguyên tử tương ứng.
- Layer 2: Kết hợp (connectivity), đại diện cho kết nối giữa các nguyên tử trong phân tử. Sử dụng các chỉ số số nguyên để chỉ ra quan hệ kết nối giữa các nguyên tử.
- Layer 3: Chức năng hóa học (chemical functionality), đại diện cho các tính chất chức năng của phân tử, bao gồm các nhóm chức năng và các khối chức năng (functional groups).
- Layer 4: Stereochemistry (không gian hình học), đại diện cho thông tin về cấu trúc không gian của phân tử, bao gồm các loại stereochemistry như cis/trans, R/S, E/Z.
- Layer 5: Thông tin về isotop (isotopic information), đại diện cho thông tin về các đồng vị của các nguyên tố có trong phân tử.
- Layer 6: Thông tin về protonation state (tautomerization and protonation), đại diện cho thông tin về tautomeric và trạng thái protonation của phân tử.

Mỗi phần được phân tách bằng dấu gạch chéo ("/") để tạo thành một chuỗi InChI hoàn chỉnh. Cấu trúc này cho phép mô tả chi tiết về cấu trúc và tính chất hóa học của một phân tử một cách chính xác và duy nhất.

Chẳng hạn chuỗi InChI đại diện cho phân tử ethanoic acid (acid acetic):

```
InChI=1S/C2H4O2/c1-2(3)4/h1H3,(H,3,4)
```

Trong đó,

- Layer 1: C2H4O2 đại diện cho công thức hóa học của phân tử ethanoic acid.
- Layer 2: Không có thông tin kết nối được đưa ra vì phân tử chỉ có 2 nguyên tử C và 4 nguyên tử H được nối với nhau một cách đơn giản.
- Layer 3: Đại diện cho chức năng hóa học của phân tử, trong trường hợp này là một nhóm carbocylic acid (COOH).

- Layer 4: Không có thông tin stereochemistry được đưa ra vì ethanoic acid không chứa các trạng thái không gian quan trọng.
- Layer 5: Không có thông tin về isotop vì ethanoic acid không chứa các đồng vị.
- Layer 6: Không có thông tin về protonation state vì ethanoic acid không tồn tại trong các trạng thái protonation khác nhau.

Chuỗi InChI này mô tả đầy đủ các khía cạnh cấu trúc và tính chất hóa học của ethanoic acid, giúp đảm bảo tính duy nhất và chính xác của mô tả hóa học.

Chương trình hiển thị chuỗi InChI bằng cách dùng hàm MolFromInchi(), thay cho hàm MolFromSmiles() khi đọc chuỗi SMILES

```
from rdkit import Chem
from rdkit.Chem import Draw

# Chuỗi InChI của phân tử ethanoic acid
inchi = "InChI=1S/C2H4O2/c1-2(3)4/h1H3,(H,3,4)"

# Tạo đối tượng Molecule từ chuỗi InChI
mol = Chem.MolFromInchi(inchi)

# Vẽ cấu trúc hoá học
Draw.MolToImage(mol)
```

Chuỗi InChI của 4 nucleotide trong trình tự DNA:

Adenine (A) : InChI=1S/C5H5N5/c6-4-3-5(9-1-7-3)10-2-8-4/h1-2H,(H3,6,7,8,9,10)
Cytosine (C): InChI=1S/C4H5N3O/c5-3-1-2-6-4(8)7-3/h1-2H,(H3,5,6,7,8)
Guanine (G) : InChI=1S/C5H5N5O/c6-5-3-1-2-7-4(3)8-9-5/h1-2H,(H3,6,7,8,9)
Thymine (T) : InChI=1S/C5H6N2O2/c8-5-3-7-2-1-6-4(5)9/h1-3H,(H2,6,8,9)

Lưu ý rằng chuỗi InChI này chỉ biểu diễn cấu trúc hóa học của từng nucleotide và không bao gồm thông tin về vị trí liên kết trong chuỗi nucleotide.

Chúng ta có thể tạo chuỗi InChI từ cấu trúc hoá học, chẳng hạn với cấu trúc trong chuỗi SMILES được viết như sau với thư viện RDKit:

```
from rdkit import Chem

smiles = 'C1=NC2=NC=NC(=C2N1)N'      ### Adenine
mol = Chem.MolFromSmiles(smiles)
inchi = Chem.MolToInchi(mol)
print(inchi)
```

B. BÀI TOÁN XÁC ĐỊNH HOẠT TÍNH HỢP CHẤT HOÁ HỌC

1. ĐẶT VẤN ĐỀ

Trong tập dữ liệu về các thử nghiệm sinh học (dùng trong sàng lọc - screens), việc đo lường hoạt tính (activity) của các hợp chất khác nhau đối với các mục tiêu sinh học khác nhau thường có sự mất cân bằng lớn giữa loại có hoạt tính (Active) và không có hoạt tính (Inactive), trong đó số lượng dữ liệu không có hoạt tính lớn hơn nhiều. Chính vì vậy, việc huấn luyện cũng phải sử dụng mô hình học máy phù hợp. Một số công việc sau đây cần xử lý trước khi dùng phương pháp học máy để huấn luyện:

- Khảo sát sự phụ thuộc của các thuộc tính hay đặc trưng (feature) trong tập dữ liệu, có thể sử dụng các phương pháp như ANOVA F-test để đánh giá sự phụ thuộc của mỗi đặc trưng đối với biến mục tiêu, hoặc có thể sử dụng hệ số tương quan. Qua đó có thể rút gọn thuộc tính.
- Tinh chỉnh dữ liệu để tránh sự chênh lệch các quan sát (data point) trong cùng một đặc trưng.
- Xử lý mất cân bằng dữ liệu bằng kỹ thuật lấy mẫu lại (resampling) như tăng cường mẫu (oversampling) hay giảm số lượng mẫu (undersampling).
- Đồng thời cũng phải kết hợp thêm kỹ thuật tìm kiếm lưới (Grid-Search) phối hợp kiểm tra chéo (Cross-Validates) để tìm các tham số dạng sensitive như learning rate, epoch, batch và cả các tham số dạng insensitive.

Để giải quyết vấn đề này, chúng ta có thể dùng thư viện Scikit Learn (https://scikit-learn.org/stable/)

2. RÚT GỌN THUỘC TÍNH

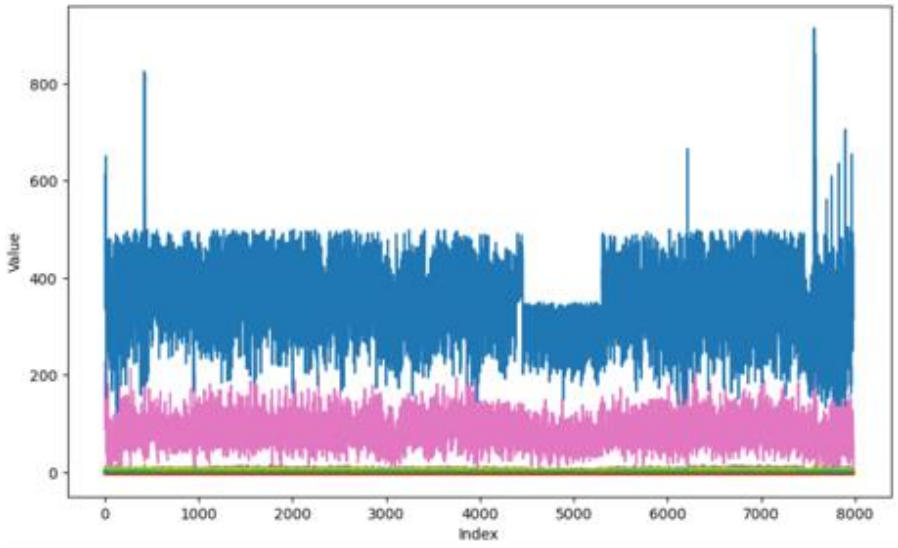
a) Xét sự ảnh hưởng lên biến mục tiêu

Phương pháp phân tích biến số ANOVA (Analysis of Variance) là một phương pháp thống kê được sử dụng để đánh giá mức độ phụ thuộc của các đặc trưng (features) đối với biến mục tiêu (target variable). Mục đích để kiểm tra xem có sự khác biệt đáng kể giữa các nhóm được tạo ra bởi các đặc trưng khác nhau hay không. Cách thức thực hiện bằng cách tính toán sự biến động giữa các nhóm và sự biến động trong các nhóm, và từ đó đưa ra giá trị F-statistic (hat F-test). Giá trị này được so sánh với một ngưỡng (chẳng hạn 0.05) để xác định xem có sự khác biệt đáng kể giữa các nhóm hay không.

Trong tập dữ liệu, có thể sử dụng các phương pháp như ANOVA F-test để đánh giá sự phụ thuộc của mỗi đặc trưng đối với biến mục tiêu, qua đó có thể rút gọn thuộc tính.

Sau khi trực quan hoá dữ liệu để quan sát như Hình 5. Từ dữ liệu đọc được từ tập tin csv tương ứng.

```
df_train = pd.read_csv('data/BioassayDatasets/AID456red_train.csv')
df_test  = pd.read_csv('data/BioassayDatasets/AID456red_test.csv')
```



Hình 5. *Trực quan dữ liệu huấn luyện*

Dữ liệu được chuyển từ các DataFrame sang mảng để xử lý.

```
X_tr = df_train.drop('Outcome', axis=1).values
X_te = df_test.drop('Outcome', axis=1).values

y_tr = df_train['Outcome'].values
y_te = df_test['Outcome'].values
```

Trước khi rút gọn thuộc tính, cũng cần xoá các quan sát bị thiếu bằng cách xoá các hàng có giá trị bị khuyết (missing values).

```
df_train.dropna(inplace=True)
```

Thông qua lớp SelectBest của scikit-learn để rút gọn, gỉa sử chỉ chọn 10 đặc trưng độc lập nhất thông qua chỉ số *k*, sau đó chỉ lấy các đặc trưng này:

```
selector = SelectKBest(score_func=f_classif, k=10)
X_train_se = selector.fit_transform(X_tr, y_tr)
X_test_se = selector.transform(X_te)

selected_features = selector.get_support()
print('Số đặc trưng được chọn:',df_train.columns[:-1][selected_features] )
```

Kết quả có 10 đặc trưng sau có độ độc lập cao.

```
Số đặc trưng được chọn: Index(['NEG_04_NEG', 'NEG_02_ARC', 'NEG_05_ARC', 'NEG_07_ARC', 'POS_05_POS',
       'POS_06_ARC', 'ARC_03_ARC', 'HYP_02_HYP', 'WBN_GC_L_0.25', 'WBN_GC_H_0.25'], dtype='object')
```

b) Khảo sát sự phụ thuộc giữa các đặc trưng

Có thể sử dụng hệ số tương quan (Correlation Coefficient) để rút gọn thuộc tính bằng phương pháp chọn lọc đặc trưng. Phương pháp này giúp lựa chọn các thuộc tính quan trọng nhất dựa trên hệ số tương quan của chúng với biến mục tiêu. Khi hệ số tương quan là +1 hay -1 cho biết có mối tương quan cao. Để thực hiện rút gọn thuộc tính bằng hệ số tương quan, các bước như sau:

- Tính ma trận tương quan: Sử dụng hàm corr() trong Pandas để tính ma trận tương quan của các thuộc tính trong một DataFrame.

```
corr_matrix = df_train.drop('Outcome', axis=1).corr()
```

- Định nghĩa một ngưỡng (threshold) để quyết định xem thuộc tính có mức độ tương quan đủ cao để giữ lại hay không. Giá trị ngưỡng này phụ thuộc vào bài toán cụ thể và mức độ quan trọng của các thuộc tính, thông thường là 0,75 khi xét trị tuyệt đối của các hệ số tương quan.
- Lựa chọn thuộc tính: Dựa trên ma trận tương quan và ngưỡng đã chọn, để đưa ra các thuộc tính có hệ số tương quan vượt qua ngưỡng, từ đó giữ lại thuộc tính cần thiết.

```
threshold = 0.75
corr_features = set()

for i in range(len(corr_matrix.columns)):
    for j in range(i):
        if abs(corr_matrix.iloc[i, j]) > threshold:
            colname = corr_matrix.columns[i]
            corr_features.add(colname)

selected_features = set(df_train.columns) - corr_features
```

Việc rút gọn thuộc tính dựa trên hệ số tương quan có thể là một phương pháp đơn giản. Tuy nhiên, cần xem xét kỹ lưỡng và thử nghiệm để đảm bảo rằng các thuộc tính được giữ lại vẫn giữ được thông tin quan trọng trong dữ liệu. Đây là một thách thức đặt ra khi xử lý bài toàn trong lĩnh vực xử lý số liệu liên quan đến ngành Cheminformatics.

Sau khi có số đặc trưng, sử dụng X_train và X_test để huấn luyện và kiểm chứng mô hình như là bước đầu tiên để có nhận xét sơ bộ về mô hình sử dụng cho việc huấn luyện.

```
X_train, X_test = X_train_se, X_test_se
```

3. TINH CHỈNH DỮ LIỆU

Khi giá trị dữ liệu trong một cột (một đặc trưng) chênh lên quá nhiều, có thể gây ảnh hưởng đến quá trình huấn luyện mô hình. Đây là một vấn đề phổ biến trong xử lý dữ liệu và có một số phương pháp để xử lý như sau:

- Min-max scaling: Sử dụng phép tỷ lệ tuyến tính để chuyển đổi giá trị dữ liệu về khoảng giá trị mong muốn, thường là từ 0 đến 1. Công thức tính toán cho Min-max scaling như sau:

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

- Standardization: Chuẩn hóa dữ liệu theo giá trị trung bình và độ lệch chuẩn (mean và standard deviation) để chuyển đổi dữ liệu về phân phối chuẩn (mean=0, std=1). Công thức tính toán cho Standardization như sau:

$$X_{scaled} = \frac{X - X_{mean}}{X_{std}}$$

- Log transformation: Áp dụng biến đổi logarithm cho dữ liệu để làm giảm độ chênh lệch giữa các giá trị. Điều này có thể giúp dữ liệu trở nên phân phối đều hơn và tạo điều kiện tốt hơn cho quá trình huấn luyện.
- Robust scaling: Sử dụng phép tỷ lệ theo giá trị trung vị và phạm vi tương đối để xử lý giá trị dữ liệu chênh lên. Đây là một phương pháp phù hợp khi dữ liệu chứa nhiễu hoặc giá trị ngoại lệ (outliers).

Với dữ liệu này, chúng ta có thể sử dụng phương pháp Min-max scaling để tinh chỉnh dữ liệu, sau đó dùng dữ liệu X_train_mms và X_test_mms trong các lệnh sau để huấn luyện.

```
minmax_scaler = MinMaxScaler()
X_train_mms = minmax_scaler.fit_transform(X_train_se)
X_test_mms = minmax_scaler.fit_transform(X_test_se)
```

4. XỬ LÝ MẤT CÂN BẰNG DỮ LIỆU

Để giải quyết vấn đề mất cân bằng lớn trong tập dữ liệu, có thể sử dụng các thư viện Python như imbalanced-learn. Đây là một thư viện Python mạnh mẽ cho xử lý mất cân bằng lớn.

Để tăng cường mẫu, có thể sử dụng các phương pháp như SMOTE, ADASYN, ClusterCentroids. Trong đó SMOTE (Synthetic Minority Over-sampling Technique) là một phương pháp oversampling phổ biến, các mẫu tạo thêm cho lớp thiểu số bằng cách kết hợp các mẫu gần nhất với nhau. Còn ADASYN (Adaptive Synthetic Sampling) là một phương pháp oversampling dựa trên SMOTE; nhằm tạo ra các mẫu mới cho lớp thiểu số dựa trên mức độ mất cân bằng trong dữ liệu. Còn ClusterCentroids là tạo thêm các dữ liệu mới trong lớp đa số bằng cách lựa chọn các centroid của các cụm dữ liệu trong lớp đa số và sau đó lấy các điểm gần centroid nhất để tạo mẫu nhân. Quá trình này nhằm giảm số lượng mẫu trong lớp đa số mà vẫn giữ được đặc trưng và phân phối của dữ liệu ban đầu. Còn RandomUnderSampler là một phương pháp giảm số lượng mẫu đơn giản bằng cách loại bỏ ngẫu nhiên một số mẫu từ lớp đa số để làm cân bằng với lớp thiểu số.

Đặc biệt, thư viện imbalanced-learn là một thư viện Python cung cấp các kỹ thuật khác nhau để xử lý sự mất cân bằng giữa các lớp, bao gồm oversampling, undersampling và các phương pháp kết hợp. Thư viện này tập trung vào việc sửa đổi tập huấn luyện để đạt được phân phối cân bằng giữa các lớp.

Với SMOTE để tạo thêm dữ liệu huấn luyện

```
smote = SMOTE(random_state=42)
X_train_smote, y_train_smote = smote.fit_resample(X_train_mms, y_tr)
print("Số lượng mẫu sau khi resample:", len(X_train_smote))
```

ADASYN cũng để tăng cường thêm

```
adasyn = ADASYN()
X_train_adasyn, y_train_adasyn = smote.fit_resample(X_train_se, y_tr)
print("Số lượng mẫu sau khi resample:", len(X_train_smote))
```

Hoặc dùng Cluster Centroid

```
cc = ClusterCentroids()
X_train_cc, y_train_cc = smote.fit_resample(X_train_se, y_tr)
print("Số lượng mẫu sau khi resample:", len(X_train_smote))
```

Bên cạnh đó, kỹ thuật học phối hợp (emsemble learning) như Bagging, Boosting có thể được chọn để giải quyết vấn đề này. Ngoài ra còn có thư viện imbalanced-learn-ensemble là một phần mở rộng của imbalanced-learn, tập trung đặc biệt vào các phương pháp ensemble cho việc phân loại dữ liệu mất cân bằng. Bằng cách tận dụng sức mạnh của các thuật toán học phối hợp, như Random Forest, AdaBoost và EasyEnsemble **[10]**, để xử lý hiệu quả dữ liệu mất cân bằng. Ở đây qua sự khảo sát trong phần thực nghiệm, lớp EasyEnsemble cũng có trong gói thư viện imbalenced-learn được chọn lựa.

- Lớp EasyEnsemble được sử dụng để giải quyết vấn đề mất cân bằng dữ liệu trong bài toán phân loại bằng cách tạo ra nhiều tập con (subsets) của dữ liệu gốc theo cách tiếp cận kỹ thuật undersampling trên lớp đa số (majority class). Cụ thể hơn, EasyEnsemble chia dữ liệu thành các tập con nhỏ có số lượng mẫu tương đương với lớp thiểu số (minority class). Sau đó, mỗi tập con được sử dụng để huấn luyện một mô hình phân loại đơn lẻ, chẳng hạn như Decision Tree hoặc Random Forest. Cuối cùng, các mô hình đơn lẻ này được kết hợp lại để tạo thành một mô hình ensemble. Mục tiêu của EasyEnsemble là tạo ra các mô hình phân loại có khả năng xử lý mất cân bằng dữ liệu và tăng cường hiệu suất phân loại trên lớp thiểu số. Bằng cách tạo ra nhiều mô hình từ các tập con dữ liệu, EasyEnsemble giúp giảm hiện tượng học quá dư thừa, hay quá khớp (overfitting) trên dữ liệu thiểu số và cung cấp dự đoán chính xác trên cả hai lớp.
- EasyEnsemble là một lựa chọn phổ biến để xử lý mất cân bằng dữ liệu trong bài toán phân loại và có thể được áp dụng cho các tập dữ liệu có tỷ lệ mất cân bằng lớn.
- Cũng bình luận thêm, trong công trình **[10]**, các tác giả đã đề xuất thuật toán EasyEnsemble để khắc phục nhược điểm mất cất bằng dữ liệu. EasyEnsemble lấy mẫu từ nhiều phần nhỏ của lớp đa số, huấn luyện một bộ học sử dụng mỗi phần nhỏ và kết hợp kết quả từ các bộ học đó. Kết quả thực nghiệm cho thấy kết quả nhận được khá tốt so với các phương pháp khác vào thời điểm đó.

5. TÌM CÁC THAM SỐ TỐI ƯU CHO MÔ HÌNH HUẤN LUYỆN

Tìm các tham số tối ưu cho mô hình huấn luyện là một bước quan trọng trong quá trình xây dựng mô hình học máy. Bằng cách tìm kiếm và tối ưu các tham số, chúng ta có thể cải thiện hiệu suất của mô hình và đạt được kết quả tốt hơn. Có nhiều phương pháp để tìm các tham số tối ưu cho mô hình huấn luyện, một trong số đó là sử

dụng tìm kiếm lưới (Grid Search). Tìm kiếm lưới là một phương pháp thử tất cả các giá trị tham số có thể có trong một tập hợp đã định sẵn, và đánh giá hiệu suất của mô hình với từng giá trị tham số để tìm ra giá trị tối ưu.

Trong Python, thư viện scikit-learn cung cấp lớp GridSearchCV để thực hiện tìm kiếm lưới cho mô hình. Đầu tiên, cần xác định các tham số muốn tìm kiếm và các giá trị mà muốn thử qua. Sau đó, tạo một đối tượng GridSearchCV, truyền mô hình và các tham số cùng với giá trị thử vào đó. Tiếp theo, dùng phương thức fit() trên đối tượng GridSearchCV để bắt đầu quá trình tìm kiếm lưới.

Chẳng hạn, với phương pháp huấn luyện dùng EasyEnsemble() ở trên, vấn đề đặt ra là tìm các tham số (hyperparameter) tối ưu phương pháp này

```
param_grid = {
   'n_estimators': [90,100,120],
   'random_state': np.arange(0,25,10)
}
gs = GridSearchCV( EasyEnsembleClassifier(),param_grid,cv=5)
gs.fit(X_train, y_train)
```

Sau khi việc tìm kiếm kết thúc, để biết kết quả tốt nhất thông qua thuộc tính best_params_ của đối tượng gs này; cũng có thể truy cập mô hình tốt nhất đã được huấn luyện thông qua thuộc tính best_estimator_ mà không cần huấn luyện lại.

6. ĐÁNH GIÁ MÔ HÌNH

Trong bài toán tìm hợp chất có hoạt tính sinh học nào đó hay không, thì việc phân lớp để từ đó dự báo (forecast) thì nhãn có hoạt tính (Active) được xếp vào nhóm dương tính (Positive). Khi phân lớp, có thể dùng các thước đo như Accuracy, Precision, Recall, AUC, ... để đánh giá mô hình huấn luyện được tạo ra. Trong đó Accuracy nhằm để đánh giá trong số các dự đoán (là Active + Inactive) thì có bao nhiêu phần trăm là dự đoán đúng (True Active + True Inactive). Thước đo Accuracy thường được dùng khi đánh giá tổng thể, để có cái nhìn chung cả trường hợp positive (Active) và cả negative (Inactive). Hạn chế của Accuracy là đo lường trên tất cả các phân loại mà không quan tâm đến độ chính xác trên từng nhãn.

Còn khi chỉ quan tâm đến Active (positive) khi dự đoán, thì thước đo Precision hay Positive Predictive Value (PPV - trị số tiên lượng positive) để xác định trong tất cả các dự đoán là Acitive (cả true và false), thì có bao nhiêu phần dự đoán positive là đúng. Và thước đo Recall hay Sensitivity (độ nhạy) hay True Positive Rate (TPR - tỷ số positive thật) để xác định trong tất cả các trường hợp thực tế là Active, thì có bao nhiêu dự đoán Active là đúng. Để cân bằng cả 2 thước đo Precision và Recall có thước đo F1-Score. Ngoài ra còn có thước đo G-mean để đánh giá việc phân lớp cho bài toán mất cân bằng, ở đó G-mean là căn bậc hai của tích giữa độ nhạy và độ đặc hiệu (Specificity). Với độ đặc hiệu hay còn gọi là tỷ số âm tính thật (True Negative Rate - TNR) nhằm xác định trong tất cả các trường hợp thực tế Inactivate (cả true và false), thì có bao nhiêu phần dự đoán Inactivate là đúng.

Giá trị AUC (Area Under the ROC Curve) được sử dụng để đánh giá hiệu suất của mô hình phân lớp. AUC nằm trong khoảng từ 0 đến 1, và một giá trị AUC cao hơn thường được coi là tốt hơn.

- AUC = 0.5: là giá trị AUC ngẫu nhiên, có nghĩa là mô hình không có khả năng phân loại.
- 0.5 < AUC < 0.7: giá trị AUC thấp, mô hình có hiệu suất phân lớp yếu.
- 0.7 ≤ AUC < 0.8: giá trị AUC trung bình, mô hình có khả năng phân lớp vừa phải.
- 0.8 ≤ AUC < 0.9: giá trị AUC tốt, mô hình có khả năng phân lớp tốt.
- AUC ≥ 0.9: Đây là giá trị AUC rất tốt, mô hình có khả năng phân lớp rất cao.

Tuy nhiên, giá trị AUC cần được đánh giá kết hợp với bối cảnh và yêu cầu của bài toán cụ thể. Một mô hình có AUC cao không đồng nghĩa với việc nó hoàn hảo. Thông thường, việc so sánh AUC giữa các mô hình khác nhau hoặc theo tiêu chí ngưỡng chấp nhận được là một phương pháp hữu ích để đánh giá hiệu suất phân lớp. Đây cũng là một thách thức đặt ra khi giải quyết bài toán phân loại hoạt tính trong Cheminformatics.

Còn việc đánh giá học quá dư thừa chỉ hiện tượng mô hình đạt kết quả rất tốt trên tập huấn luyện nhưng lại kém trên tập dữ liệu kiểm định. Thực chất đây là việc học quá, tức là học cả thành phần nhiễu lẫn trong mẫu huấn luyện. Còn underfitting (chưa phù hợp hay học còn yếu, chưa đủ, chưa tới), chỉ hiện tượng mô hình huấn luyện được tạo ra có kết quả xấu trên cả tập huấn luyện và tập kiểm định. Việc khảo sát các tính chất này nhằm tìm ra giải pháp giúp mô hình tốt hơn.

IV. KẾT QUẢ THỬ NGHIỆM

A. DATASET THỬ NGHIỆM

Để thử nghiệm, dùng bộ dữ liệu xét nghiệm sinh học (bioassay) có trên Kaggle tại https://www.kaggle.com/datasets/uciml/bioassay-datasets. Đây là bộ dữ liệu của PubChem cũng đã được đưa lên kho lưu trữ dữ liệu học máy UC Irvine **[11]**. Việc khám phá thuốc (drug discovery) là giai đoạn đầu tiên trong quá trình phát triển thuốc (drug-development process); trong đó việc tìm kiếm các hợp chất để kiểm tra và sàng lọc đối với mục tiêu sinh học cụ thể. Giai đoạn đầu tiên này được gọi là sàng lọc ban đầu và thường bao gồm sàng lọc hàng ngàn hợp chất.

Tập dữ liệu này là một bộ sưu tập gồm 21 xét nghiệm sinh học (dạng sàng lọc-screens) đo lường hoạt tính (activity) của các hợp chất khác nhau đối với các mục tiêu sinh học khác nhau. Trong trình bày này, chúng tôi thử nghiệm trên tập mẫu AID456red_train.csv. Tập mẫu này chứa các xét nghiệm sàng lọc sơ bộ từ tổ chức Burnham Center for Chemical Genomics, nhằm ức chế biểu hiện bề mặt tế bào VCAM-1 do gen TNFa gây ra. Sau khi đã xử lý bằng cách bỏ các mẫu dữ liệu không đầy đủ trên tập dữ liệu huấn luyện, số lượng còn lại là 22 mẫu hợp chất có hoạt tính (Active) và 7.964 mẫu hợp chất không có hoạt tính. Tỷ lệ 1 hợp chất có hoạt tính so hợp chất không có hoạt tính là 1/362; một sự chênh lệch rất lớn khi phải phân lớp. Tương tự như vậy cho bộ dữ liệu kiểm tra có 1991 hợp chất không có hoạt tính và 5 hợp chất có hoạt tính.

Mẫu dữ liệu này có giá trị như sau:

- Các cột đầu tiên (từ cột 1 đến cột 95): giá trị các cột này đều là 0 và 1, đại diện cho các đặc trưng nhị phân.
- Các cột tiếp theo (từ cột 96 đến cột 153): giá trị các cột này là các số thực, chứa thông tin về các đặc trưng hoá - lý của hợp chất.
- Các cột cuối cùng (cột 154): đại diện cho trạng thái hoạt tính của mẫu, với "Active" và "Inactive" cho biết rằng mẫu có hoạt tính sinh học hay không. Cột này đóng vai trò là nhãn mục tiêu (target) cho biết trạng thái hoạt tính của mẫu.

B. THANG ĐIỂM ĐÁNH GIÁ

Với đặc thù của bài toán, thước đo được chọn theo thứ tự ưu tiên là G-mean để mong muốn đều dự đoán đúng cả Positive và Negative, tiếp theo là là Precision (để tránh dự đoán sai lớp Positive), sau đó là AUC cũng phải có giá trị cao để mô hình có khả năng phân lớp.

Khi đó, điểm đánh giá là:

$$Score = G - mean \times 3 + Precision \times 2 + AUC \times 1$$

C. KẾT QUẢ

Qua các thử nghiệm với các phương pháp tiếp cận khác nhau như rút gọn thuộc tính bằng hệ số tương quan (Correlation); xử lý dữ liệu bằng tỷ lệ Min-max (Minmax Scaling), Robuts Scaling, Standard Scaling; cân bằng dữ liệu bằng các phương pháp như SMOTE, ADASYN, Cluster Centroid; các phương pháp học như Mạng neural truyền thẳng (MLP), Balanced Bagging, Random Forest, Gradient Boosting, EasyEnsemble, RUS Boost **[12],[13]**. Cũng tương tự như EasyEnsemble, mặc dù RUS Boost ra đời cũng đã 15 năm nay, nhưng khi sử dụng cho bài toán phân lớp trong Hoá tin vẫn có những kết quả khích lệ. RUS Boost ra đời nhằm để giảm thiểu vấn đề mất cân bằng khi phân lớp. RUSBoost kết hợp việc lấy mẫu dữ liệu ngẫu nhiên (sampling) và tăng cường dữ liệu (boosting). Phương pháp này giảm bớt số lượng mẫu thuộc lớp đa số (majority class) để cân bằng với số lượng mẫu thuộc lớp thiểu số (minority class) nhằm giúp giảm sự chênh lệch giữa các lớp và tăng khả năng mô hình nhận biết lớp thiểu số

Trong việc chọn tham số, cũng có nhiều cách tiếp cận, chẳng hạn với phương pháp tối ưu với Bayes (Bayesian Optimization); ở đây dùng Grid Search kết hợp cùng kỹ thuật Cross – Validation để tìm các tham số tối ưu.

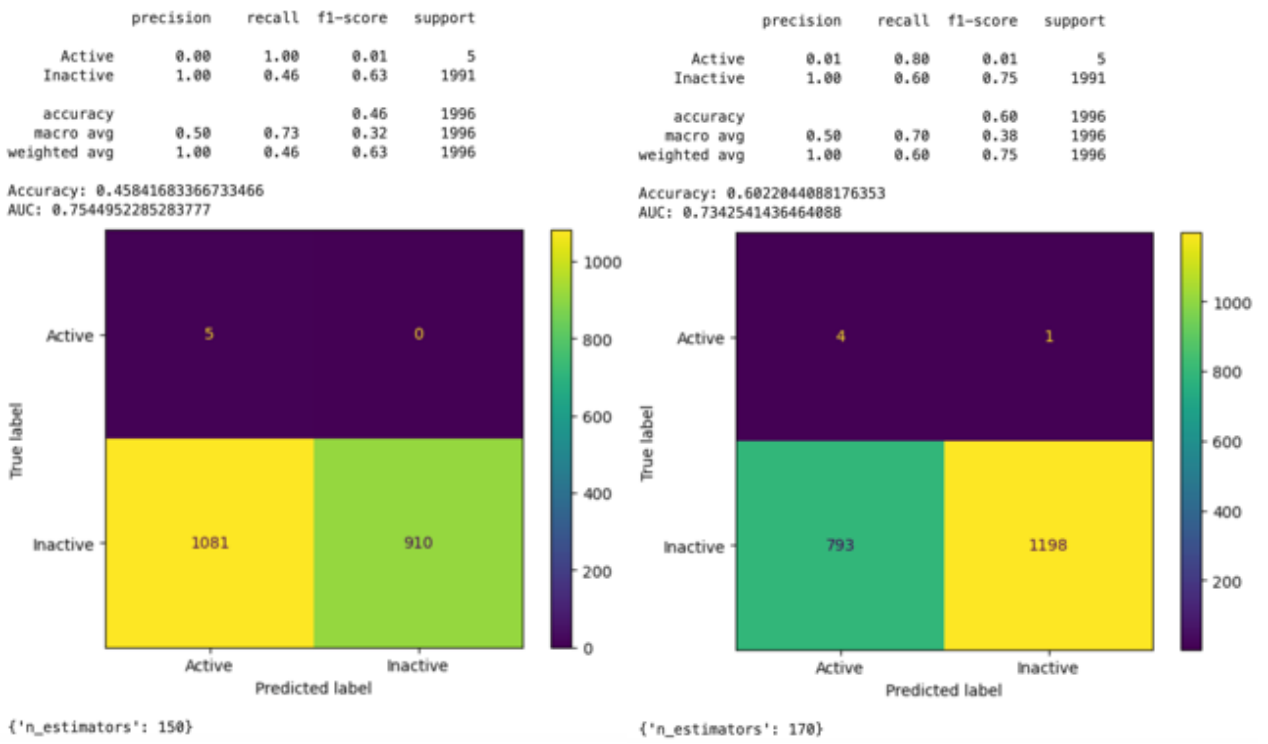
Kết quả sắp xếp theo thước đo đánh giá như Bảng 1 được thực thi bằng chương trình Python dùng các thư viện như trình bày. Chương trình cũng như dữ liệu và kết quả 7 lần thử nghiệm cho 45 phương pháp tiếp cận khác nhau cho mỗi lần được lưu trữ tại GitHub với địa chỉ https://github.com/langtv/cheminformatics.

Bảng 1. *Kết quả tính toán được sắp xếp theo Score với mức độ ưu tiên như trên của lần thử nghiệm thứ #3.*

ID	Phương pháp tiếp cận	Precision	G-mean	AUC	Đoán Active		Đoán Inactive		Score
					Đúng	Nhầm	Đúng	Nhầm	
18	Easy Ensemble, Feature Reduction, MinMax Scaling	1	0,0044	0,77	5	0	859	1132	2,779
8	RUS Boost, Feature Reduction	1	0,0051	0,73	5	0	1017	974	2,745

ID	Phương pháp tiếp cận	Precision	G-mean	AUC	Đoán Active		Đoán Inactive		Score
41	Easy Ensemble, Feature Reduction, MinMax Scaling, Cluster Centroid	1	0,0026	0,57	5	0	45	1946	2,578
43	RUS Boost, Feature Reduction, Robust Scaling, Cluster Centroid	1	0,0027	0,53	5	0	119	1872	2,538
44	Easy Ensemble, Cluster Centroid	1	0,0026	0,52	5	0	51	1940	2,528
42	Easy Ensemble, Feature Reduction, Cluster Centroid	1	0,0025	0,52	5	0	33	1958	2,528
45	RUS Boost, Feature Reduction, Cluster Centroid	1	0,0025	0,51	5	0	33	1958	2,516
12	Easy Ensemble, Feature Reduction	0,8	0,0045	0,75	4	1	1109	882	2,367
26	RUS Boost, Feature Reduction, Robust Scaling	0,8	0,0047	0,75	4	1	1143	848	2,362
6	EasyEnsemble	0,8	0,0043	0,73	4	1	1076	915	2,346
14	RUS Boost, Feature Reduction, MinMax Scaling	0,8	0,0036	0,66	4	1	877	1114	2,273
2	RUS Boost	0,8	0,0041	0,66	4	1	1030	961	2,271
30	Easy Ensemble, Feature Reduction, Robust Scaling	0,6	0,0039	0,75	3	2	1222	769	1,957
24	Easy Ensemble, Feature Reduction, Standard Scaling	0,6	0,0037	0,75	3	2	1189	802	1,956
20	RUS Boost, Feature Reduction, Standard Scaling	0,4	0,0025	0,61	2	3	1201	790	1,421
31	Easy Ensemble, Feature Reduction, MinMax Scaling, SMOTE	0,2	0,0188	0,87	1	4	1939	52	1,328
19	Neural Nework, Feature Reduction, Standard Scaling	0	0,0000	0,87	0	5	1988	3	0,875
25	Neural Nework, Feature Reduction, Robust Scaling	0		0,87	0	5	1991	0	0,869
1	Neural Nework	0		0,80	0	5	1991	0	0,802
36	Easy Ensemble, Feature Reduction, MinMax Scaling, ADASYN	0	0,0000	0,79	0	5	1962	29	0,789
39	Easy Ensemble, ADASYN	0	0,0000	0,77	0	5	1984	7	0,771
17	Random Forest, Feature Reduction, MinMax Scaling	0		0,76	0	5	1991	0	0,756
32	Easy Ensemble, Feature Reduction, SMOTE	0	0,0000	0,75	0	5	1975	16	0,751

ID	Phương pháp tiếp cận	Precision	G-mean	AUC	Đoán Active		Đoán Inactive		Score
37	Easy Ensemble, Feature Reduction, ADASYN	0	0,0000	0,74	0	5	1982	9	0,744
16	Balanced Bagging, Feature Reduction, MinMax Scaling	0	0,0000	0,74	0	5	1990	1	0,740
11	Random Forest, Feature Reduction	0		0,70	0	5	1991	0	0,703
5	Random Forest	0		0,69	0	5	1991	0	0,685
29	Random Forest, Feature Reduction, Robust Scaling	0		0,66	0	5	1991	0	0,659
23	Random Forest, Feature Reduction, Standard Scaling	0		0,66	0	5	1991	0	0,657
13	Neural Nework, Feature Reduction, MinMax Scaling	0	0,0000	0,66	0	5	1988	3	0,657
34	Easy Ensemble, SMOTE	0	0,0000	0,64	0	5	1983	8	0,640
7	Neural Nework, Feature Reduction	0		0,58	0	5	1991	0	0,583
22	Balanced Bagging, Feature Reduction, Standard Scaling	0	0,0000	0,57	0	5	1988	3	0,574
28	Balanced Bagging, Feature Reduction, Robust Scaling	0	0,0000	0,57	0	5	1990	1	0,574
10	Balanced Bagging, Feature Reduction	0	0,0000	0,57	0	5	1988	3	0,568
4	Balanced Bagging	0	0,0000	0,57	0	5	1987	4	0,565
21	Gradient Boosting, Feature Reduction, Standard Scaling	0	0,0000	0,50	0	5	1981	10	0,502
27	Gradient Boosting, Feature Reduction, Robust Scaling	0	0,0000	0,50	0	5	1983	8	0,502
33	RUS Boost, Feature Reduction, Robust Scaling, SMOTE	0	0,0000	0,50	0	5	1986	5	0,499
38	RUS Boost, Feature Reduction, Robust Scaling, ADASYN	0	0,0000	0,50	0	5	1986	5	0,499
35	RUS Boost, Feature Reduction, SMOTE	0	0,0000	0,50	0	5	1983	8	0,498
40	RUS Boost, Feature Reduction, ADASYN	0	0,0000	0,50	0	5	1982	9	0,498
9	Gradient Boosting, Feature Reduction	0	0,0000	0,49	0	5	1978	13	0,490
15	Gradient Boosting, Feature Reduction, MinMax Scaling	0	0,0000	0,39	0	5	1974	17	0,385
3	Gradient Boosting	0	0,0000	0,38	0	5	1980	11	0,382



Hình 6. *Kết quả với phương pháp tiếp cận với Easy Ensemble và RUS Boosting*

Để mô hình có thể sử dụng trong việc phân lớp được, thì $AUC \geq 0.7$; trong 7 lần thử nghiệm mang tính thực nghiệm với số phương pháp tiếp cận tất cả là $45 \times 7 = 315$. Kết quả phương pháp tiếp cận có ID là #18 (*Easy Ensemble, Feature Reduction, MinMax Scaling*) có 4 lần có Score cao nhất và 3 lần xếp thứ hai; phương pháp tiếp cận với ID là #8 (*RUS Boost, Feature Reduction*) có 3 lần có Score cao nhất, và 1 lần xếp thứ hai. Với 2 phương pháp tiếp cận nổi trội này, kết hợp với việc lựa chọn tham số thông qua việc tìm kiếm lưới (Grid Search) và xác thực chéo (Cross – Validation) kết quả như Hình 6.

V. KẾT LUẬN

Kết quả thực nghiệm cho thấy việc sử dụng các phương pháp nêu trên đã mang lại kết quả tốt trong việc phân lớp dữ liệu về hoạt tính và không hoạt tính trong dataset về thực nghiệm sinh học. Qua việc rút gọn thuộc tính dựa trên hệ số tương quan, ta đã giảm số lượng thuộc tính ban đầu để tạo ra một bộ dữ liệu thuộc tính nhỏ gọn và có khả năng dự đoán tốt. Việc xử lý độ chênh lệch của dữ liệu bằng tỷ lệ min-max đã giúp đưa các giá trị dữ liệu về cùng một khoảng giá trị và đảm bảo tính đồng nhất trong quá trình phân lớp. Sử dụng kỹ thuật cân bằng mẫu dữ liệu thông qua Easy Ensemble và RUS Boost cũng đã tăng cường khả năng phân lớp bằng cách tạo thêm mẫu dữ liệu cho các lớp thiểu số. Ngoài ra, do RUS Boost bản chất là tạo ra bộ dữ liệu mang tính ngẫu nhiên, nên giá trị của mỗi lần tính toán có thể khác nhau khi không áp đặt tham số ngẫu nhiên phù hợp.

Bên cạnh đó kỹ thuật tìm kiếm các tham số tối ưu trong quá trình huấn luyện thông qua kỹ thuật tìm kiếm lưới (Grid Search) kết hợp với kỹ thuật kiểm tra chéo (Cross - Validation) trên tập dữ liệu huấn luyện cũng tăng độ chính xác của các mô hình phân lớp.

Một điểm quan trọng chưa được thử nghiệm trong khuôn khổ bài viết này là sử dụng Deep Learning trong việc phân lớp dữ liệu. Deep learning có thể mang lại khả năng học tập sâu và khả năng phân lớp tốt hơn cho dữ liệu phức tạp. Việc áp dụng deep learning trong bài toán này có thể mở ra cơ hội để khám phá các mô hình mạng nơ-ron sâu và tối ưu hóa kết quả phân lớp. Bên cạnh đó, GAN (Generative Adversarial Networks) cũng có thể được sử dụng để xử lý vấn đề mất cân bằng dữ liệu trong bài toán phân lớp. Khi dữ liệu mất cân bằng, tức là số lượng mẫu của một lớp quá ít so với lớp khác, các mô hình học máy có thể gặp khó khăn trong việc học các mẫu của lớp thiểu số. Cách tiếp cận để sử dụng GAN là tạo ra những mẫu dữ liệu nhân tạo cho lớp thiểu số. GAN có thể được huấn luyện để tạo ra những mẫu dữ liệu mới có cấu trúc tương tự với lớp thiểu số, nhưng với đa dạng và số lượng lớn hơn. Quá trình huấn luyện GAN bao gồm việc tạo ra các mẫu dữ liệu giả mạo từ phân phối tiềm năng của lớp thiểu số và cố gắng đánh lừa mô hình phân biệt giữa dữ liệu thật và giả. Khi GAN được huấn luyện thành công, ta có thể sử dụng các mẫu dữ liệu giả tạo bằng GAN để cân bằng lại số lượng mẫu giữa các lớp.

VI. TÀI LIỆU THAM KHẢO

[1] Son Tung Ngo, Shang-Ting Fang, Shu-Hsiang Huang, Chao-Liang Chou, Pham Dinh Quoc Huy, Mai Suan Li, and Yi-Cheng Chen, Anti-arrhythmic Medication Propafenone a Potential Drug for Alzheimer's Disease Inhibiting Aggregation of Aβ: In Silico and in Vitro Studies (2016), Journal of Chemical Information and Modeling. **56** (7), pp.1344-1356, DOI: 10.1021/acs.jcim.6b00029.

[2] Xu, Jun & Hagler, Arnold. Chemoinformatics in Drug Discovery. *Molecules*. **7**(8), 566-600, https://doi.org/10.3390/70800566. (https://www.mdpi.com/1420-3049/7/8/566), 2002.

[3] G. M. Downs, J. M. Barnard, Clustering Methods and Their Uses in Computational Chemistry. *Reviews in Computational Chemistry*, Vol.**18**, pp. 1-40. https://doi.org/10.1002/0471433519.ch1, 2003.

[4] Tetko, I.V., Engkvist, O. From Big Data to Artificial Intelligence: chemoinformatics meets new challenges. *Journal of Cheminformatics*, **12**(74). https://doi.org/10.1186/s13321-020-00475-y, 2020.

[5] Scalfani, V.F., Patel, V.D. & Fernandez, A.M. Visualizing chemical space networks with RDKit and NetworkX. J Cheminform **14**(87). https://doi.org/10.1186/s13321-022-00664-x, 2022.

[6] Ucak, U.V., Ashyrmamatov, I. & Lee, J. Improving the quality of chemical language model outcomes with atom-in-SMILES tokenization. Journal of Cheminformatics **15**(55). https://doi.org/10.1186/s13321-023-00725-9, 2023.

[7] Tiago Alves de Oliveira, Michel Pires da Silva et al. Virtual Screening Algorithms in Drug Discovery: A Review Focused on Machine and Deep Learning Methods, *Drugs and Drug Candidates* **2**(2), 311-334; https://doi.org/10.3390/ddc2020017, 2023.

[8] Dutschmann, TM., Kinzel, L., ter Laak, A. et al. Large-scale evaluation of k-fold cross-validation ensembles for uncertainty estimation. Journal of Cheminformatics **15**(49). https://doi.org/10.1186/s13321-023-00709-9, 2023.

[9] Huawei Feng, Li Zhang, Shimeng Li, et al. Predicting the reproductive toxicity of chemicals using ensemble learning methods and molecular fingerprints, *Toxicology Letters*, V.**340**, 1 April 2021, pp.4-14,ISSN:0378-4274, https://doi.org/10.1016/j.toxlet.2021.01.002 (https://www.sciencedirect.com/science/article/pii/S0378427421000035), 2021.

[10] Liu XY, Wu J, Zhou ZH (2009). Exploratory undersampling for class-imbalance learning. *IEEE Trans Syst Man Cybern B Cybern*. **39**(2):539-50. doi: 10.1109/TSMCB.2008.2007853. Epub 2008 Dec 16. PMID: 19095540.

[11] Lichman M. UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science, 2013.

[12] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse and A. Napolitano. RUSBoost: Improving classification performance when training data is skewed, 2008 19th *International Conference on Pattern Recognition*, Tampa, FL, USA, 2008, pp. 1-4, doi: 10.1109/ICPR.2008.4761297, 2008.

[13] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse and A. Napolitano. RUSBoost: A Hybrid Approach to Alleviating Class Imbalance, *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, **40**(1), pp. 185-197, Jan. 2010, doi: 10.1109/TSMCA.2009.2029559, 2010.

THE ADVANCEMENTS AND CHALLENGES OF MACHINE LEARNING IN CHEMINFORMATICS

Le Thi Thuy Huong, Tran Van Lang, Pham Minh Quan

ABSTRACT— Machine Learning (ML) has become one of the powerful techniques in Cheminformatics, also known as Computational Chemistry. It has been applied to various problems in the field. For instance, in Chemistry, machine learning is used in drug discovery, toxicity prediction, and materials design. In this paper, we aim to provide a general survey of ML in Cheminformatics. We begin by discussing the fundamental concepts of ML and then explore different types of ML algorithms that have been applied to Cheminformatics problems. This provides researchers and practitioners in the field of Cheminformatics with a comprehensive understanding of the application of computational techniques and methods. Additionally, we present some challenges and opportunities for further research. The final part of paper showcases a case study on activity prediction based on a dataset containing screening assays performed by the Burnham Center for Chemical Genomics, targeting the inhibition of VCAM-1 cell surface expression induced by the TNFa gene. This dataset exhibits a significant imbalance between active and inactive compounds, making it an interesting sample for experimentation. The results indicate that the selected classification model is relatively appropriate based on the AUC and G-mean metric.

Lê Thị Thuỳ Hương (1996), tốt nghiệp Khoa Hóa tại Trường Đại học Sư phạm Hà Nội 2, năm 2018. Hiện là nghiên cứu sinh tại Học viện Khoa học và Công nghệ, nghiên cứu viên tại Viện Hóa học các hợp chất thiên nhiên, Viện Hàn Lâm KH&CN Việt Nam. Lĩnh vực quan tâm hiện nay là sử dụng công cụ hỗ trợ máy tính trong nghiên cứu các hoạt chất thiên nhiên nhằm sàng lọc tìm kiếm và thiết kế thuốc mới và khảo sát mối liên quan hoạt tính cấu trúc.

Trần Văn Lăng (1959), tốt nghiệp Khoa Toán tại Trường Đại học Tổng hợp TPHCM năm 1982; thực tập sinh tại Trung tâm Tính toán Dorodnitsyn, Viện Hàn lâm Khoa học Liên Xô; nhận học vị tiến sĩ Toán – Lý năm 1996. Ông nguyên là nghiên cứu viên cao cấp của Viện Hàn lâm Khoa học và Công nghệ Việt Nam. Hiện nay là phó giáo sư tin học của Trường Đại học Ngoại ngữ - Tin học TP. HCM. Lĩnh vực quan tâm hiện nay của ông là tính toán hiệu năng cao trên hệ thống phân tán cũng như hệ thống đa xử lý; giải quyết một số bài toán đặt ra trong Sinh tin, Hoá tin bằng các phương pháp trong học máy, học sâu.

Phạm Minh Quân (1989), tốt nghiệp Khoa Hóa tại Trường Đại học Bách Khoa Hà Nội năm 2011; Nghiên cứu sinh tại Đại học Paul Sabatier, Cộng hòa Pháp nhận học vị tiến sĩ năm 2016. Hiện nay là phó giáo sư Hóa học, Phó Viện trưởng Viện Hóa học các Hợp chất thiên nhiên, Viện Hàn Lâm Khoa học và Công nghệ Việt Nam. Lĩnh vực quan tâm hiện nay của ông là sử dụng công cụ hỗ trợ máy tính trong nghiên cứu các hoạt chất thiên nhiên nhằm sàng lọc tìm kiếm và thiết kế thuốc mới và khảo sát mối liên quan hoạt tính cấu trúc và phân lập xác định cấu trúc và đánh giá hoạt tính sinh học các hợp chất thiên nhiên, đặc biệt là các hợp chất lipit.