HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

NGUYỄN MINH HÀ

NGHIÊN CỨU PHÂN LỚP TRÊN DỮ LIỆU MẤT CÂN BẰNG VÀ ỨNG DỤNG

LUẬN VĂN THẠC SĨ KỸ THUẬT

(Theo định hướng ứng dụng)

HÀ NỘI - 2020

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

NGUYỄN MINH HÀ

NGHIÊN CỨU PHÂN LỚP TRÊN DỮ LIỆU MẤT CÂN BẰNG VÀ ỨNG DỤNG

CHUYÊN NGÀNH: KHOA HỌC MÁY TÍNH

MÃ SỐ: 8.48.01.01

LUẬN VĂN THẠC SĨ KỸ THUẬT

(Theo định hướng ứng dụng)

NGƯỜI HƯỚNG DẪN KHOA HỌC: TS. VŨ VĂN THỎA

HÀ NỘI - 2020

LỜI CAM ĐOAN

Tôi cam đoan đây là công trình nghiên cứu của tôi. Nội dung của luận văn có tham khảo và sử dụng các tài liệu, thông tin được đăng tải trên những tạp chí và các trang web theo danh mục tài liệu tham khảo. Tất cả các tài liệu tham khảo đều có xuất xứ rõ ràng và được trích dẫn hợp pháp.

Tôi xin hoàn toàn chịu trách nhiệm và chịu mọi hình thức kỷ luật theo quy định cho lời cam đoan của mình.

Hà nội, ngày tháng năm 2020

Người cam đoan

Nguyễn Minh Hà

LỜI CẢM ƠN

Trong thời gian thực hiện luận văn này, Học viên luôn nhận được sự hướng dẫn, chỉ bảo rất tận tình của Thầy giáo - **TS. Vũ Văn Thỏa**, giảng viên Khoa Công nghệ thông tin 1 là cán bộ trực tiếp hướng dẫn khoa học. Thầy đã dành nhiều thời gian trong việc hướng dẫn học viên cách đọc tài liệu, thu thập và đánh giá thông tin cùng phương pháp nghiên cứu để hoàn thành một luận văn cao học.

Học viên xin chân thành cảm ơn các Thầy, Cô giáo công tác trong Học viện Công nghệ Bưu chính Viễn thông đã luôn nhiệt tình giúp đỡ và tạo điều kiện tốt nhất cho học viên trong suốt quá trình học tập tại trường.

Xin chân thành cảm ơn các anh, các chị và các bạn học viên cùng lớp Cao học đã luôn động viên, giúp đỡ và nhiệt tình chia sẻ với học viên những kinh nghiệm học tập, công tác trong suốt khoá học.

Học viên cũng xin chân thành cảm ơn các đồng chí lãnh đạo và các bạn đồng nghiệp tại cơ quan đã luôn tạo mọi điều kiện tốt nhất để học viên có thể hoàn thành tốt đẹp khoá học Cao học này.

Học viên xin chân thành cảm ơn !

Hà Nội, ngày tháng năm 2020

Người viết

Nguyễn Minh Hà

MỤC LỤC

DANH MỤC CÁC THUẬT NGỮ VIẾT TẮT

Viết tắt	Tiếng Anh	Tiếng việt
AUC	Area Under the Curve	Diện tích nằm dưới đường cong ROC
DEC-SVM	Differential Evolution Clustering Support Vector Machines	Phân cụm tiến hóa khác biệt hỗ trợ máy vec-tơ
FN	False Negative	Số lượng phần tử lớp thiểu số bị phân loại nhầm là phần tử lớp đa số.
FP	False Positive	Số lượng phần tử lớp đa số bị phân loại nhầm là phần tử lớp đa số.
HBU	Hypothesis margin based Borderline Under-sampling	Giảm phần tử dựa vào giá trị lề giả thuyết ưu tiên loại bỏ các phần tử nằm ở biên
HMU	Hypothesis Margin based Undersampling	Giảm phần tử dựa vào giá trị lề giả thuyết
KDD	Knowledge Discovery and Data Mining	Phát hiện tri thức và khai phá dữ liệu
K-NN	K-nearest neighbors	K láng giềng gần nhất
RBU	Random border undersampling	Giảm phần tử ngẫu nhiên trên đường biên
ROC	Receiver operating characteristic	Đường cong đặc trưng hoạt động của bộ thu nhận

SMOTE	Synthetic Minority Over-sampling Technique	Phương pháp sinh thêm mẫu nhân tạo lớp thiểu số
SVM	Support Vector Machines	Máy véc tơ hỗ trợ
TN	True Negative	Số lượng phần tử lớp đa số được phân loại chính xác.
TP	True Positive	Số lượng phần tử lớp thiểu số được phân loại chính xác.
WEKA	Waikato Environment for Knowledge Acquisition	Công cụ kiểm thử học máy

DANH MỤC BẢNG

DANH MỤC HÌNH

MỞ ĐẦU

Trong những năm gần đây, vấn đề học máy từ dữ liệu phân bố không cân bằng là một thách thức lớn cho các nhà nghiên cứu trong rất nhiều miền ứng dụng thực tế: mạng internet, bảo mật, viễn thông, quản lý tài chính và tin sinh học… Việc phân tích và hiểu được dữ liệu thô là mục đích của các hệ thống xử lý hỗ trợ ra quyết định ngày càng đóng vai trò quan trọng và trở nên cần thiết. Chúng được áp dụng và đã đạt được nhiều thành công to lớn trong nhiều ứng dụng của cuộc sống như khai phá tri thức, kỹ thuật xử lý dữ liệu, và nhiều ứng dụng khác. Tuy nhiên, những năm gần đây với sự xuất hiện của dữ liệu phân bố mất cân bằng đang trở thành nguyên nhân gây ra nhiều khó khăn ảnh hưởng đến các thuật toán học máy chuẩn, những thuật toán được thiết kế và áp dụng vào ứng dụng của dữ liệu phân bố cân bằng . Khi những thuật toán chuẩn này được áp dụng vào dữ liệu mất cân bằng, chúng xử lý dữ liệu một cách lệch lạc, dẫn đến không đạt được độ chính xác cao giữa các lớp của dữ liệu. Thêm vào đó, vấn đề phân bố dữ liệu mất cân bằng đang ngày càng trở nên quan trọng trong thực tế, với lượng lớn các ứng dụng. Do đó vấn đề này đang nhận được sự quan tâm từ các quỹ tài trợ của chính phủ, các viện nghiên cứu, các cơ sở công nghiệp…. Khi áp dụng các thuật toán phân lớp truyền thống lên các tập dữ liệu mất cân bằng, hầu hết các phần tử thuộc lớp đa số sẽ được phân lớp đúng và các phần tử thuộc lớp thiểu số cũng sẽ được gán nhãn lớp là nhãn lớp của lớp đa số. Điều này dẫn đến kết quả là độ chính xác (accuracy) của việc phân lớp có thể rất cao, trong khi giá trị độ nhạy (sensitivity) lại rất thấp.

Xuất phát từ thực tế và mục tiêu như trên, học viên chọn thực hiện đề tài luận văn tốt nghiệp chương trình đào tạo thạc sĩ có tên "**Nghiên cứu phân lớp trên dữ liệu mất cân bằng và ứng dụng**".

Mục tiêu của luận văn là nghiên cứu một số kỹ thuật để nâng cao hiệu năng phân lớp dữ liệu trên tập dữ liệu mất cân bằng và ứng dụng.

Đối tượng nghiên cứu của luận văn là bài toán phân lớp dữ liệu trên dữ liệu mất cân bằng và các vấn đề liên quan.

Phạm vi nghiên cứu của luận văn là các thuật toán, phương pháp để phân lớp dữ liệu mất cân bằng và ứng dụng.

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

Chương 1: Tổng quan về bài toán phân lớp dữ liệu trên các dữ liệu mất cân bằng

Nội dung chính của chương 1 là khảo sát tổng quan về bài toán phân lớp dữ liệu trên các tập dữ liệu mất cân bằng và các vấn đề liên quan.

Chương 2: Một số thuật toán phân lớp dữ liệu mất cân bằng

Nội dung chính của chương 2 là khảo sát một số kỹ thuật để nâng cao hiệu năng phân lớp dữ liệu cho các dữ liệu mất cân bằng và một số vấn đề liên quan.

Chương 3: Ứng dụng

Nội dung chính của chương 3 là thực hiện thử nghiệm và đánh giá một số thuật toán phân lớp dữ liệu cho tập dữ liệu mất cân bằng được lựa chọn.

CHƯƠNG 1. TỔNG QUAN VỀ BÀI TOÁN PHÂN LỚP DỮ LIỆU TRÊN CÁC DỮ LIỆU MẤT CÂN BẰNG

Nội dung của chương này sẽ khảo sát về bài toán phân lớp dữ liệu, học máy, dữ liệu mất cân bằng và các vấn đề liên quan.

1.1. Giới thiệu về bài toán phân lớp dữ liệu

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

- *Phân lớp dữ liệu:*

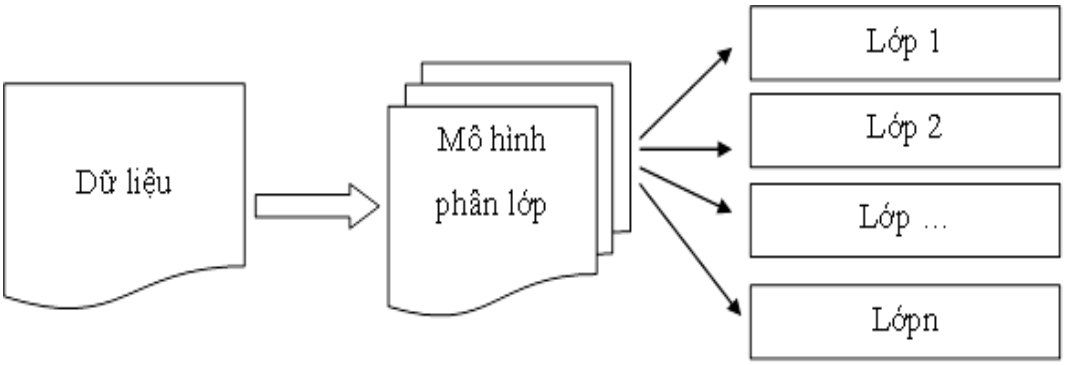
Phân lớp dữ liệu(classification) là một trong những hướng nghiên cứu chính của khai phá dữ liệu. Thực tế đặt ra nhu cầu là từ một cơ sơ dữ liệu với nhiều thông tin ẩn con người có thể rút trích ra các quyết định nghiệp vụ thông minh. Phân lớp là một dạng của phân tích dữ liệu nhằm rút trích ra một mô hình mô tả các lớp dữ liệu quan trọng hay dự đoán xu hướng dữ liệu trong tương lai.

Phân lớp dự đoán giá trị của những nhãn xác định hay những giá trị rời rạc, có nghĩa là phân lớp thao tác với những đối tượng dữ liệu mà có bộ giá trị là biết trước. Tóm lại, phân lớp là quá trình nhóm các đối tượng giống nhau vào một lớp dựa trên các đặc trưng dữ liệu của chúng. Trong những năm qua, phân lớp dữ liệu đã thu hút sự quan tâm các nhà nghiên cứu trong nhiều lĩnh vực khác nhau như: học máy(machine learning), hệ chuyên gia (expert system), thống kê (statistics) Công nghệ này cũng ứng dụng trong nhiều lĩnh vưc khác nhau như: thương mại, ngân hàng, maketing, nghiên cứu thị trường, bảo hiểm, y tế, giáo dục…

- *Bài toán phân lớp dữ liệu:*

Là quá trình phân lớp một đối tượng dữ liệu vào một hay nhiều lớp đã cho trước nhờ một mô hình phân lớp (model). Mô hình này được xây dựng dựa trên một tập dữ liệu được xây dựng trước đó có gán nhãn (còn gọi là tập huấn luyện). Quá trình phân lớp là quá trình gán nhãn cho đối tượng dữ liệu. Nhiệm vụ của bài toán phân lớp là cần tìm một mô hình phân lớp để khi có dữ liệu mới thì có thể xác định

được dữ liệu đó thuộc vào phân lớp nào. Bài toán phân lớp dữ liệu có thể được mô tả như hình 1.1 dưới đây [7].



Hình 1.1 Mô hình mô tả bài toán phân lớp dữ liệu

Bài toán phân lớp dữ liệu có thể phát biểu tổng quát như sau:

Cho $U = \{A_1, A_2, ..., A_m\}$ là tập có m thuộc tính, $Y = \{y_1, y_2, ..., y_n\}$ là tập các nhãn của lớp: với $D = A_1 \times \ldots \times A_m$ là tích Đề - các của các miền của m thuộc tính tương ứng có n số lớp và N là số mẫu dữ iệu. Mỗi dữ liệu $d_i \in D$ thuộc một lớp $y_i \in Y$ tương ứng tạo thành từng cặp $(d_i, y_i) \in (D, Y)$.

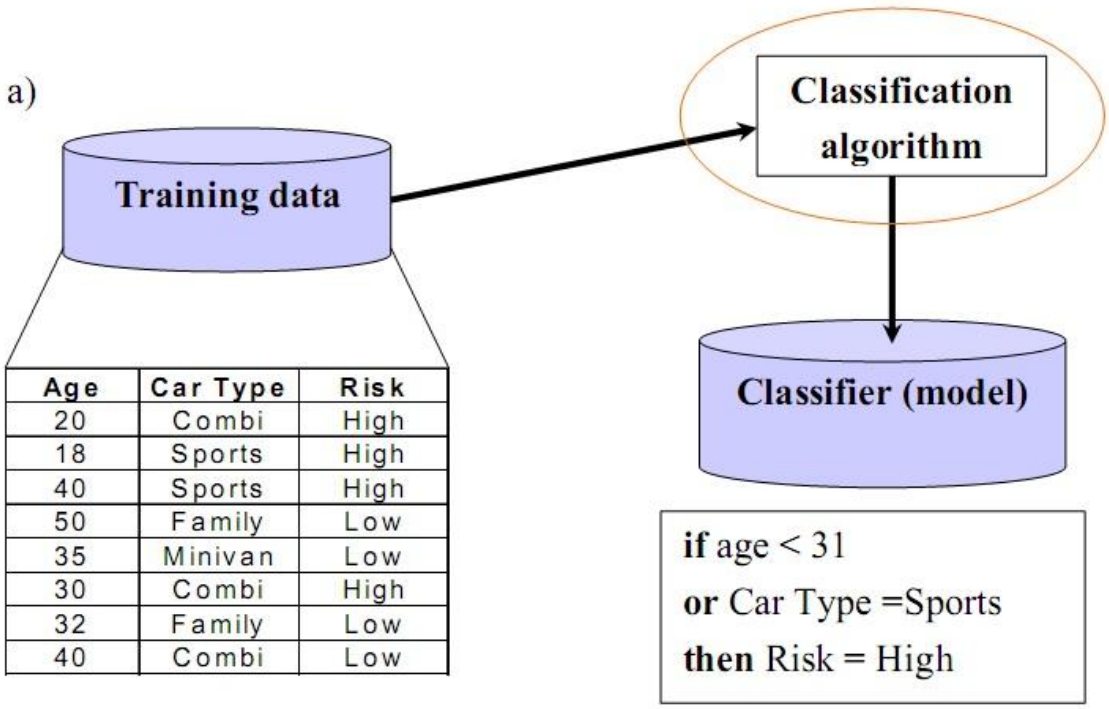
1.1.2. Quy trình thực hiện phân lớp dữ liệu:

Quy trình thực hiện phân lớp dữ liệu thường được thực hiện theo 2 bước: Bước thứ nhất (learning) quá trình học để xây dựng mô hình phân lớp và bước thứ hai áp dụng mô hình phân lớp ở bước thứ nhất để phân lớp dữ liệu mới.

- *Bước thứ nhất (learning)*

Quá trình học nhằm xây dựng một mô hình mô tả một tập các lớp dữ liệu hay các khái niệm định trước. Đầu vào của quá trình này là một tập dữ liệu có cấu trúc được mô tả bằng các thuộc tính và được tạo ra từ tập các bộ giá trị của các thuộc tính đó. Mỗi bộ giá trị được gọi chung là một phần tử dữ liệu (data tuple), có thể là các mẫu (sample), ví dụ (example), đối tượng (object), bản ghi (record) hay trường hợp (case). Trong tập dữ liệu này, mỗi phần tử dữ liệu được giả sử thuộc về một lớp định trước, lớp ở đây là giá trị của một thuộc tính được chọn làm thuộc tính gán

nhãn lớp hay thuộc tính phân lớp (class lable attribute). Đầu ra của bước này thường là các quy tắc phân lớp dưới dạng luật dạng if-then, cây quyết định, công thức logic, hay mạng nơron. Quá trình này được mô tả như hình 1.2 [7].



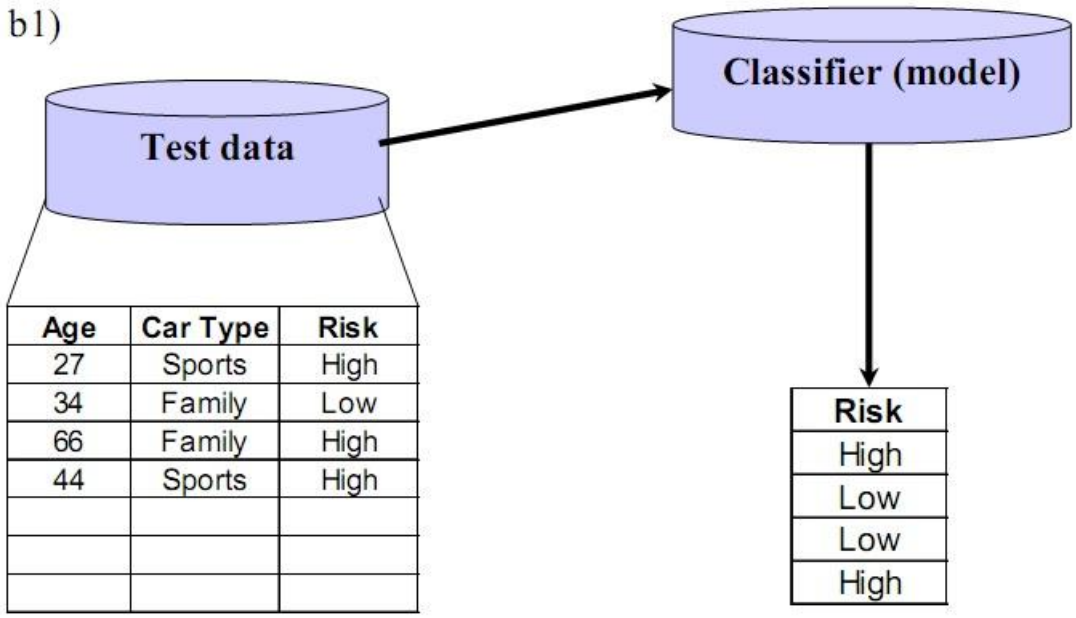
Age	Car Type	Risk
20	Combi	High
18	Sports	High
40	Sports	High
50	Family	Low
35	Minivan	Low
30	Combi	High
32	Family	Low
40	Combi	Low

Hình 1.2 Quá trình phân lớp dữ liệu - (a) Bước xây dựng mô hình phân lớp

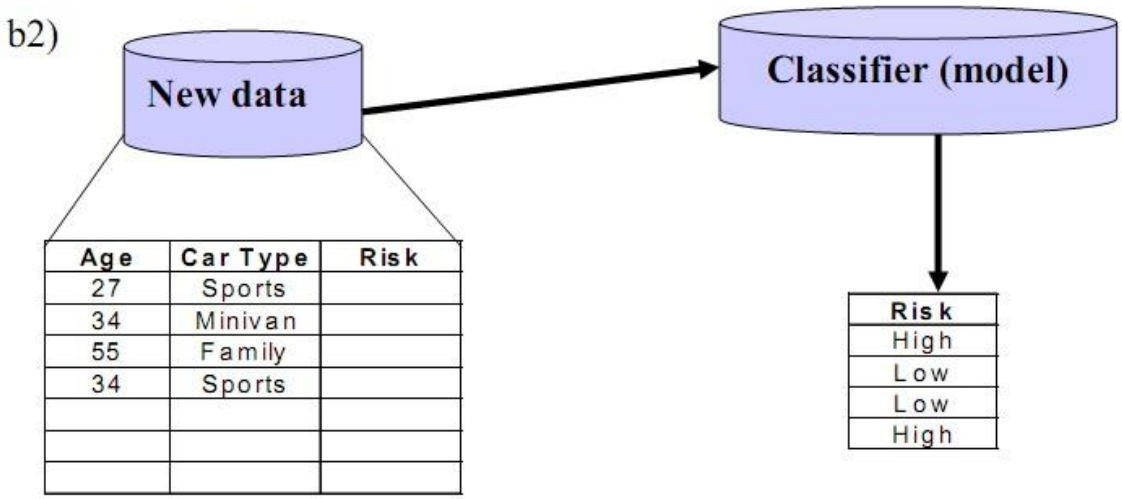
- *Bước thứ hai (classification)*

Bước thứ hai dùng mô hình đã xây dựng ở bước thứ nhất để phân lớp dữ liệu mới. Trước tiên độ chính xác mang tính chất dự đoán của mô hình phân lớp vừa tạo ra được ước lượng. Holdout là một kỹ thuật đơn giản để ước lượng độ chính xác đó. Kỹ thuật này sử dụng một tập dữ liệu kiểm tra với các mẫu đã được gán nhãn lớp. Các mẫu này được chọn ngẫu nhiên và độc lập với các mẫu trong tập dữ liệu đào tạo. Độ chính xác của mô hình trên tập dữ liệu kiểm tra đã đưa là tỉ lệ phần trăm các mẫu trong tập dữ liệu kiểm tra được mô hình phân lớp đúng (so với thực tế). Nếu độ chính xác của mô hình được ước lượng dựa trên tập dữ liệu đào tạo thì kết quả thu được là rất khả quan vì mô hình luôn có xu hướng "quá vừa" dữ liệu. Quá vừa dữ liệu là hiện tượng kết quả phân lớp trùng khít với dữ liệu thực tế vì quá trình xây

dựng mô hình phân lớp từ tập dữ liệu đào tạo có thể đã kết hợp những đặc điểm riêng biệt của tập dữ liệu đó. Do vậy cần sử dụng một tập dữ liệu mà giá trị của thuộc tính phân lớp là chưa biết.



Hình 1.3 Quá trình phân lớp dữ liệu - (b1) Ước lượng độ chính xác của mô hình



Hình 1.4 Quá trình phân lớp dữ liệu - (b2) Phân lớp dữ liệu mới

Trong mô hình phân lớp, thuật toán phân lớp giữ vai trò trung tâm, quyết định tới sự thành công của mô hình phân lớp. Do vậy chìa khóa của vấn đề phân lớp dữ liệu là tìm ra được một thuật toán phân lớp nhanh, hiệu quả, có độ chính xác cao và có khả năng mở rộng được. Trong đó khả năng mở rộng của thuật toán được đặc biệt chú trọng và phát triển.

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

Để đánh giá một mô hình phân lớp dữ liệu, việc đầu tiên, và thông thường nhất mà các chuyên gia thường thực hiện đó là chia tập dữ liệu thành các phần phục vụ cho việc huấn luyện mô hình (Training Data) và kiểm chứng mô hình (Testing Data). Xây dựng model thông qua việc lựa chọn thuật toán, xác định các biến dữ liệu, hay điều chỉnh các tham số… sao cho phù hợp sẽ dựa trên dữ liệu training. Sau khi hoàn thành cơ bản mô hình, tiến hành sử dụng dữ liệu test để thử nghiệm mô hình bằng cách kết hợp nhiều công thức, chỉ số khác nhau để đánh giá mức độ chính xác với input là các kết quả có được khi chạy mô hình với dữ liệu test.

Vậy quá trình đánh giá mô hình phân lớp thường chia làm 2 phần hay hướng tiếp cận: *phân chia bộ dữ liệu để huấn luyện* và *kiểm chứng mô hình*; sử dụng các chỉ số để đánh giá độ hiệu quả.

Một số tiêu chí mô tả độ hiệu quả của mô hình phân lớp [7]:

- Accuracy: khả năng mô hình phân lớp dự báo, phân loại hay xác định đúng class cho dữ liệu cần phân loại.
- Speed: tốc độ hay khả năng mô hình đưa ra kết quả phân tích nhanh chóng, nó còn liên qua đến chi phí tính toán khi xây dựng, và sử dụng mô hình.
- Robustness: khả năng của mô hình xử lý nhiễu hoặc dữ liệu với các giá trị bị thiếu và đưa ra dự đoán chính xác.
- Scalability: Phương pháp hay khả năng xây dựng mô hình phân lớp hiệu quả trong xử lý, phân tích lượng lớn dữ liệu.
- Interpreability: liên quan đến khả năng giải thích, mứa độ phức tạp của

mô hình hay nói cách khác cấu trúc mô hình, phương pháp xây dựng mô hình có dễ hiểu hay không.

Đánh giá mô hình là một phần không thể thiếu trong quá trình phát triển mô hình, giúp tìm ra mô hình tốt nhất, phù hợp nhất với mục tiêu nghiên cứu, và loại dữ liệu. Có 2 phương pháp đánh giá phổ biến là holdout và cross-validation.

- ***Holdout:***

Phương pháp Holdout, là phương pháp phân chia ngẫu nhiên tập dữ liệu thành 2 tập dữ liệu độc lập là: tập dữ liệu huấn luyện và tập kiểm định mô hình. Mục đích của phương pháp Holdout là kiểm tra độ hiệu quả của mô hình khi sử dụng nhiều tập dữ liệu khác nhau. Cụ thể trong phương pháp Holdout ta sẽ có các tập dữ liệu:

- Training set: dữ liệu phục vụ xây dựng mô hình, xác định các thuật toán, biến dữ liệu phù hợp
- Validation set: là dữ liệu được sử dụng để đánh giá hiệu suất của mô hình được xây dựng trong giai đoạn huấn luyện, hỗ trợ thử nghiệm để tinh chỉnh các tham số mô hình và chọn mô hình hoạt động tốt nhất. Tuy nhiên không phải mọi thuật toán phân lớp nào cũng cần Validation set, nên phương pháp Holdout thông thường chỉ cần 2 tập dữ liệu training và test data mà thôi.
- Test set: là dữ liệu được sử dụng để đánh giá độ hiệu quả của mô hình, mức độ chính xác trong việc phân loại dữ liệu (không chứa nhãn phân loại).

Thường tỉ lệ phân chia cho training data set là 70% và test data set là 30%. Ưu điểm của Holdout là nhanh chóng, đơn giản và linh hoạt. Tuy nhiên, phương pháp này thường dẫn đến độ biến thiên cao do sự khác biệt lớn trong 2 tập dữ liệu, dẫn đến sự khác biệt trong việc phân lớp hay dự đoán. Do đó việc áp dụng Holdout cần kết hợp các phương pháp để kiểm tra mức độ khác biệt của 2 tập dữ liệu.

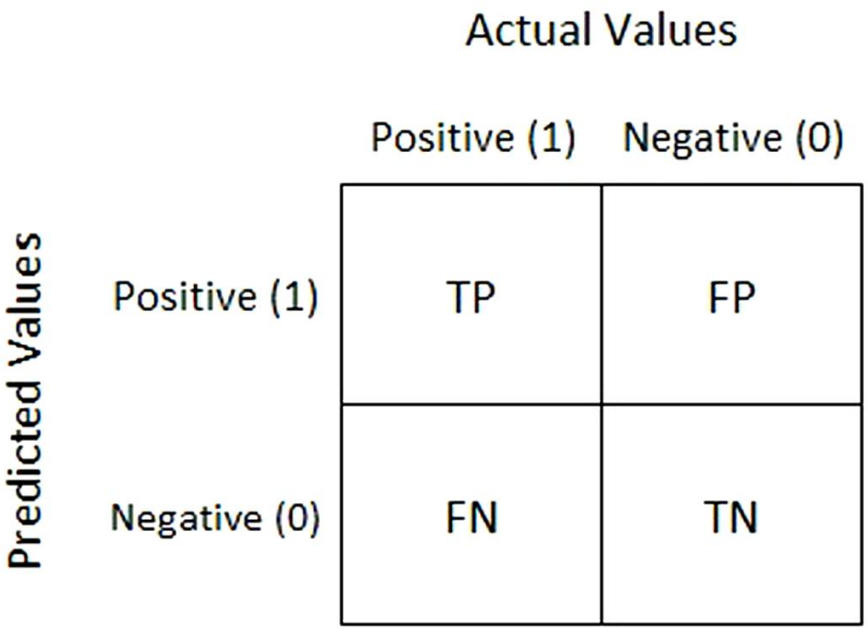
- ***Cross - validation:***

Cross - validation là một kỹ thuật phân chia tập dữ liệu ban đầu thành training data được sử dụng để huấn luyện mô hình và một tập dữ liệu độc lập được sử dụng để đánh giá. Phương pháp phổ biến nhất là K - fold, trong đó bộ dữ liệu ban

đầu được phân chia thành các tập con có kích thước bằng nhau, được gọi là "fold". Giá trị k là số tập dữ liệu được phân chia. Phương pháp này được lặp lại nhiều lần cho đến khi có k số mô hình khác nhau, sao cho mỗi lần, một trong các tập k được sử dụng làm tập kiểm thử và các tập còn lại khác được ghép lại với nhau để tạo thành tập huấn luyện. Việc ước tính độ chính xác hay lỗi (accuracy hay error) được tính trung bình trên tất cả các thử nghiệm k để đánh giá mức độ hiệu quả của cả mô hình.

- ***Confusion Matrix***

Là một phương pháp đánh giá kết quả của những bài toán phân loại với việc xem xét cả những chỉ số về độ chính xác và độ bao quát của các dự đoán cho từng lớp. Một confusion matrix gồm 4 chỉ số sau với mỗi lớp phân loại [16]:



Hình 1.5 Các chỉ số đánh giá mô hình phân lớp

Sau đây, ta sử dụng bài toán về chẩn đoán ung thư để giải thích 4 chỉ số này. Trong bài toán chẩn đoán ung thư ta có 2 lớp: lớp bị ung thư được chẩn đoán Positive và lớp không bị ung thư được chẩn đoán là Negative:

- TP (True Positive): Số lượng dự đoán chính xác. Là khi mô hình dự đoán đúng một người bị ung thư.

- TN (True Negative): Số lượng dự đoán chính xác một cách gián tiếp. Là khi mô hình dự đoán đúng một người không bị ung thư, tức là việc không chọn trường hợp bị ung thư là chính xác.
- FP (False Positive - type 1 error): Số lượng các dự đoán sai lệch. Là khi mô hình dự đoán một người bị ung thư và người đó hoàn toàn khỏe mạnh.
- FN (False Negative - type 2 error): Số lượng các dự đoán sai lệch một cách gián tiếp. Là khi mô hình dự đoán một người không bị ung thư nhưng người đó bị ung thư, tức là việc không chọn trường hợp bị ung thư là sai.

Từ 4 chỉ số này, ta có 2 đại lượng để đánh giá mức độ tin cậy của một mô hình:

- Precision: trong tất các các dự đoán Positive được đưa ra, bao nhiêu dự đoán là chính xác? Chỉ số này được tính theo công thức:

$$\text{Precision} = \frac{TP}{TP + FP}$$

- Recall: Trong tất cả các trường hợp Positive, bao nhiêu trường hợp đã được dự đoán chính xác? Chỉ số này được tính theo công thức:

$$\text{Recall} = \frac{TP}{TP + FN}$$

Giả sử có 1 tập dữ liệu gồm 100 người với 90 người khỏe mạnh (Negative) và 10 người mắc bệnh ung thư (Positive) và mô hình dự đoán đúng 2/10 người bị ung thư, tức là đưa ra dự đoán 2 người bị ung thư thì cả 2 dự đoán đều chính xác. Như vậy, chỉ số Precision khi dự đoán lớp ung thư là 1.

Tuy nhiên, 8/10 người còn lại đã bị bỏ qua, từ đó chỉ số về Recall chỉ là 0.2 - con số rất thấp. Để đánh giá độ tin cậy chung của mô hình, người ta đã kết hợp 2 chỉ số Precision và Recall thành 1 chỉ số duy nhất: F - score, được tính theo công thức:

$$\text{F - measure} = \frac{2 * \text{Re}\,call\ * \text{Pr}\,ecision}{\text{Re}\,call\ + \text{Pr}\,ecision}$$

Một mô hình có chỉ số F - score cao khi cả 2 chỉ số Precision và Recall đều cao. Trường hợp xấu nhất khi 1 trong 2 chỉ số Precision hoặc Recall bằng 0 sẽ kéo F- score về 0, tốt nhất khi cả 2 đều bằng 1 và F - score là 1.

Việc sử dụng chỉ số F - score, cho biết một thước đo đáng tin cậy về hiệu năng của mô hình trong các bài toán phân loại, đặc biệt khi dữ liệu về một lớp lớn hơn gấp nhiều lần so với dữ liệu lớp còn lại.

1.2. Dữ liệu mất cân bằng

1.2.1. Khái niệm về dữ liệu mất cân bằng

Ngày nay, với sự ảnh hưởng của công nghệ thông tin đến các lĩnh vực trong cuộc sống, khai thác và xử lý thông tin là một vấn đề rất cần thiết phải chú trọng, nó đóng vai trò quyết định sự thành công trong một số lĩnh vực.

Nhưng dữ liệu thu thập được trong thực tế xuất hiện nhiều các bộ dữ liệu mất cân bằng, nghĩa là trong tập dữ liệu có sự chênh lệch lớn về số lượng các phần tử giữa các lớp. Lớp có nhiều phần tử hơn ta gọi là lớp đa số, lớp có ít phần tử hơn ta gọi là lớp thiểu số. Các bộ dữ liệu trong nhiều ứng dụng thực tế, chẳng hạn như phát hiện các giao dịch gian lận, phát hiện xâm nhập mạng, dự đoán rủi ro trong quản lý, chẩn đoán y khoa, …, đều là các bộ dữ liệu mất cân bằng mà trong đó, lớp người ta cần quan tâm lại chiếm tỉ lệ rất nhỏ so với lớp còn lại [4].

1.2.2. Các đặc điểm phân lớp dữ liệu mất cân bằng:

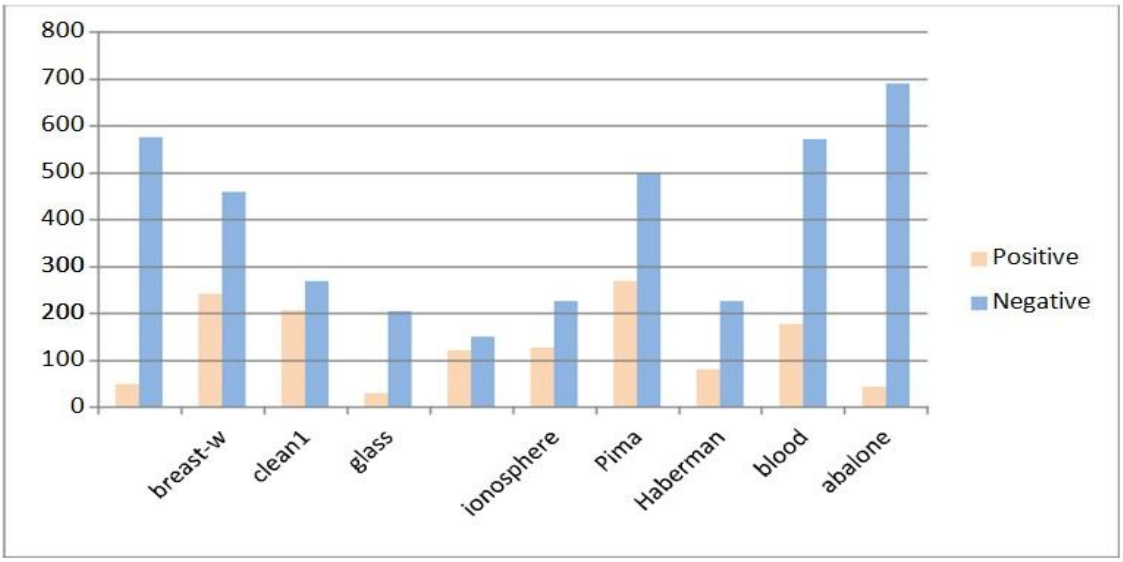
Sự chênh lệch về số lượng giữa lớp đa số và lớp thiểu số làm cho việc phân lớp đúng các mẫu thuộc lớp thiểu số bị giảm hiệu quả. Mức độ mất cân bằng của bộ dữ liệu được biểu thị bằng tỷ lệ giữa số lượng mẫu của hai lớp. Tỷ lệ mất cân bằng của tập dữ liệu càng cao thì việc phát hiện đúng các mẫu của lớp thiểu số càng khó khăn. Trong các ứng dụng thực tế, tỷ lệ mất cân bằng có thể là 1:100, 1:1000, … thậm chí có thể hơn [14]. Vì thế, phân lớp dữ liệu mất cân bằng đã và đang là bài toán được các nhà khoa học đặc biệt quan tâm.

Bảng sau thể hiện tỉ lệ mất cân bằng của một số bộ dữ liệu được lấy từ kho dữ liệu UCI, trong đó *Positive* đại diện cho các phần tử thuộc lớp thiểu số, và *Negative* đại diện cho các phần tử thuộc lớp đa số.

Bảng 1.1 Một số bộ dữ liệu mất cân bằng

Bộ dữ liệu	Positive	Negative	Tỉ lệ mất cân bằng
balance-scale	49	576	1 : 11,75
breast-w	241	458	1:1,9
clean1	207	269	1:1,3
glass	9	205	1 : 22,78
heart-statlog	120	150	1 : 1,25
ionosphere	126	225	1 : 1,79
page-blocks	28	5445	1 : 194,45

Ta có thể thấy được sự chênh lệch rõ rệt về số lượng giữa lớp thiểu số và lớp đa số của một số bộ dữ liệu trong biểu đồ hình dưới đây.



Hình 1.6 Biểu đồ mô tả tỷ lệ chênh lệch giữa lớp thiểu số và đa số

Đối với các bộ dữ liệu mất cân bằng, các bộ phân lớp chuẩn thường có xu hướng thiên vị đối với lớp đa số và bỏ qua lớp thiểu số [5]. Vì vậy, khi áp dụng các giải thuật phân lớp truyền thống chưa thể xây dựng được một bộ phân lớp tốt.

Việc phân loại sai các mẫu thuộc lớp thiểu số có thể gây nên những tổn thất lớn đối với các lĩnh vực thực tế.

1.2.3. Các ứng dụng của phân lớp dữ liệu mất cân bằng

Bài toán phân lớp dữ liệu có rất nhiều ứng dụng trong các lĩnh vực khoa học, công nghệ và đời sống xã hội. Dưới đây, luận văn liệt kê một số ứng dụng chủ yếu của phân lớp dữ liệu.

Trong ngành y tế

Việc ứng dụng phân lớp dữ liệu trong y học ngày càng phổ biến trong việc tìm ra mối liên hệ giữa các triệu chứng bệnh, giữa các bệnh với nhau để hỗ trợ chẩn đoán, điều trị và tiên lượng bệnh. Trong chẩn đoán, phân lớp dữ liệu dùng để nhận dạng và phân loại mẫu trong các thuộc tính đa biến của bệnh nhân. Trong điều trị, phân loại dữ liệu dùng để chọn lựa phương pháp điều trị phù hợp hiệu quả nhất và trong tiên lượng là dự đoán kết quả điều trị, phẫu thuật dựa trên những kết quả điều trị trước đó và tình trạng hiện tại của người bệnh. Ngoài ra có thể hỗ trợ cảnh báo dịch bệnh.

Phân tích thị trường bán lẻ:

Kỹ thuật này có thể cho phép nhà bán lẻ hiểu hành vi mua của người mua. Thông tin này có thể giúp nhà bán lẻ biết nhu cầu của người mua và thay đổi bố cục của cửa hàng cho phù hợp. Sử dụng phân tích phân tích so sánh kết quả giữa các cửa hàng khác nhau, giữa các khách hàng trong các nhóm nhân khẩu học khác nhau có thể được thực hiện.

Trong ngành giáo dục: Dự đoán hành vi học tập trong tương lai của học sinh, nghiên cứu ảnh hưởng của hỗ trợ giáo dục và nâng cao kiến thức khoa học về học tập. Khai phá dữ liệu có thể được sử dụng bởi một tổ chức để đưa ra quyết định chính xác và cũng để dự đoán kết quả của học sinh. Với kết quả, cơ sở giáo dục có thể tập trung vào những gì để dạy và cách giảng dạy.

Quy trình sản xuất:

Lĩnh vực này sử dụng trong thiết kế mức hệ thống để trích xuất các mối quan hệ giữa kiến trúc sản phẩm, danh mục sản phẩm và dữ liệu nhu cầu của khách hàng. Nó cũng có thể được sử dụng để dự đoán thời gian phát triển sản phẩm, chi phí và phụ thuộc giữa các nhiệm vụ khác.

Phát hiện gian lận:

Phương pháp được giám sát bao gồm thu thập các hồ sơ mẫu. Những hồ sơ này được phân loại là gian lận hoặc không gian lận. Một mô hình được xây dựng bằng cách sử dụng dữ liệu này và thuật toán được thực hiện để xác định xem hồ sơ có gian lận hay không.

Hỗ trợ điều tra tội phạm:

Thuật toán phân lớp dữ liệu giúp điều tra tội phạm, giám sát thông tin liên lạc của những kẻ khủng bố bị nghi ngờ.

Ngành tài chính - ngân hàng:

Phân lớp dữ liệu có thể góp phần giải quyết các vấn đề kinh doanh trong lĩnh vực ngân hàng và tài chính bằng cách tìm các mẫu, quan hệ nhân quả và tương quan trong thông tin kinh doanh và giá thị trường mà không rõ ràng với người quản lý vì dữ liệu khối lượng quá lớn hoặc được tạo ra quá nhanh.

1.3. Tổng quan kỹ thuật xử lý dữ liệu mất cân bằng

Hiện nay có nhiều kỹ thuật khác nhau để giải quyết vấn đề về phân lớp đối với các bộ dữ liệu mất cân bằng,. Tuy nhiên, có thể phân chia các kỹ thuật thành hai hướng tiếp cận chính: hướng tiếp cận ở mức độ dữ liệu và hướng tiếp cận ở mức độ thuật toán [16].

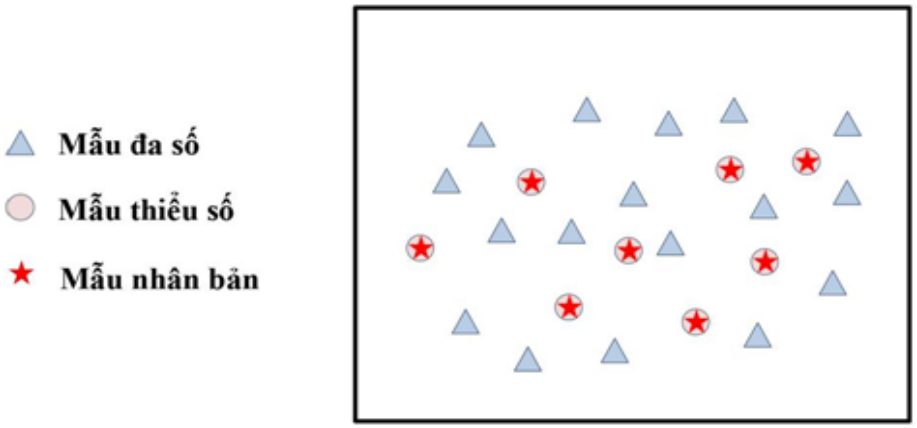
1.3.1. Hướng tiếp cận ở mức độ dữ liệu

Tiếp cận ở mức độ dữ liệu có mục tiêu điều chỉnh tỉ lệ mất cân bằng giữa hai lớp trong bộ dữ liệu. Các phương pháp ở hướng tiếp cận này có nhiều hình thức khác nhau của việc lấy mẫu như: sinh thêm các phần tử lớp thiểu số (sinh ngẫu nhiên, sinh thêm phần tử nhân tạo, …), loại bỏ các phần tử lớp đa số, hoặc kết hợp cả hai phương pháp trên.

- *Sinh thêm phần tử lớp thiểu số*

Hiện nay, có nhiều phương pháp sinh thêm phần tử cho lớp lớp thiểu số như: sinh ngẫu nhiên phần tử lớp thiểu số, lựa chọn phần tử lớp thiểu số, hay sinh thêm mẫu nhân tạo.

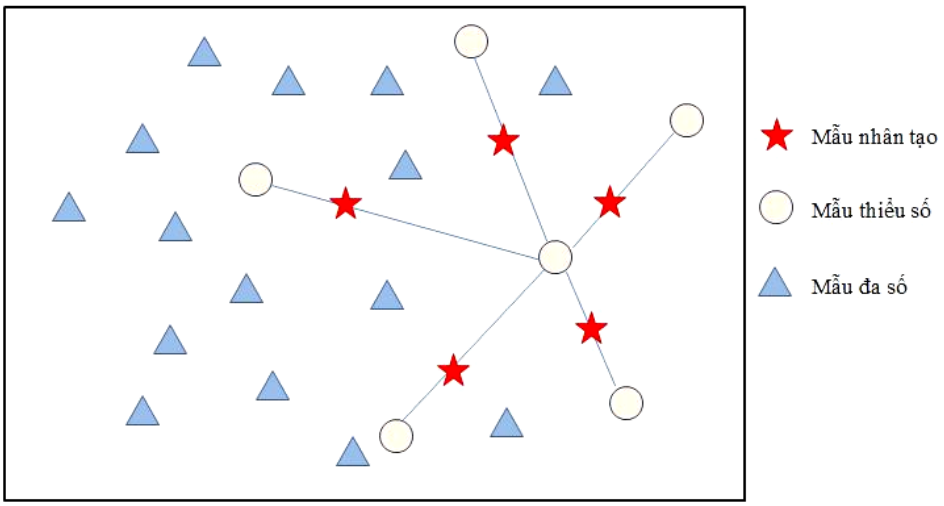
Sinh ngẫu nhiên các phần tử ở lớp thiểu số (Random Over-sampling) là phương pháp đơn giản nhất nhằm cân bằng phân lớp thông qua việc nhân bản ngẫu nhiên các mẫu lớp thiểu số. Ý tưởng của phương pháp này là lựa chọn ngẫu nhiên các mẫu thuộc lớp thiểu số và nhân bản chúng tạo ra mẫu mới giống hệt chúng.



Hình 1.7 Phương pháp sinh ngẫu nhiên phần tử lớp thiểu số

Với phương pháp sinh thêm mẫu nhân tạo lớp thiểu số, SMOTE (Synthetic Minority Over-sampling Technique) là thuật toán khá phổ biến. Ý tưởng của thuật toán SMOTE là: với mỗi mẫu thuộc lớp thiểu số tìm láng giềng gần nhất của nó trong lớp thiểu số, lựa chọn ngẫu nhiên các láng giềng gần nhất (hoặc tất cả láng giềng) tùy theo số lượng mẫu cần sinh thêm. Mẫu nhân tạo sẽ được sinh ra theo cách sau: lấy độ lệch giữa vector thuộc tính của mẫu đang xét và láng giềng của nó nhân với một số ngẫu nhiên trong khoảng rồi cộng kết quả thu được với vector thuộc tính của mẫu đang xét. Kết quả cuối cùng chính là vecto thuộc tính của mẫu nhân tạo, nhãn của mẫu nhân tạo sẽ được gán là nhãn của lớp thiểu số [10].

Hình 1.4 dưới đây mô tả quá trình sinh thêm thêm phần tử nhân tạo bằng thuật toán SMOTE [10].



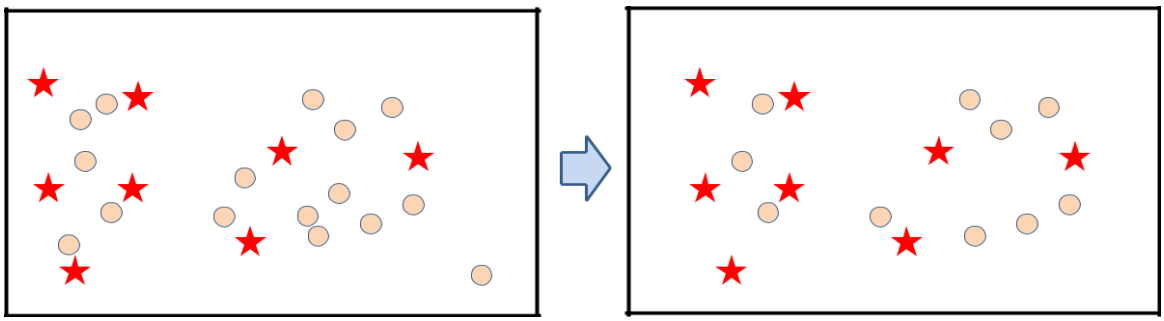
Hình 1.8 Sinh thêm phần tử nhân tạo bằng thuật toán SMOTE

Ngoài ra còn có một số thuật toán được cải tiến từ thuật toán SMOTE như: Borderline-SMOTE [13], Safe-level SMOTE[11] cũng đem lại những hiệu quả nhất định hỗ trợ quá trình phân lớp cho các bộ dữ liệu mất cân bằng.

- *Loại bỏ phần tử lớp đa số*

Loại bỏ phần tử lớp đa số là phương pháp điều chỉnh phân bố dữ liệu bằng cách giảm bớt số lượng phần tử lớp đa số.

Loại bỏ một cách ngẫu nhiên các mẫu thuộc lớp đa số (Random under - sampling) là cách đơn giản nhất. Phương pháp này thực hiện loại bỏ ngẫu nhiên phần tử thuộc lớp đa số trong tập huấn luyện cho tới khi có được tỷ lệ phù hợp giữa hai lớp. Vì lý do này, số lượng phần tử trong tập huấn luyện giảm đáng kể.



Hình 1.9 Loại bỏ phần tử lớp đa số

Tuy nhiên, việc loại bỏ mẫu có thể sẽ làm hao hụt thông tin và có khả năng làm mất đi những mẫu mang thông tin hữu ích hoặc thông tin quan trọng cho quá trình xây dựng mô hình phân lớp.

Khắc phục hạn chế của phương pháp trên, một số phương pháp loại bỏ mẫu theo mục tiêu được đề xuất như: Tomek links, One - side Selection, Neighborhood Cleaning Rule [14].

Ví dụ trong thuật toán Tomek links, cho hai mẫu E_i và E_j thuộc hai lớp khác nhau, và d(E_i, E_j) là khoảng cách giữa E_i và E_j. Một cặp (E_i, E_j) được gọi là Tomek link nếu như không có mẫu E_1 mà d(E_i, E_1) < d(E_j, E_1) hoặc d(E_j, E_1) < d(i_j, E_j). Nếu hai mẫu tạo thành một Tomek links thì một trong hai mẫu là nhiễu hoặc cả hai là phần tử biên. Tomek links có thể được sử dụng như một phương pháp loại bỏ mẫu khi chỉ các mẫu lớp đa số bị loại bỏ, hoặc như một phương pháp làm sạch dữ liệu khi mẫu bị loại bỏ thuộc cả hai lớp.

Ngoài việc sử dụng các tiêu chí đánh giá khác nhau, người ta cũng có thể làm việc để có được tập dữ liệu khác nhau. Hai cách tiếp cận để tạo ra một tập dữ liệu cân bằng trong một tập dữ liệu không cân bằng là under - sampling và over - sampling [12].

* *Under - sampling*

Việc lấy mẫu làm cân bằng tập dữ liệu bằng cách giảm kích thước của lớp trội. Phương pháp này được sử dụng khi số lượng dữ liệu là đủ. Bằng cách giữ tất cả các mẫu trong lớp hiếm và chọn ngẫu nhiên một số trong lớp trội, một tập dữ liệu cân bằng mới có thể được lấy ra để lập mô hình tiếp theo.

* *Over - sampling*

Ngược lại, over - sampling được sử dụng khi số lượng dữ liệu không đủ. Nó cố gắng cân bằng số liệu bằng cách tăng kích thước của các mẫu hiếm. Thay vì loại bỏ các mẫu phong phú, các mẫu hiếm mới được tạo ra bằng cách sử dụng ví dụ: lặp lại, bootstrapping hoặc SMOTE (Synthetic Minority Over - Sampling Technique). Lưu ý rằng không có lợi thế tuyệt đối của một phương pháp resampling hơn một

phương pháp khác. Áp dụng hai phương thức này phụ thuộc vào trường hợp sử dụng mà nó áp dụng cho và tập dữ liệu chính nó.

1.3.2. Hướng tiếp cận ở mức độ thuật toán

Như đã biết, để xây dựng các bộ phân lớp dữ liệu giải quyết các bài toán ứng dụng, thường sẽ sử dụng các thuật toán học máy [15] . Các phương pháp phân lớp dữ liệu tiêu biểu dựa trên kỹ thuật học máy có thể kể đến bao gồm:

- Phương pháp Cây quyết định.
- Phương pháp Bayes (Suy luận Bayes, mạng bayes).
- Phương pháp Máy vectơ hỗ trợ (SVM).
- Phương pháp Mạng no-ron nhân tạo (Artificial Neural Network - ANN).

Tuy nhiên, khi áp dụng trực tiếp các thuật toán học máy trong phân lớp dữ liệu mất cân bằng thường cho kết quả không như mong muốn. Vì vậy, cần có cải tiến các thuật toán học máy phù hợp cho bài toán phân lớp dữ liệu đối với các dữ liệu mất cân bằng.

Tiếp cận ở mức độ thuật toán nghĩa là điều chỉnh các thuật toán phân lớp để tăng cường độ chính xác khi phân lớp đối với dữ liệu mất cân bằng. Chiến lược chung để đối phó với vấn đề mất cân bằng trong các bộ dữ liệu là lựa chọn một khuynh hướng quy nạp thích hợp.

Ví dụ như đối với phương pháp cây quyết định, cách tiếp cận có thể là điều chỉnh dự đoán xác xuất ở lá, hoặc phát triển phương pháp cắt tỉa mới. Hay đối với phương pháp phân lớp SVM, có thể sử dụng hằng số phạt khác nhau cho các lớp hoặc điều chỉnh ranh giới lớp dựa trên ý tưởng liên hết hạt nhân [14].

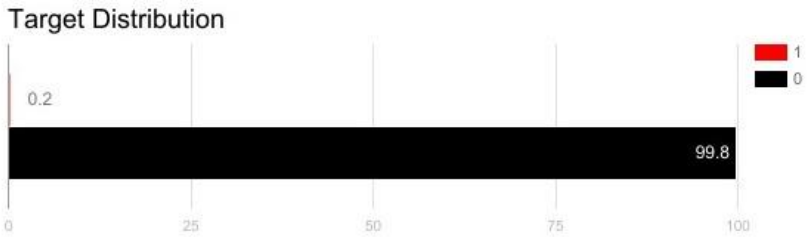
Đối với phương pháp phân lớp K-NN, có thể đề xuất một hàm khoảng cách có trọng số. Ý tưởng đằng sau khoảng cách có trọng số nhằm bù đắp cho sự mất cân bằng trong mẫu huấn luyện mà không thực sự làm thay đổi sự phân lớp.

Để minh họa cho cách tiếp cận ở mức độ thuật toán. có thể xem xét bộ dữ liệu trong các lĩnh vực như: phát hiện gian lận trong ngân hàng, đấu thầu thời gian thực trong tiếp thị, phát hiện xâm nhập trong mạng… Dữ liệu được sử dụng trong

các lĩnh vực này thường có dưới 1% các sự kiện hiếm hoi, nhưng "thú vị" (ví dụ: những kẻ lừa đảo sử dụng thẻ tín dụng, người dùng nhấp vào quảng cáo hoặc máy chủ bị hỏng quét mạng của nó). Khi đó, hầu hết các thuật toán học máy không hoạt động tốt với các bộ dữ liệu không cân bằng này. Cần phải có các kỹ thuật bổ sung để phân loại tốt các dữ liệu mất cân bằng.

Các kỹ thuật sau đây có thể giúp đào tạo một bộ phân loại để phát hiện ra lớp bất thường.

1.3.2.1. Sử dụng các chỉ số đánh giá phù hợp



Hình 1.10 Biểu đồ mô tả dữ liệu mất cân bằng

Áp dụng các chỉ số đánh giá không phù hợp cho mô hình được tạo bằng cách sử dụng dữ liệu mất cân bằng có thể nguy hiểm. Ví dụ dữ liệu đào tạo là dữ liệu được minh họa trong biểu đồ ở trên. Nếu độ chính xác được sử dụng để đo lường độ tốt của mô hình, mô hình phân loại tất cả các mẫu thử thành "0" sẽ có độ chính xác tuyệt vời (99,8%), nhưng rõ ràng, mô hình này sẽ không cung cấp bất kỳ thông tin giá trị nào cho chúng ta. Trong trường hợp này, các chỉ số đánh giá thay thế khác có thể được áp dụng như:

- Độ chính xác / độ đặc hiệu: bao nhiêu trường hợp được chọn có liên quan. Nhớ lại / Độ nhạy: bao nhiêu trường hợp có liên quan được chọn.

- Điểm số F1: trung bình hài hòa của độ chính xác và thu hồi.

- MCC: hệ số tương quan giữa phân loại nhị phân được quan sát và được dự đoán.

- AUC: mối quan hệ giữa tỷ lệ thực dương và tỷ lệ dương tính giả.

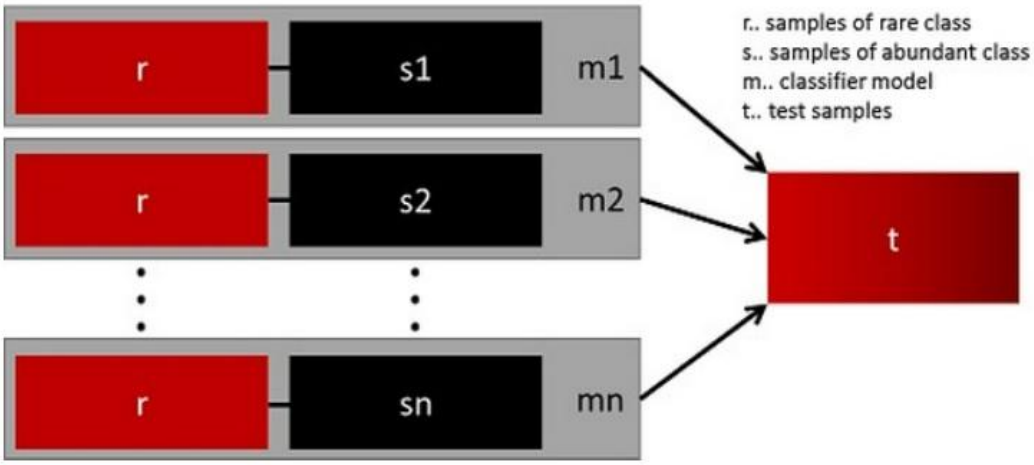
1.3.2.2. Sử dụng K - fold Cross - Validation đúng cách

Đáng chú ý là cross - validation phải được áp dụng đúng cách trong khi sử dụng phương pháp over - sampling để giải quyết các vấn đề mất cân đối. Hãy ghi

nhớ rằng over - sampling phải quan sát các mẫu hiếm và áp dụng bootstrapping để tạo ra dữ liệu ngẫu nhiên mới dựa trên một hàm phân phối. Nếu cross - validation được áp dụng sau khi over - sampling, về cơ bản những gì chúng ta đang làm là đưa mô hình của chúng ta bị overfitting trên một kết quả bootstrapping đặc biệt. Đó là lý do tại sao cross - validation phải luôn được thực hiện trước khi over - sampling, cũng giống như cách lựa chọn tính năng được thực hiện. Chỉ bằng cách resampling dữ liệu nhiều lần, ngẫu nhiên có thể được đưa vào tập dữ liệu để đảm bảo rằng sẽ không bị vấn đề overfitting.

1.3.2.3. Tập hợp các tập dữ liệu được lấy mẫu khác nhau

Cách dễ nhất để khái quát hóa mô hình thành công là sử dụng nhiều dữ liệu hơn. Vấn đề là các bộ phân loại out - of - the - box như hồi quy logistic hoặc rừng ngẫu nhiên có xu hướng tổng quát hóa bằng cách loại bỏ lớp hiếm. Một thực hiện tốt nhất là xây dựng n mô hình sử dụng tất cả các mẫu của các mẫu hiếm và n - khác biệt của lớp phong phú. Do bạn muốn tập hợp 10 mô hình, bạn sẽ giữ nguyên ví dụ: 1.000 trường hợp lớp hiếm và lấy mẫu ngẫu nhiên 10.000 trường hợp của lớp phong phú. Sau đó, bạn chỉ cần chia 10.000 trường hợp trong 10 khối và đào tạo 10 mô hình khác nhau.



Hình 1.11 Minh họa tập hợp các tập dữ liệu được lấy mẫu

Cách tiếp cận này đơn giản và hoàn toàn có thể mở rộng theo chiều ngang nếu bạn có nhiều dữ liệu, vì bạn chỉ có thể đào tạo và chạy các mô hình của mình

trên các nút cụm khác nhau. Các mô hình kết hợp cũng có xu hướng khái quát hóa tốt hơn, khiến cho phương pháp này dễ xử lý.

1.3.2.4. Lấy mẫu với các tỷ lệ khác nhau

Cách tiếp cận trước đó có thể được tinh chỉnh bằng cách thay đổi tỷ lệ giữa lớp hiếm và phong phú. Tỷ lệ tốt nhất phụ thuộc nhiều vào dữ liệu và các mô hình được sử dụng. Nhưng thay vì đào tạo tất cả các mô hình với tỷ lệ tương tự nhau, có thể tổng hợp các tỷ lệ khác nhau. Vì vậy, nếu 10 mô hình được đào tạo, có thể điều chỉnh để một mô hình có tỷ lệ 1:1 (hiếm: phong phú) và một mô hình khác với 1:3, hoặc thậm chí 2:1. Tùy thuộc vào mô hình được sử dụng này có thể ảnh hưởng đến trọng lượng mà một lớp được.

1.4. Kết luận chương 1

Chương I của luận văn đã giới thiệu về bài toán phân lớp dữ liệu nói chung và phân lớp dữ liệu mất cân bằng nói riêng cùng các vấn đề liên quan. Luận văn cũng khảo sát tổng quan về dữ liệu mất cân bằng, các đặc điểm của phân lớp dữ liệu mất cân bằng.

Từ đó, luận văn đã khảo sát hai hướng tiếp cận giải quyết bài toán phân lớp dữ liệu mất cân bằng là tiếp cận ở mức độ dữ liệu và tiếp cận ở mức độ thuật toán.

Trong chương tiếp theo luận văn sẽ nghiên cứu một số thuật toán phân lớp trên dữ liệu mất cân bằng..

CHƯƠNG 2. MỘT SỐ THUẬT TOÁN PHÂN LỚP DỮ LIỆU MẤT CÂN BẰNG

Trong chương 2 luận văn sẽ khảo sát một số thuật toán để giải quyết bài toán phân lớp dữ liệu mất cân bằng. Các thuật toán được khảo sát bao gồm: thuật toán DEC - SVM, thuật toán HMU, thuật toán HBU và thuật toán RBU.

2.1. Thuật toán DEC - SVM

2.1.1. Giới thiệu thuật toán

Thuật toán máy vector hỗ trợ (Support Vector Machines - SVM) thường được sử dụng xây dựng các bộ phân lớp dữ liệu.

Thuật toán SVM được Cortes và Vapnik giới thiệu vào năm 1995 trên cơ sở mở rộng từ chuyên đề lý thuyết học thống kê (Vapnik 1982), dựa trên nguyên tắc tối thiểu rủi ro cấu trúc (structural risk minimization). Ý tưởng chính của SVM để giải quyết bài toán phân lớp dữ liệu là ánh xạ tập dữ liệu mẫu thành các vector điểm trong không gian vector R^d và tìm các siêu phẳng có hướng để chia tách chúng thành các lớp khác nhau.

Định lý 2.1 sau đây đảm bảo cơ sở toán học cho SVM [7].

Định lý 2.1: Cho tập hợp gồm m điểm trong không gian R^d. Ta chọn một điểm nào đó trong chúng làm điểm gốc và tạo thành m-1 vector điểm. Khi đó m điểm đã cho có thể được phân tách bởi một siêu phẳng có hướng khi và chỉ khi tập hợp các vector điểm là độc lập tuyến tính.

Khoảng cách của điểm dữ liệu gần nhất của mỗi lớp đến siêu phẳng phân tách gọi là biên (hay lề). Trong số các siêu phẳng thỏa mãn định lý 2.1, siêu phẳng tối ưu có biên lớn nhất sẽ được lựa để phân tách các điểm. Các kỹ thuật SVM nhằm nghiên cứu xây dựng các siêu phẳng tối ưu này một cách hiệu quả nhất.

Ưu điểm nổi bật của phương pháp SVM là thực hiện tối ưu toàn cục cho mô hình phân lớp. Do đó, mô hình SVM có chất lượng cao, chịu đựng được nhiễu. Mặt khác, SVM là một phương pháp tốt (phù hợp) đối với những bài toán phân lớp có

không gian biểu diễn thuộc tính lớn. Các đối tượng cần phân lớp được biểu diễn bởi một tập rất lớn các thuộc tính.

Tuy nhiên khi áp dụng trực tiếp thuật toán SVM cho phân lớp dữ liệu mất cân bằng có thể không đạt được kết quả mong muốn. Lý do là SVM thường có xu hướng thiên vị đối với lớp đa số và bỏ qua lớp thiểu số (xử lý chúng như là nhiễu) [8]. Việc phân loại sai các mẫu thuộc lớp thiểu số có thể gây nên những tổn thất lớn đối với các bài toán thực tế.

Để khắc phục vấn đề trên, phương pháp sinh thêm phần tử nhân tạo cho lớp thiểu số là phương pháp khá phổ biến thường được sử dụng. Tuy nhiên, trong nhiều trường hợp, việc sinh thêm mẫu có thể sẽ tạo ra những mẫu dư thừa hoặc nhiễu làm ảnh hưởng tới hiệu quả phân lớp.

Do đó, trước khi sử dụng SVM cần áp dụng thuật toán DEC (a novel Differential Evolution Clustering hybrid resampling- DEC) [9] để điều chỉnh dữ liệu cho bài toán phân lớp dữ liệu mất cân bằng. Thuật toán DEC là sự kết hợp giữa phương pháp sinh thêm phần tử cho lớp thiểu số và sử dụng kỹ thuật phân cụm K-means để loại bỏ bớt phần tử dư thừa, nhiễu trong dữ liệu. Với mỗi mẫu thuộc lớp thiểu số, tạo ra một mẫu đột biến từ hai trong số những láng giềng gần nó nhất, sau đó sử dụng thuật toán di truyền để sinh thêm phần tử cho lớp thiểu số từ mẫu thiểu số ban đầu và mẫu đột biến mới tạo ra. Sau khi điều chỉnh dữ liệu bằng thuật toán DEC, thuật toán SVM sẽ được sử dụng để xây dựng mô hình phân lớp.

Như vậy, thuật toán DEC-SVM có thể xem như sự kết hợp giữa hai thuật toán DEC và SVM. Nội dung trình bày trong mục này tham khảo từ tài liệu [4].

2.1.2. Khảo sát nội dung thuật toán

2.1.2.1. Điều chỉnh dữ liệu bằng thuật toán DE (Differential Evolution over-sampling)

Với thuật toán SMOTE, mẫu mới sẽ được sinh ra từ một mẫu positive ban đầu và một trong những láng giềng của nó. Với nền tảng là thuật toán SMOTE, tuy nhiên, trong thuật toán DE, từ hai trong số các láng giềng gần nhất của một mẫu

positve sẽ tạo ra một mẫu "đột biến", và mẫu mới được sinh ra bằng cách lai ghép chéo mẫu đột biến này và mẫu positive ban đầu.

- *Đột biến*

Trong tập dữ liệu huấn luyện, đầu tiên chọn ngẫu nhiên một mẫu positive x_i và tìm k láng giềng gần nhất của nó, sau đó chọn ngẫu nhiên hai láng giềng trong k láng giềng đó: x_{n1} và x_{n2} . Một mẫu đột biến x_{mu} sẽ được tạo ra bằng cách sử dụng công thức (1) với *rand(0,1)* là hằng số ngẫu nhiên trong khoảng [0,1]:

$$x_{mu} = x_i + rand(0,1) \times (x_{n1} - x_{n2}) \quad (1)$$

- *Crossover:*

Qua bước đột biến, ta tạo ra số lượng mẫu đột biến đúng bằng số lượng mẫu positive ban đầu trong tập dữ liệu huấn luyện. Ở bước này, ta sẽ sử dụng các mẫu đột biến cùng với các mẫu positive ban đầu để tạo ra mẫu nhân tạo mới.

Cụ thể, các mẫu mới sẽ được hình thành dựa theo (2):

$$x_{new,j} = \begin{cases} x_{i,j} & if\ (rand(j) > CR)\ and\ j \neq rand(s) \\ x_{mu,j} & if\ (rand(j) < CR)\ or\ j = rand(s) \end{cases} \quad (2)$$

Trong đó $x_{i,j}$ đại diện cho thuộc tính thứ j của mẫu thứ i

CR là hằng số crossover được lựa chọn ngẫu nhiên trong [0, 1] và được xác định trước bởi người dùng.

rand(j) là giá trị được lựa chọn ngẫu nhiên trong khoảng [0, 1].

Giá trị của biến *rand(s)* là chỉ số của các thuộc tính được lấy một cách ngẫu nhiên, đảm bảo rằng mẫu mới sinh ra sẽ có ít nhất một thuộc tính từ mẫu đột biến.

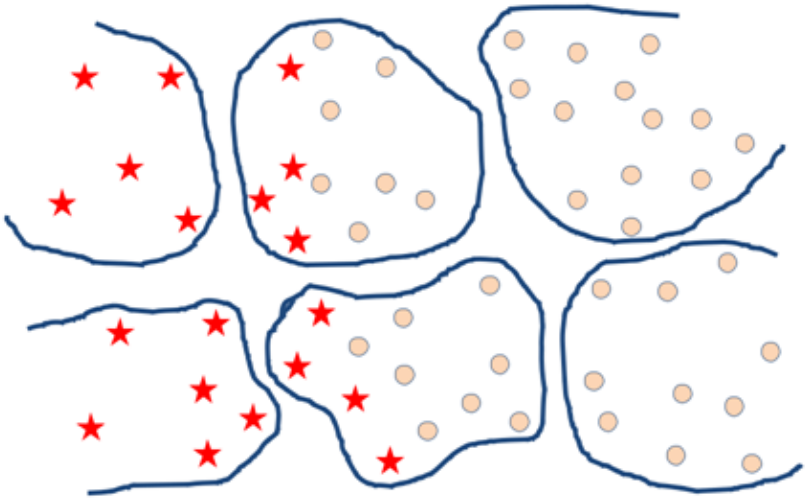
Số mẫu nhân tạo được tạo ra đúng bằng số mẫu nhân tạo ban đầu. Tùy thuộc vào số lượng mẫu possitive cần lấy, ta sẽ lặp lại các bước đột biến và crossover cho dữ liệu huấn luyện.

2.1.2.2. Kỹ thuật làm sạch dữ liệu sử dụng phân cụm

Sau khi thực hiện thuật toán DE, dữ liệu thu được đã được cải thiện hơn về tỉ lệ giữa hai lớp. Tuy nhiên, không loại trừ khả năng sinh ra những mẫu dư thừa hoặc

nhiễu. Để khắc phục, ta sẽ sử dụng kỹ thuật phân cụm để phân cụm cho tập dữ liệu với mục đích loại bỏ những mẫu không cần thiết.

Chẳng hạn ta thu được các cụm và giả sử được đặt tên là A, B, C, D, E, F như Hình 2.. Trong đó, một số cụm chứa tất cả các mẫu có cùng nhãn lớp (các cụm C, D, E và F), những cụm khác chứa các mẫu có nhãn lớp khác nhau (cụm A và B), dự đoán rằng có thể siêu phẳng của SVM sẽ đi qua các cụm này.



Hình 2.1 Minh họa phân cụm tập dữ liệu mất cân bằng

Nếu như tất cả các mẫu trong một cụm đều có cùng một nhãn lớp (tức là hoặc cùng là positive hoặc cùng là negative), ta sẽ tiến hành loại bỏ những mẫu dư thừa hoặc nhiễu. Ví vụ với cụm F có chứa tất cả các mẫu negative, ta sẽ thực hiện theo những bước sau:

- Xác định ngưỡng tương đồng trong [0,1]
- Tính $\bar{x}$ theo công thức (3)

$$\bar{x} = \frac{1}{n_i}\sum_{j=1}^{n_i} x_i \qquad (3)$$

- Tìm mẫu trung tâm $\bar{x}_c \in F$ gần $\bar{x}$ nhất
- Tính độ tương đồng S_{ic} giữa mỗi đầu $x_i \in F$ và $\bar{x}_c$ theo (4). Nếu S_{ic} lớn hơn ngưỡng tương đồng thì x_i sẽ bị loại khỏi F

$$S_{ic} = \frac{\sum_{k=1}^{n} x_{ik^*} x_{jk}}{\sqrt{(\sum_{k=1}^{n} x^2{}_{ik})(\sum_{k=1}^{n} x^2{}_{jk})}} \qquad (4)$$

Ngưỡng tương đồng càng nhỏ thì càng nhiều mẫu bị loại bỏ

Trong đó: n_i là số lượng mẫu trong cụm thứ i, x_{ik} là thuộc tính thứ k của mẫu x_i, S_{ij} là độ tương đồng giữa x_i và x_j

2.1.2.3. Thuật toán DEC-SVM

Sau khi sử dụng thuật toán DEC để điều chỉnh dữ liệu, ta sử dụng thuật toán SVM để phân lớp cho cho tập dữ liệu huấn luyện tạo nên một mô hình phân lớp.

Nội dung thuật toán DEC-SVM có thể trình bày dưới dạng giả mã của như sau [4]:

DEC-SVM(N, m, K, s, T)

Input: Số mẫu lớp thiểu số N, số thuộc tính m, số cụm K, ngưỡng tương đồng s, số lượng của DE là T%.

Output: Mô hình huấn luyện

```
Void DEC-SVM() {
/******** Sinh thêm mẫu bằng DE ********/
st = 0;
G = int(N*T%); //số mẫu lớp thiểu số được tạo ra
For (t = 0; t< ceil(T/100); t++) {
    For (i=0; i<N; i++) {
     /*Đánh giá khoảng cách giữa tất cả các mẫu thiểu số*/
        For (p=0; p<N; p++) {
            For (j=0; j<m; j++){
            Euclidean_distance[i][j] += pow((x[i][j] - x[p][j]), 2);
            }
        }
        Tìm hai mẫu x(n1) và x(n2) trong k láng giềng gần nhất của x(i);
        For(j = 0; j < m; j++) {//Đột biến và crossover của DE
```

```
                    If(rand_j > CR && j != rand_s) {
                    x[N+st][j] = x[i][j];
                    }
                    Else if(rand_j <=CR || j == rand_s){
                    x[N+st][j] = x[i][j] + rand(0,1)(x[n1][j] - x[n2][j]);
                    }
                }
                st++;
                If (st==G) break;
            }
            If (st==G) break;
        }
/********Làm sạch dữ liệu bằng phân cụm********/
        Phân cụm tập dữ liệu thành K cụm;
        Để lại những cụm có các mẫu mang nhãn hỗn hợp
            For(t = 0; t < K; t++) {
            /* Tính trung bình của các cụm mà tất cả các mẫu có cùng nhãn*/
            If (cụm t có tất cả các mẫu cùng nhãn){
            n = số mẫu trong cụm t;
                For(j = 0; j < m; j++) {
                    For(i = 0; i < n; i++) {
                        Mean[j] += x[i][j];
                    }
                    Mean[j] = Mean[j] / n;
                }
                Tìm mẫu trung tâm x[p] của cụm t;
                For (i = 0; i < n; i++) {//Loại bỏ các mẫu dư thừa
                    Tính độ tương đồng s[i][p];
                    If (s[i][p] >s) Loại bỏ x[i];
```

```
            }
        }
        }
/********Phân lớp bằng SVM********/
SVM training;
SVM prediction;
}
```

2.1.3. Đánh giá thuật toán

Thuật toán DEC-SVM có thể chia làm hai pha: pha tiền xử lý dữ liệu và pha phân lớp dữ liệu.

Pha tiền xử lý dữ liệu gồm hai bước:

Bước 1: Điều chỉnh dữ liệu bằng thuật toán DE.

Bước 2: Làm sạch dữ liệu sử dụng phân cụm.

Pha phân lớp dữ liệu sử dụng thuật toán SVM.

Với số lượng mẫu lớp thiểu là N, số thuộc tính m, số cụm K, ngưỡng tương đồng s, số lượng của DE là T% thì độ phức tạp tính toán của bước điều chỉnh dữ liệu bằng thuật toán DE là $O(N^3*m)$.

Với n là số mẫu của cụm thứ t trong K cụm cần phân cụm thì độ phức tạp tính toán của bước làm sạch dữ liệu sử dụng phân cụm là $O(N^2*m*K)$ vì n không thể vượt quá N.

Như vậy, thuật toán DEC-SVM so với thuật toán SVM chỉ bổ sung thêm phần tiền xử lý dữ liệu với độ phức tạp tính toán công thêm là $O(N^3*m)$.

So sánh với một số thuật toán cùng hướng tiếp cận như thuật toán sinh thêm mẫu nhân tạo lớp thiểu số (SMOTE) bằng cách kết hợp thuật toán nhúng tuyến tính cục bộ (locally linear embedding - LLE) rồi sử dụng SVM có độ phức tạp tính toán tương đồng [8].

2.2. Thuật toán HMU

2.2.1. Giới thiệu thuật toán

Phương pháp sinh phần tử ngẫu nhiên *(Random Oversampling)* là phương pháp sinh thêm phần tử đơn giản nhất bằng cách tăng số lượng một số phần tử được chọn ngẫu nhiên thuộc lớp thiểu số để cân bằng tỷ lệ. Tuy nhiên, kỹ thuật này có nhược điểm là dễ dẫn đến tình trạng quá khớp với dữ liệu huấn luyện *(overfitting).* Ngoài ra, nếu tập dữ liệu có kích thước lớn thì chi phí thời gian và bộ nhớ cho giai đoạn phân lớp sẽ gia tăng đáng kể.

Trái lại, phương pháp giảm số phần tử ngẫu nhiên *(Random Undersampling)* sẽ chọn ngẫu nhiên và loại bỏ một số phần tử thuộc lớp đa số để làm giảm tỷ lệ mất cân bằng của các tập dữ liệu. Phương pháp này tuy tốn ít chi phí về thời gian cũng như bộ nhớ cho quá trình phân lớp nhưng lại dễ làm mất các thông tin quan trọng của lớp đa số.

Để giải quyết các vấn đề trên trong mục này luận văn sẽ khảo sát thuật toán HMU (Hypothesis Margin based Undersampling - HMU) dựa trên tài liệu [1].

Ý tưởng của thuật toán HMU là làm giảm số phần tử thuộc lớp đa số mới nhắm tới xử lý các đối tượng khó phân lớp và khắc phục nhược điểm đã đề cập ở trên. Trước hết, cần phân tích thêm một số độ đo đánh giá hiệu suất phân lớp đã trình bày ở chương 1.

Do các tập dữ liệu là không cân bằng, việc sử dụng độ đo accuracy làm cơ sở để đánh giá hiệu suất phân lớp sẽ không thể hiện được hết yêu cầu đặt ra là dự đoán cả hai nhãn lớp cần đạt được độ chính xác cao. Vì vậy, các độ đo khác thích hợp hơn thường được sử dụng làm độ đo hiệu suất của việc phân lớp, như: Sensitivity, specificity, precision, F-measure, G-mean

$$\text{Sensitivity} = \text{Recal} = \frac{TP}{TP + FN} \tag{2.1}$$

$$\text{Specificity} = \frac{TN}{TN + FP} \quad (2.2)$$

$$\text{Precision} = \frac{TP}{TP + FP} \quad (2.3)$$

$$\text{F - measure} = \frac{(1 + \beta^2).\text{Precision}.\ \text{Recall}}{\beta^2.\text{Precision} + \text{Recall}} \quad (2.4)$$

Trong đó, β là hệ số điều chỉnh mối quan hệ giữa Precision với Recall và thông thường β =1. F-measure thể hiện sự tương quan hài hòa giữa Precision và Recall. Giá trị của F- measure cao khi cả Precision và Recall đều cao.

G-mean là sự kết hợp của Sensitivity và Specificity, được tính bởi công thức:

$$\text{G - mean} = \sqrt{Sensitivity \times Specificity} \quad (2.5)$$

TP và TN lần lượt là số phần tử thuộc lớp thiểu số và lớp đa số được dự đoán đúng với nhãn lớp thực sự của chúng; FN và FP lần lượt là số phần tử thuộc lớp thiểu số và lớp đa số bị dự đoán sai nhãn lớp so với nhãn lớp thực sự của chúng.

Trong thuật toán HMU sử dụng F-measure và G-mean làm độ đo chính để điều chỉnh quá trình huấn luyện cho mô hình phân lớp dữ liệu mất cân bằng.

2.2.2. Khảo sát nội dung thuật toán

2.2.2.1. Phân loại lề

Lề (margin), đóng vai trò quan trọng trong lĩnh vực học máy, thể hiện tính hiệu quả khi phân lớp của bộ phân lớp (classifier).

Có hai cách xác định giá trị lề cho một phần tử dựa trên quy tắc phân lớp [10]. Cách thứ nhất là đo khoảng cách từ phần tử đang xét tới biên quyết định được xác định bởi bộ phân lớp và lề trong trường hợp này gọi là lề phần tử (sample margin). Đối với cách thứ hai, lề là khoảng cách mà bộ phân lớp có thể di chuyển sao cho không làm thay đổi nhãn lớp của các phần tử đã được xác định, và được gọi là lề giả thuyết (hypothesis margin).

Trong trường hợp sử dụng bộ phân lớp láng giềng gần nhất, các kết quả sau đây được công nhận [7]:

- Lề giả thuyết là giới hạn dưới của lề phần tử.
- Lề giả thuyết của phần tử x trong tập dữ liệu A được tính bởi công thức:

$$\theta_A = \frac{1}{2}(\|x - \text{nearestmiss}_A(x)\| - \|x - \text{nearesthit}_A(x)\|) \quad (2.6)$$

trong đó: $\text{nearesthit}_A(x)$ là phần tử gần nhất có cùng nhãn lớp với x trong A. $\text{nearestmiss}_A(x)$ là phần tử gần nhất khác nhãn lớp với x trong A.

Từ đó có thể suy ra, nếu một tập các phần tử có giá trị lề giả thuyết lớn thì giá trị lề phần tử tương ứng của nó cũng lớn.

Do đó, chúng ta có thể áp dụng kết luận này vào bài toán xử lý dữ liệu mất cân bằng bằng phương pháp làm giảm bớt phần tử.

Giả sử phần tử x thuộc lớp đa số N được chọn để loại bỏ, lúc đó, lề giả thuyết của các phần tử y trong tập dữ liệu A sẽ là:

$$\theta_{A\setminus\{x\}}(y) = \frac{1}{2}(\|y - \text{nearestmiss}_{A\setminus\{x\}}(y)\| - \|y - \text{nearesthit}_{A\setminus\{x\}}(y)\|), \forall y \neq x$$

Ở đây, $\text{nearestmiss}_A(y)$, $\text{nearesthit}_A(y)$ lần lượt là phần tử gần nhất khác nhãn lớp và phần tử gần nhất cùng nhãn lớp của y trên tập A.

Nếu y_p thuộc vào lớp thiểu số P, thì:

$$\|y_p - \text{nearesthit}_{A\setminus\{x\}}(y_p)\| = \|y_p - \text{nearesthit}_A(y_p)\|$$

Và $$\|y_p - \text{nearestmiss}_{A\setminus\{x\}}(y_p)\| \geq \|y_p - \text{nearestmiss}_A(y_p)\|$$

Do đó: $$\theta_{A\setminus\{x\}}(y_p) \geq \theta_A(y_p).$$

Tương tự, với y_n là phần tử thuộc lớp đa số N, $y_n \neq x$, ta có:

$$\|y_n - \text{nearesthit}_{A\setminus\{x\}}(y_n)\| \geq \|y_n - \text{nearesthit}_A(y_n)\|$$

$$\|y_n - \text{nearestmiss}_{A\setminus\{x\}}(y_n)\| = \|y_n - \text{nearestmiss}_A(y_n)\|$$

Nên

$$\theta_{A\setminus\{x\}}(y_n) \leq \theta_A(y_n).$$

Điều này có nghĩa rằng việc loại bỏ đi một phần tử thuộc lớp đa số làm tăng giá trị lề của các phần tử lớp thiểu số và giảm giá trị lề của phần tử thuộc lớp đa số. Do đó, nếu các phần tử được chọn để loại bỏ có lề lớn hơn các phần tử còn lại sẽ làm tăng khả năng phân lớp sai của bộ phân lớp. Hay nói cách khác, việc chọn các phần tử có giá trị lề giả thuyết bé nhất thay vì chọn một cách ngẫu nhiên để loại bỏ sẽ làm tăng hiệu suất của việc phân lớp.

2.2.2.2. Thuật toán HMU

Thuật toán HMU dựa vào giá trị lề giả thuyết để làm giảm phần tử của lớp đa số. HMU ưu tiên chọn các phần tử có giá trị lề bé nhất để loại bỏ trước tiên nhằm tạo ra một tập dữ liệu dễ phân lớp hơn. Thuật toán HMU được mô tả dưới dạng giả code như sau [1]:

HMU Algorithm

***Input**: lớp đa số **N**; số lượng phần tử cần loại bỏ **d**;*

Output**: lớp đa số sau khi đã làm giảm số phần tử **N*;

Begin

1. *nos = |N| -d*
2. *N* = N*
3. *While (|N*| > nos)*
4. *Tính giá trị lề mar(x) của tất cả các phần tử x thuộc N* trên toàn bộ tập dữ liệu và lưu vào mảng @margin*
5. *Sắp xếp mảng @margin*
6. *Loại bỏ phần tử có giá trị lề tương ứng bé nhất trong mảng @margin*
7. *Cập nhật lại N**
8. *End while*

End

Ghi chú:

- Lề của các phần tử lớp đa số được tính dựa vào công thức (2.6)

- Kích thước của lớp đa số sau khi làm giảm bớt số phần tử N* được xác định dựa vào số lượng phần tử cần loại bỏ d. Chỉ số d này phụ thuộc vào từng tập dữ liệu cụ thể.

- Khoảng cách được sử dụng để xác định lề trong thuật toán này là khoảng cách Euclidean.

Sau khi sử dụng HMU có thể sử dụng các thuật toán học máy truyền thống như SVM để xây dựng các bộ phân lớp dữ liệu đối với các dữ liệu mất cân bằng.

Có thể đề xuất thuật toán phân lớp dữ liệu HMU-SVM như sau:

```
Void HMU-SVM() {
        Thực hiện HMU tiền xử lý dữ liêu;
                SVM training;
                SVM prediction;
        }
```

2.2.3. Đánh giá thuật toán

Thuật toán HMU ứng dụng trong pha tiền xử lý dữ liệu trong bài toán phân lớp dữ liệu mất cân bằng.

Với lớp đa số gồm N phần tử và số lượng phần tử cần loại bỏ từ lớp đa số là d thì độ phức tạp tính toán là $O(N^2*d)$.

So sánh với thuật toán DEC-SVM thì thuật toán HMU trong pha tiền xử lý dữ liệu có độ phức tạp tính toán nhỏ hơn. Tuy nhiên, do HMU làm giảm lớp đa số nên thường chỉ được áp dụng khi tập dữ liệu cân mất cân bằng có lớp thiểu số đủ lớn để đảm bảo hiệu suất phân lớp. Trong trường hợp lớp thiểu số quá thưa thì HMU sẽ không cho kết quả tốt.

2.3. Thuật toán HBU

2.3.1. Giới thiệu thuật toán

Trong mục 2.2 luận văn đã khảo sát thuật toán HMU. HMU đã làm tăng đáng kể hiệu quả của việc phân lớp. Tuy nhiên, trong một số trường hợp, khi các phần tử thuộc hai lớp đa số và thiểu số nằm gần nhau, đặc biệt là khi các phần tử lớp đa số phân tán xen giữa các phần tử lớp thiểu số, các phần tử này sẽ dễ bị phân lớp nhầm và thuật toán HMU sẽ không hoạt động tốt.

Trong mục này, luận văn khảo sát thuật toán HBU (Hypothesis margin based Borderline Under-sampling - HMU) nhằm khác phục những nhược điểm của HMU. Ý tưởng của thuật toán HBU là dựa vào giá trị lề giả thuyết và ưu tiên loại bỏ các phần tử nằm ở biên. Với mỗi phần tử lớp thiểu số x, một số lượng n các phần tử lớp đa số nằm gần x nhất sẽ được chọn để loại bỏ. Giá trị n thay đổi phụ thuộc vào giá trị lề của mỗi x khác nhau [2].

2.3.2. Khảo sát nội dung thuật toán

Dựa vào những phân tích ở phần trên, thuật toán HBU có thể mô tả dưới dạng giả code như sau [2]:

HBU Algorthim

Input: tập các phần tử lớp thiểu số P; tập các phần tử lớp đa số N; số lượng phần tử cần loại bỏ N' ; tham số k

Output: tập các phần tử lớp đa số mới N;*

1. *Begin*
2. *Tính giá trị lề mar(x) của tất cả các phần tử lớp thiểu số x trên tập dữ liệu đã cho*
3. $max = \max_{x \in P} mar(x)$
4. $min = \min_{x \in P} mar(x)$
5. $p = |P|$

6. *Foreach x in P*

7. ***nos****= int((N'/p)*(k +(max-mar(x))/(max-min)));*

8. *Loại bỏ* ***nos*** *phần tử lớp đa số mà gần với x nhất*

9. *N' = N' - nos*

10. *p = p - 1*

11. *End-for*

12. *End*

Ghi chú:

Lề của các phần tử lớp đa số được tính dựa vào công thức (2.6). Trong phần này, sử dụng khoảng cách Euclide để xác định giá trị lề cho các đối tượng.

Kích thước của lớp đa số sau khi làm giảm bớt số phần tử N* được xác định dựa vào số lượng phần tử cần loại bỏ N', giá trị này phụ thuộc vào từng tập dữ liệu cụ thể.

Tương tự HMU, trong pha tiền xử lý dữ liệu có thể sử dung HBU và sau đó sử dụng các thuật toán học máy truyền thống như SVM để xây dựng các bộ phân lớp dữ liệu đối với các dữ liệu mất cân bằng.

Có thể đề xuất thuật toán phân lớp dữ liệu HBU-SVM như sau:

```
Void HBU-SVM() {
        Thực hiện HBU tiền xử lý dữ liêu;
                SVM training;
                SVM prediction;
        }
```

2.3.3. Đánh giá thuật toán

Tương tự HMU, thuật toán HBU ứng dụng trong pha tiền xử lý dữ liệu trong bài toán phân lớp dữ liệu mất cân bằng. Độ phức tạp tính toán của HBU tương đồng với HMU.

So với thuật toán HMU, HBU khắc phục nhược điểm của HMU trong các bài toán phân lớp khi các phần tử thuộc hai lớp đa số và thiểu số nằm gần nhau.

So sánh với thuật toán DEC-SVM thì thuật toán HBU cũng có các đặc điểm tương tự đã trình bày trong mục 2.2.3 ở trên.

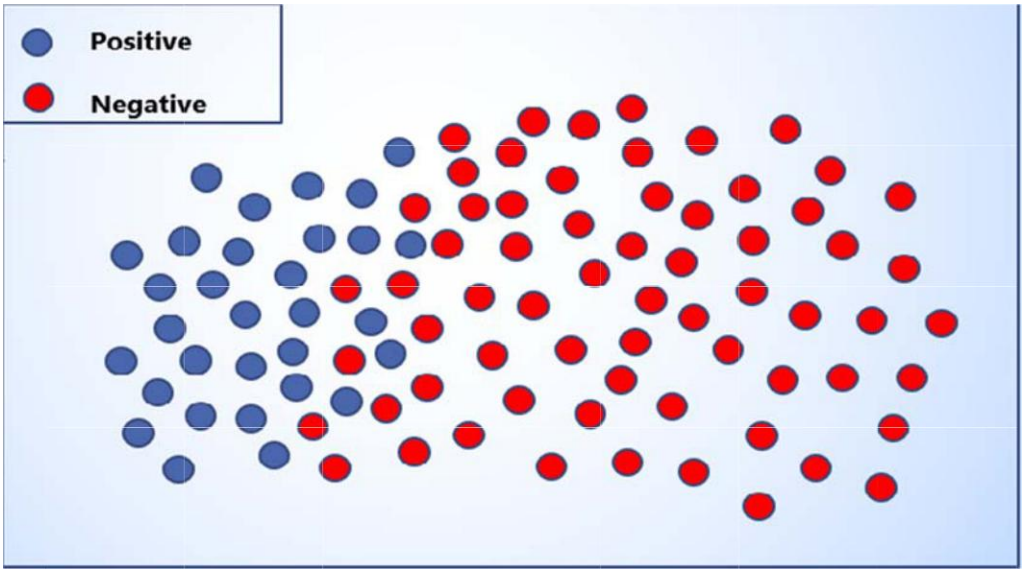
2.4. Thuật toán RBU

2.4.1. Giới thiệu thuật toán

Thuật toán RBU (Random Border Undersampling - RBU) được cải tiến từ thuật toán Random undersampling đã có sẵn, sử dụng việc giảm ngẫu nhiên phần tử trên đường biên. Để xác định được các phần tử trên đường biên, thuật toán xác định dựa vào số láng giềng là thuộc lớp thiểu số m trong tổng số k láng giềng gần nhất. Nếu $k/2 \leq m < k$ thì phần tử đó là phần tử biên [6].

Dưới đây luận văn mô tả về bộ dữ liệu mất cân bằng và quá trình xác định phần tử lớp đa số thuộc đường biên để tiến hành giảm bớt các phần tử đó theo tỉ lệ phần trăm dựa theo tài liệu [6].

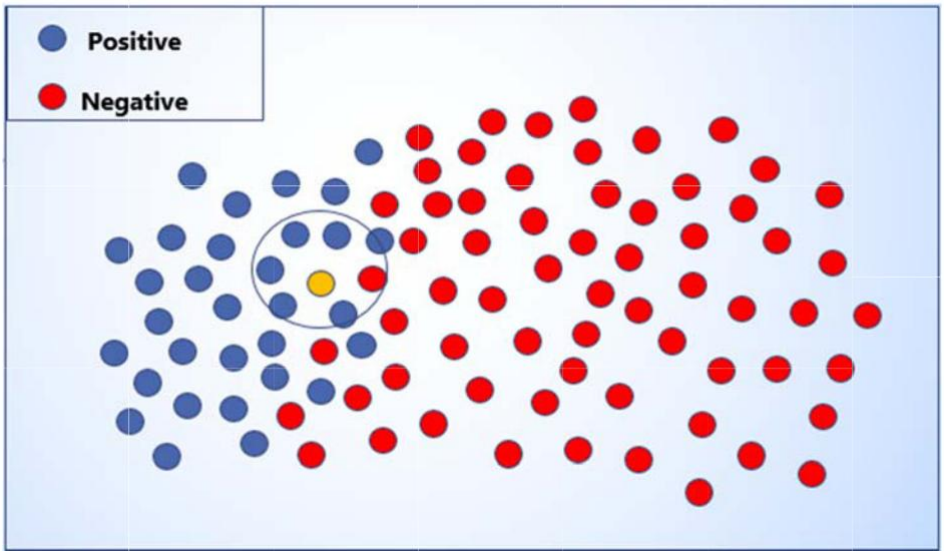
Hình 2.2 mô tả một vùng của bộ dữ liệu lớn. Có thể nhận thấy ngay sự chênh lệch về số phần tử giữa hai lớp đa số và lớp thiểu số.



Hình 2.2 Phân bố dữ liệu

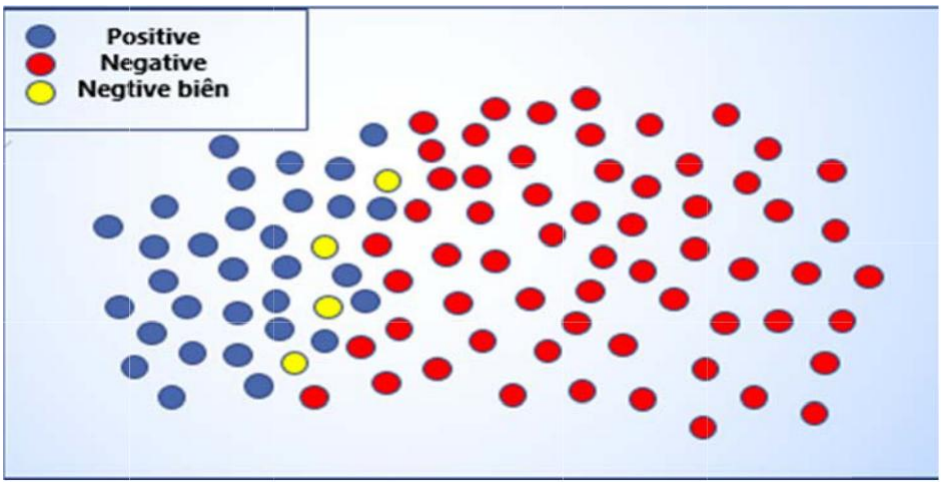
Trong Hình 2.2, phần tử hình tròn màu đỏ là thuộc lớp đa số, được gán nhãn negative. Phần tử hình tròn màu xanh da trời là thuộc lớp thiểu số và được gán nhãn positive.

Hình 2.3 mô tả quá trình xác định k láng giềng cho từng phần tử lớp đa số. Giả sử trong k láng giềng gần nhất có m láng giềng là thuộc lớp thiểu số. Nếu m thỏa mãn điều kiện **k/2 ≤ m < k** thì ta nói phần tử thuộc lớp đa số đang xét là phần tử thuộc đường biên. Ví dụ trên hình 2.3, phần tử đang xét là phần tử được đánh dấu màu vàng, trong số 7 láng giềng gần nhất của nó thì có 6 phần tử thuộc lớp thiểu số, 1 phần tử thuộc lớp đa số. Vậy phần tử đó thuộc biên.

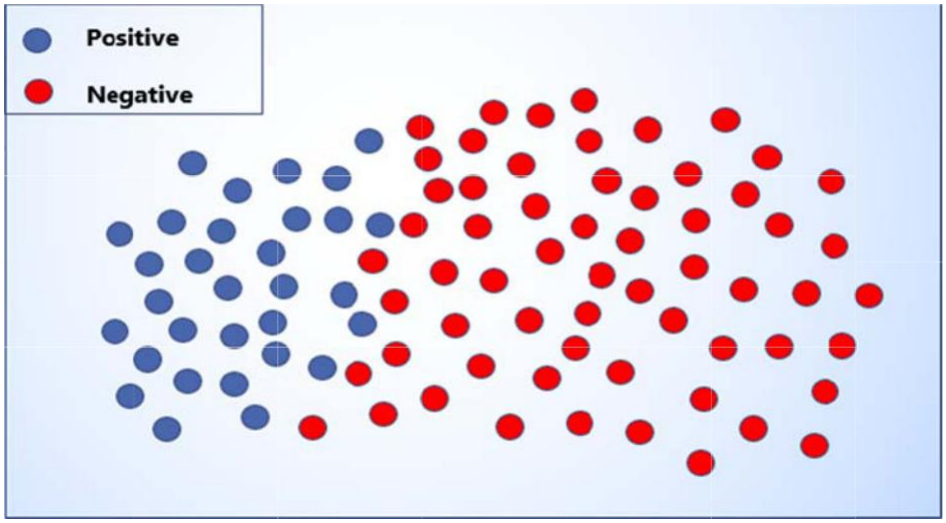


Hình 2.3 Xác định k - láng giềng

Tương tự, ta xác định được các phần tử lớp đa số thuộc đường biên là các phần tử màu vàng như trong hình 2.4.



Hình 2.4 Các phần tử biên



Hình 2.5 Xóa phần tử biên

Tiến hành xóa n% số phần tử biên thuộc lớp đa số đã xác định. Hình 2.5 là các phần tử thuộc biên đã bị xóa toàn bộ. Tuy nhiên, trong thuật toán mới, số phần tử biên sẽ bị xóa theo tỉ lệ phần trăm, phụ thuộc vào tham số n.

2.4.2. Khảo sát nội dung thuật toán

Dựa vào những phân tích ở phần trên, thuật toán RBU có thể mô tả dưới dạng giả code như sau [2]:

RBU Algorthim

Input: Bộ dữ liệu huấn luyện T gồm P positive lớp thiểu số, N negative lớp đa số

n: tỉ lệ phần trăm số phần tử trên biên bị giảm

k: số láng giềng gần nhất đối với một phần tử lớp đa số

m- số phần tử lớp đa số trên đườ ng biên

Output: Tập dữ liệu đã giảm các phần tử trên biên theo tỉ lệ phần trăm n

Các bước thực hiện của thuật toán:

Bước 1: Tìm các phần tử biên thuộc lớp đa số

Với mỗi phần tử trong tập lớp đa số N = {N_1, N_2, $N_{3...}$ N_i} tính k láng giềng gần nhất của nó trong toàn bộ tập dữ liệu huấn luyện T. Gọi số láng giềng thuộc lớp thiểu số trong tổng số k láng giềng gần nhất là m.

Bước 2: Xác định phần tử biên lớp đa số. Nếu **k/2 ≤ m < k** nghĩa là số láng giềng của N thuộc lớp thiểu số lớn hơn số láng giềng thuộc lớp đa số.

Bước 3: Đưa những phần tử thuộc N này vào một mảng border (mảng chứa các phần tử trên đường biên). Các phần tử trong mảng border là phần tử biên lớp đa số.

Bước 4: Giảm theo n phần trăm số phần tử trên đường biên để làm giảm tính mất cân bằng bộ dữ liệu.

Ghi chú:

- Trong thuật toán RBU, thông thường cho hai tham số đầu vào n chạy từ 1 đến 7 và k chạy từ 2 đến 10 để thuật toán khách quan và tổng quát hơn.

Tương tự các thuật toán HMU và HBU, trong pha tiền xử lý dữ liệu có thể sử dung RBU và sau đó sử dụng các thuật toán học máy truyền thống như SVM để xây dựng các bộ phân lớp dữ liệu đối với các dữ liệu mất cân bằng.

Có thể đề xuất thuật toán phân lớp dữ liệu RBU-SVM như sau:

```
Void RBU-SVM() {
        Thực hiện RBU tiền xử lý dữ liêu;
                SVM training;
                SVM prediction;
        }
```

2.4.3. Đánh giá thuật toán

Tương tự các thuật toán HMU và HBU, thuật toán RBU ứng dụng trong pha tiền xử lý dữ liệu trong bài toán phân lớp dữ liệu mất cân bằng. Các thuật toán HMU, HBU và RBU đều nhằm giảm số lượng phân tử của lớp đa số. Độ phức tạp tính toán của RBU tương đồng với HMU và HBU.

So sánh với thuật toán DEC-SVM thì thuật toán RBU cũng có các đặc điểm tương tự đã trình bày trong mục 2.2.3 ở trên.

2.5. Kết luận chương 2

Trong chương 2, luận văn đã khảo sát một số thuật toán: DEC - SVM, HMU, HBU, RBU. Các thuật toán đều thực hiện các ý tưởng nhằm điều chỉnh dữ liệu của tập dữ liệu mất cân bằng trước khi tiến hành quá trình phân lớp dữ liệu. Điều đó sẽ giúp cho các mô hình phân lớp sử dụng các thuật toán học máy hiệu quả hơn.

Trong chương 3, luận văn sẽ ứng dụng các thuật toán cho bài toán phân lớp dữ liệu mất cân bằng đối với bộ dữ liệu thực tế.

CHƯƠNG 3. ỨNG DỤNG

Trong chương 3 luận văn sẽ nghiên cứu ứng dụng các thuật toán đã khảo sát ở chương 2 để thực hiện thử nghiệm phân lớp cho bộ dữ liệu về bệnh tiểu đường.

3.1. Khảo sát và lựa chọn bộ dữ liệu để thử nghiệm

3.1.1. Giới thiệu

Bệnh tiểu đường trong thời đại hiện nay là một trong những căn bệnh thường gặp ở nhiều nước trên thế giới. Ở một số nước, số người mắc căn bệnh này chiếm tỉ lệ tới 10% dân số và số người mắc bệnh ngày một tăng cao. Phần lớn bệnh nhân mắc chứng tiểu đường type 2 và tỉ lệ người bệnh tăng cao liên quan trực tiếp với cách sống trong cuộc sống hiện đại ngày nay.

Một thí dụ điển hình là số phận của những người Ấn Độ Pima. Các bác sỹ phát hiện ra rằng ở những người da đỏ này có gien tiềm ẩn bệnh tiểu đường type 2. Hiện nay những người Pima này sống ở hai vùng khác biệt. Nhóm phía nam sống ở Mexicô, họ vẫn giữ nhiều tập tục sống cổ xưa. Họ phải hoạt động rất nhiều để thu hái được lượng thức ăn hiếm hoi trên sa mạc, sống bằng nghề nông, đánh cá rất vất vả. Nhóm phía bắc sống ở Mỹ, thuộc bang Arizona. Nhóm này sống đúng theo phong cách Mỹ: họ đi làm bằng ô tô, mua đồ trong siêu thị, hoạt động ít, ăn uống quá đầy đủ. Khi nhìn vào tình trạng sức khỏe của hai nhóm bộ tộc này mới thấy được phong cách sống ảnh hưởng tới sức khỏe con người lớn như thế nào. Trong khi những người thuộc nhóm phía nam có vóc dáng nhỏ nhắn, khỏe mạnh, những người thuộc nhóm phía bắc giữ kỷ lục thế giới về béo phì và bệnh tiểu đường, tỉ lệ mắc bệnh tiểu đường trong những người trưởng thành đạt trên 50 %, tỉ lệ này cho người trên 60 tuổi là 80%.

Trong cơ thể những người da đỏ này có loại gien làm cho các tế bào kém nhậy cảm với insulin (hiện tượng nhờn nhẹ với insulin) mà hậu quả của nó là các tế bào chuyển hóa lượng đường rất ít thành năng lượng. Trong thời kỳ thiếu thức ăn,

khả năng đó là ưu điểm: nó làm cho cơ thể phải thích nghi với một lượng năng lượng nhỏ và không không chuyển hóa những tế bào cơ thành năng lượng để hoạt động.

Việc khám phá kiến thức từ cơ sở dữ liệu y tế là rất quan trọng để giúp chẩn đoán y tế hiệu quả. Mục đích của khai thác dữ liệu là trích xuất kiến thức từ thông tin được lưu trữ trong cơ sở dữ liệu và tạo ra các mô tả rõ ràng và dễ hiểu về các mẫu. Trong phần này, luận văn lựa chọn Bộ dữ liệu bệnh tiểu đường của người Indian Pima (Pima Indians Diabetes dataset) [18] để thực hiện ứng dụng các thuật toán đã khảo sát trong chương 2 cho bài toán phân lớp dữ liệu nhị phân: lớp bệnh nhân mắc và không mắc bệnh tiểu đường.

3.1.2. Mô tả bộ dữ liệu Pima-indians-diabetes

Bộ dữ liệu bệnh tiểu đường của người Indian Pima, do Vincent Sigillito tài trợ, là một tập hợp các báo cáo chẩn đoán y tế từ 768 hồ sơ bệnh nhân nữ ít nhất 21 tuổi của người Indian Pima, một dân số sống gần Phoenix, Arizona, Hoa Kỳ.

Trong số chín thuộc tính, sáu thuộc tính mô tả kết quả kiểm tra thể chất, phần còn lại của các thuộc tính là kiểm tra hóa học. Biến phân lớp (tiểu đường = 1 (có mắc bệnh), tiểu đường = 0 (không mắc bệnh)), được biểu thị bằng biến thứ 9. Mục đích là sử dụng 8 biến đầu tiên để dự đoán giá trị của biến thứ 9.

Các thuộc tính của bộ dữ liệu Pima-indians-diabetes được mô tả chi tiết trong Bảng 3.1 dưới đây [18].

Bảng 3.1 Các thuộc tính của bộ dữ liệu Pima-indians-diabetes

TT	Tên thuộc tính	Mô tả	Tính chất
1	Pregnancies	Số lần mang thai	
2	Glucose	Nồng độ glucose huyết tương 2h trong xét nghiệm dung nạp glucose đường uống	

TT	Tên thuộc tính	Mô tả	Tính chất
3	BloodPressure	Huyết áp tâm trương (mm Hg)	
4	Skinthickness	Độ dày nếp gấp da (mm)	
5	Insulin	Huyết thanh 2 giờ (mu U / ml)	
6	BMI	Chỉ số khối cơ thể	
7	Diabetespedigree	Chức năng phả hệ tiểu đường	
8	Age	Tuổi (năm).	
9	Outcome	Thuộc tính phân lớp (0,1)	

Các kỹ thuật xử lý dữ liệu, khi được áp dụng trước khi khai thác, có thể cải thiện đáng kể chất lượng tổng thể của các mẫu được khai thác hoặc thời gian cần thiết cho khai thác thực tế. Tiền xử lý dữ liệu là một bước quan trọng, vì các quyết định chất lượng phải dựa trên dữ liệu chất lượng. Trong 768 trường hợp, 5 bệnh nhân có glucose là 0, 11 bệnh nhân có chỉ số khối cơ thể là 0, 28 người khác có huyết áp tâm trương là 0, 192 người khác có chỉ số độ dày nếp gấp da 0, và 140 người khác có nồng độ insulin huyết thanh bằng 0 là điều không thể. Sau khi xóa các trường hợp này, có 392 trường hợp không có giá trị thiếu (130 trường hợp dương tính được kiểm tra và 262 trường hợp âm tính).

3.2. Xây dựng kịch bản và lựa chọn công cụ thử nghiêm

3.2.1. Xây dựng kịch bản thử nghiệm

Trong mục này, luận văn sẽ thực hiện thử nghệm với bài toán sau:

Dữ liệu đầu vào:

(1) Bộ dữ liệu ***pima-indians-diabetes***

(2) Các thuật toán thử nghiệm:

- Thuật toán DEC-SVM
- Thuật toán HMU-SVM

- Thuật toán HBU-SVM
- Thuật toán RBU-SVM

Dữ liệu ra:

Các tiêu chí, kết quả đánh giá hiệu năng của các thuật toán nghiên cứu trong chương 2 áp dụng vớ bộ dữ liệu ***pima-indians-diabetes***

Luận văn sẽ tiến hành thử nghiệm theo hai kịch bản trình bày dưới đây.

Kịch bản thứ nhất:

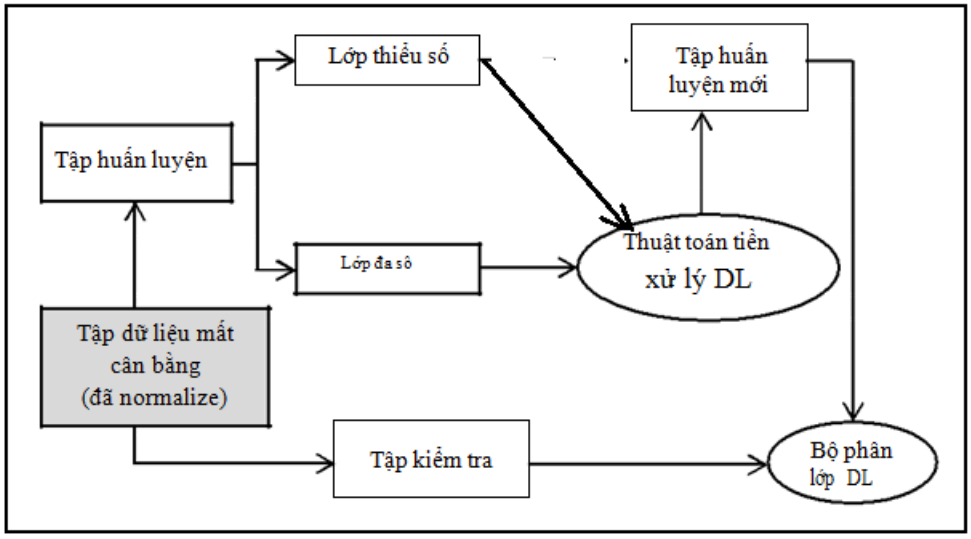
Trong kịch bản thứ nhất, luận văn sẽ thực hiện sử dụng thuật toán SVM để phân lớp dữ liệu với bộ dữ liệu đã chọn mà không sử dụng các thuật toán tiền xử lý dữ liệu mất cân bằng.

Kịch bản thứ hai:

Trong kịch bản thứ hai, luận văn sẽ thực hiện phân lớp dữ liệu sau khi xử lý dữ liệu mất cân bằng sử dụng thuật toán xử lý dữ liệu mất cân bằng.

3.2.2. Mô hình thử nghiệm

Mô hình tiến hành thử nghiệm được mô tả trong hình 3.1 dưới đây.



Hình 3.1 Mô hình thử nghiệm

Trong mô hình trên, các thuật toán tiền xử lý dữ liệu sẽ được lựa chọn lần lượt là DEC-SVM, HMU, HBU và RBU.

3.2.3. Lựa chọn công cụ thử nghiệm

Weka là một phần mềm miễn phí về học máy được viết bằng Java, phát triển bởi University of Wekato. Weka có thể coi như là bộ sưu tập các thuật toán về học máy dùng trong phân tích và khai phá dữ liệu. Các thuật toán đã được xây dựng sẵn và người dùng chỉ việc lựa chọn để sử dụng. Do đó Weka rất thích hợp cho việc thử nghiệm các mô hình mà không mất thời gian để xây dựng chúng. Weka có giao diện sử dụng đồ họa trực quan và cả chế độ command line. Ngoài các thuật toán về học máy như dự đoán, phân loại, phân cụm, Weka còn có các công cụ để trực quan hóa dữ liệu rất hữu ích trong quá trình nghiên cứu, phân tích dữ liệu lớn.

Từ những lý do trên, luận văn lựa chọn công cụ thực nghiêm là phần mềm Weka version 3.7.12 [19].

Hình 3.2 Màn hình khởi động Weka

Các tính năng chính của Weka:

- Weka bao gồm một tập các công cụ tiền xử lý dữ liệu, các thuật toán học máy để khai phá dữ liệu và các phương pháp thử nghiệm đánh giá.

- Weka có giao diện đồ họa (gồm cả tính năng hiển thị hóa dữ liệu)

- Weka bao gồm các môi trường cho phép so sánh các thuật toán học máy trên bộ dữ liệu do người dùng lựa chọn.

Các môi trường chính trong Weka:

(1) Simple CLI : giao diện đơn giản kiểu dòng lệnh (như MS-DOS).

(2) Explorer : môi trường cho phép sử dụng tất cả các khả năng của Weka để khám phá dữ liệu.

(3) Experimenter: môi trường cho phép tiến hành các thí nghiệm và thực hiện các kiểm tra thống kê (statistical tests) giữa các mô hình máy học. Môi trường này bao gồm:

- **Preprocess:** Để chọn và thay đổi (xử lý) dữ liệu làm việc.
- **Classify:** Để huấn luyện và kiểm tra các mô hình học máy (phân loại, hoặc hồi quy/dự đoán).
- **Cluster:** Để học các nhóm từ dữ liệu (phân cụm).
- **Associate:** Để khám phá các luật kết hợp từ dữ liệu.
- **Select attributes:** Để xác định và lựa chọn các thuộc tính liên quan (quan trọng) nhất của dữ liệu.
- **Visualize:** Để xem (hiển thị) biểu đồ tương tác 2 chiều đối với dữ liệu.

(4) KnowledgerFlow: môi trường cho phép bạn tương tác đồ họa kiểu kéo/ thả để thiết kế các bước(các thành phần) của một thí nghiệm.

Để tiến hành thử nghiệm, cần lựa chọn "Explorer": giao diện cho phép sử dụng tất cả các chức năng cơ sở của Weka bằng cách lựa chọn menu.

Để đánh giá hiệu năng các bộ phân loại cần lựa chọn các tùy chọn cho việc kiểm tra trong (test options) bao gồm:

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

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

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

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

3.3. Thử nghiệm và đánh giá kết quả thử nghiệm

3.3.1. Mô tả thử nghiệm

Máy tính sử dụng cho quá trình chạy Weka để đánh giá hiệu năng các thuật toán là laptop có cấu hình:

- Bộ xử lý Intel -Core i3 4005U,
- RAM: 4GB.

Bộ công cụ weka phiên bản 3.7.12.

Bộ dữ liệu thử nghiệm là *pima-indians-diabetes.csv* gồm 768 bản ghi, 9 thuộc tính.

Các thuật toán thử nghiêm:

- Thuật toán DEC-SVM
- Thuật toán HBU
- Thuật toán HMU
- Thuật toán RBU

Thực hiện thử nghiệm theo hai kịch bản nêu ở mục 3.2.1.

Các bước thực hiên như sau:

Bước 1: Chuẩn hóa dữ liệu bằng Filter standardize của Weka. Dữ liệu được xử lý để có kỳ vọng tại 0 và có độ lệch chuẩn bằng 1. Việc chuẩn hóa giúp các thuật toán không bị thiên lệch về một số đặc trưng và hơn nữa giúp quá trình học hội tụ nhanh hơn.

Đối với kịch bản 2: thực hiện bước 2

Bước 2: Cân bằng dữ liệu bằng một trong các thuật toán đề xuất (RBU, HBU, HMU, DEC-SVM)

Bước 3: Với bộ dữ liệu thu được, thực hiện phân lớp bằng thuật toán SVM trong Weka

3.3.2. Kết quả thử nghiệm

Trong mục này luận văn trình bày một số kết quả chính khi chạy trên Weka. Do giới hạn về số trang của luận văn nên không thể nêu chi tiết các thao tác.

(1) Kết quả phân lớp trước khi xử lý dữ liệu mất cân bằng theo kịch bản 1

Kết quả phân lớp trước khi xử lý dữ liệu mất cân bằng sử dụng thuật toán SVM được trình bày trong bảng 3.2

Bảng 3.2 Kết quả phân lớp trước khi xử lý dữ liệu mất cân bằng sử dụng thuật toán SVM

```
=== Detailed Accuracy By Class ===
TP Rate   FP Rate   Precision   Recall  F-Measure  ROC Area  Class
0.866     0.448      0.783     0.866     0.822      0.709    0
0.552     0.134      0.688     0.552     0.613      0.709    1
0.757     0.338      0.75      0.757     0.749      0.709    Avg.

=== Confusion Matrix ===
   a   b   <-- classified as
 433  67 |   a = 0
 120 148 |   b = 1
```

(2) Kết quả phân lớp sau khi xử lý dữ liệu mất cân bằng theo kịch bản 2

Kết quả phân lớp sau khi xử lý dữ liệu mất cân bằng với thuật toán DEC-SVM được trình bày trong bảng 3.3

Bảng 3.3 Kết quả phân lớp sau khi xử lý dữ liệu mất cân bằng với thuật toán DEC-SVM

```
=== Detailed Accuracy By Class ===

TP Rate  FP Rate   Precision   Recall  F-Measure  ROC Area  Class
0.757     0.213      0.775     0.757     0.766      0.772    0
0.787     0.243      0.77      0.787     0.778      0.772    1
0.772     0.228      0.772     0.772     0.772      0.772    Avg.

=== Confusion Matrix ===

   a   b   <-- classified as
 355 114 |   a = 0
 103 381 |   b = 1
```

Kết quả phân lớp sau khi xử lý dữ liệu mất cân bằng với thuật toán HMU được trình bày trong bảng 3.4

Bảng 3.4 Kết quả phân lớp sau khi xử lý dữ liệu mất cân bằng với thuật toán HMU

```
=== Detailed Accuracy By Class ===

TP Rate  FP Rate   Precision   Recall  F-Measure   ROC Area Class
0.741     0.187      0.8       0.741     0.769      0.777    0
0.813     0.259      0.757     0.813     0.784      0.777    1
0.777     0.223      0.779     0.777     0.777      0.777    Avg.

=== Confusion Matrix ===

   a   b   <-- classified as
 200  70 |   a = 0
  50 218 |   b = 1
```

Kết quả phân lớp sau khi xử lý dữ liệu mất cân bằng với thuật toán HBU được trình bày trong bảng 3.5

Bảng 3.5 Kết quả phân lớp sau khi xử lý dữ liệu mất cân bằng với thuật toán HBU

```
=== Detailed Accuracy By Class ===

TP Rate  FP Rate   Precision  Recall  F-Measure   ROC Area  Class
0.866     0.146      0.856     0.866     0.861      0.86     0
0.854     0.134      0.864     0.854     0.859      0.86     1
0.86      0.14       0.86      0.86      0.86       0.86     Avg.

=== Confusion Matrix ===

   a   b   <-- classified as
 232  36 |   a = 0
  39 229 |   b = 1
```

Kết quả phân lớp sau khi xử lý dữ liệu mất cân bằng với thuật toán RBU được trình bày trong bảng 3.6

Bảng 3.6 Kết quả phân lớp sau khi xử lý dữ liệu mất cân bằng với thuật toán RBU

```
=== Detailed Accuracy By Class ===

TP Rate   FP Rate   Precision  Recall  F-Measure  ROC Area  Class
0.906     0.302      0.827     0.906     0.865      0.802    0
0.698     0.094      0.824     0.698     0.756      0.802    1
0.826     0.222      0.825     0.826     0.822      0.802    Avg.

=== Confusion Matrix ===

   a   b   <-- classified as
 386  40 |   a = 0
  81 187 |   b = 1
```

Kết quả phân lớp trước và sau khi xử lý dữ liệu mất cân bằng với các thuật toán DEC-SVM, HBU, HMU, RBU được tổng hợp theo bảng 3.7

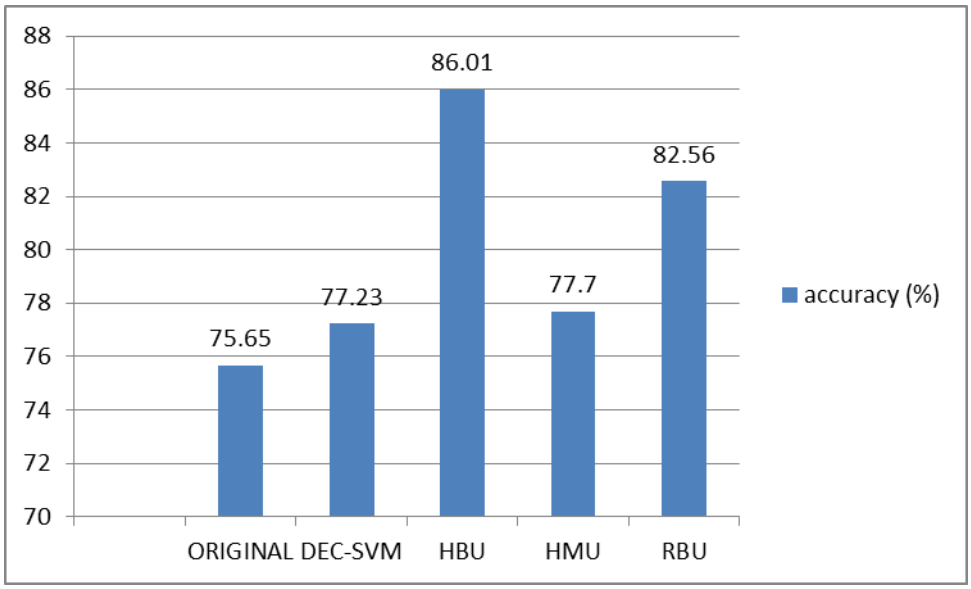
Bảng 3.7 Bảng tổng hợp kết quả phân lớp trước và sau khi xử lý dữ liệu mất cân bằng

Thuật toán	accuracy (%)	Negative			Positive		
		Pre	Rec	F1	Pre	Rec	F1
ORIGINAL	75.65	78.3	86.6	82.2	68.8	55.2	61.3
DEC-SVM	77.23	77.5	75.7	76.6	77.0	78.7	77.8
HBU-SVM	86.01	85.6	86.6	86.1	86.4	85.4	85.9
HMU-SVM	77.70	80	74.1	76.9	75.7	81.3	78.4
RBU	82.56	82.7	90.6	86.5	82.4	69.8	75.6

3.3.3. Đánh giá kết quả thử nghiệm

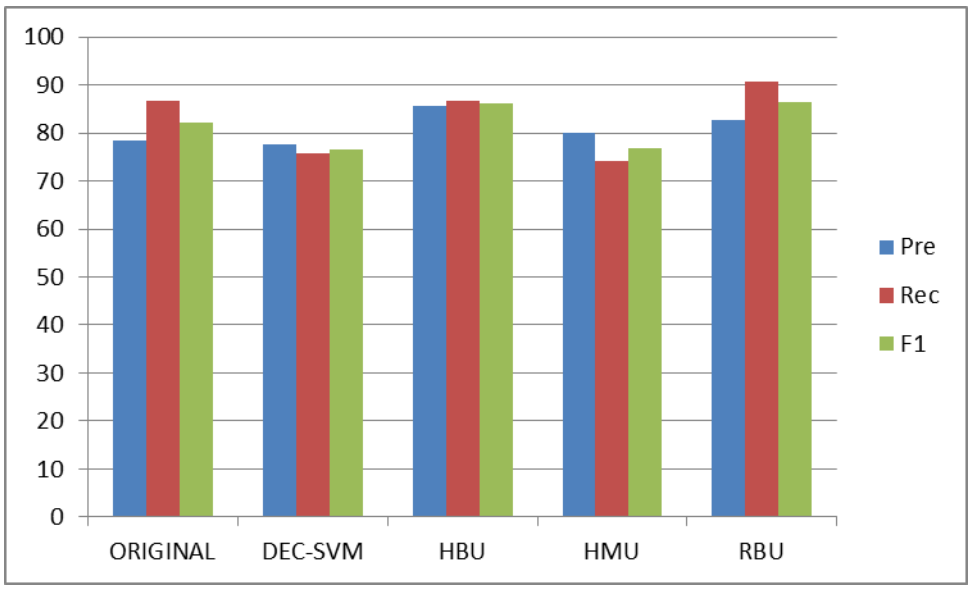
Dựa vào kết quả thử nghiệm đã trình bày ở mục trên, mục này luận văn sẽ thực hiện phân tích và đánh giá kết quả.

Kết quả độ chính xác của các thuật toán thử nghiệm theo hai kịch bản được biểu diễn dưới dạng biểu đồ như trong hình 3.3.

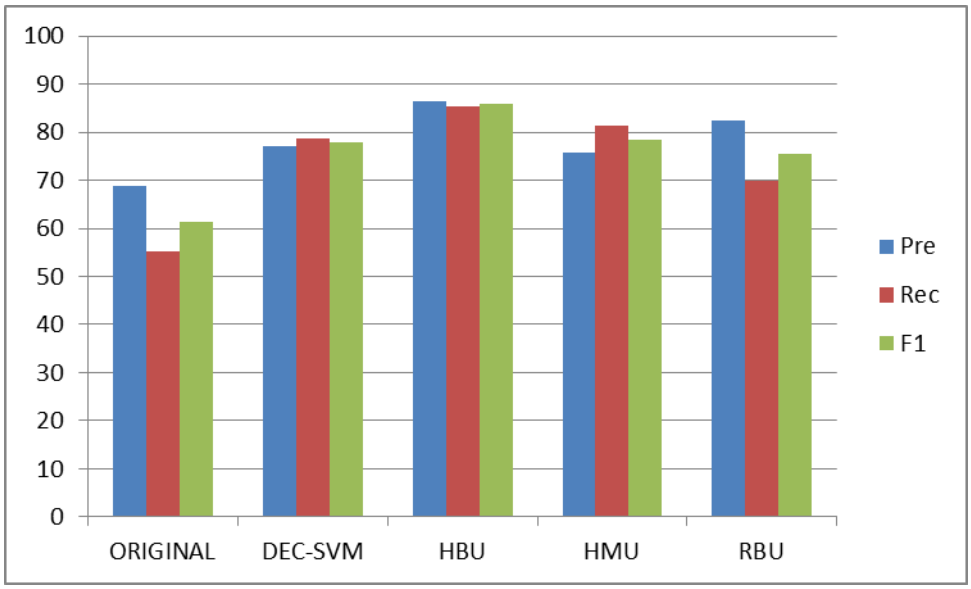


Hình 3.3 Biểu đồ so sánh độ chính xác của phân lớp trên dữ liệu trước và sau khi xử lý dữ liệu mất cân bằng.

Quan sát biểu đồ trên hình 3.3 nhận thấy rằng, các thuật toán thử nghiệm đều cho kết quả có tỉ lệ phân loại chính xác cao hơn so với bộ dữ liệu ban đầu khi chưa áp dụng thuật toán.



Hình 3.4 Biểu đồ kết quả phân lớp lớp Negative trước và sau khi xử lý dữ liệu mất cân bằng



Hình 3.5 Biểu đồ kết quả phân lớp lớp Positive trước và sau khi xử lý dữ liệu mất cân bằng

Từ các kết quả ở trên ta thấy sau khi điều chỉnh bộ dữ liệu bởi các thuật toán tiền xử lý dữ liệu mất cân bằng DEC-SVM, HBU, HMU, RBU thì hiệu quả phân lớp các bộ dữ liệu cao hơn hẳn so với việc phân lớp của bộ dữ liệu ban đầu.

Các thuật toán đã khảo sát có thể kết hợp với một số kỹ thuật như trích chọn đặc trưng phù hợp có thể cho kết quả tốt hơn, đặc biệt là với các tập dữ liệu có kích thước lớn.

3.4. Kết luận chương 3

Trong chương 3 luận văn đã tiến hành thử nghiệm các thuật toán DEC-SVM, HMU, HBU và RBU cho bài toán phân lớp dữ liệu trên dữ liệu mất cân bằng cho bộ dữ liệu về chứng tiểu đường của người Indian Pima.

Kết quả thử nghiệm bước đầu cho thấy các thuật toán phân lớp trên có thể triển khai trong thực tế và phù hợp với các yêu cầu đề ra cho bài toán phân lớp dữ liệu trên dữ liệu mất cân bằng

KẾT LUẬN

Kết quả đạt được của luận văn

Với mục tiêu nghiên cứu một số kỹ thuật để nâng cao hiệu năng phân lớp dữ liệu trên tập dữ liệu mất cân bằng và ứng dụng, luận văn đã đạt được một số kết quả như sau:

- Nghiên cứu tổng quan về bài toán phân lớp dữ liệu và các vấn đề liên quan.
- Khảo sát tổng quan về dữ liệu mất cân bằng.
- Khảo sát hướng tiếp cận về dữ liệu và hướng tiếp cận về thuật toán để nâng cao hiệu năng phân lớp dữ liệu trên dữ liệu mất cân bằng.
- Khảo sát chi tiết các thuật toán: DEC-SVM, HMU, HBU và RBU.
- Khảo sát bộ dữ liệu về bệnh tiểu đường pima-indians-diabetes.
- Thực hiện thử nghiệm phân lớp dữ liệu với DEC-SVM, HMU, HBU và RBU trên bộ dữ liệu pima-indians-diabetes. Kết quả thử nghiệm cho thấy hiệu quả phân lớp dữ liệu sau khi sử dụng thuật toán đã được khảo sát.

Tuy nhiên, do hạn chế về mặt thời gian, luận văn chưa tiến hành thử nghiệm với các bộ dữ liệu lớn, Do đó, hiệu quả thử nghiệm chưa cao.

Hướng phát triển tiếp theo

Trên cơ sở nghiên cứu và các kết quả đạt được, đề tài luận văn có thể phát triển tiếp theo như sau:

- Tiếp tục hoàn thiện các kết quả đã có để có thể xây dựng các mô hình phân lớp trên dữ liệu mất cân bằng với các bộ dữ liệu trong thực tế thường có kích thước lớn, các thuộc tính của các phần tử dữ liệu thường bao gồm cả dạng số và dạng phi số.
- Nghiên cứu thêm về các kỹ thuật trích chọn đặc trưng cho các bộ dữ liệu mất cân bằng nhằm nâng cao hiệu quả cho các mô hình phân lớp.

DANH MỤC CÁC TÀI LIỆU THAM KHẢO

TÀI LIỆU TIẾNG VIỆT

[1] Nguyễn Thị Lan Anh (2017). Thuật toán HMU trong bài toán phân lớp dữ liệu mất cân bằng. *Tạp chí Khoa học và Giáo dục, Trường Đại học Sư phạm Huế,* **2,** 101–108.

[2] Nguyễn Thị Lan Anh (2018). Phân lớp dữ liệu mất cân bằng với thuật toán HBU. Tạp chí Khoa học và Giáo dục, Trường Đại học Sư phạm Huế, 4, 110–116.

[3] Bùi Dương Hưng,, Đặng Xuân Thọ, Vũ Văn Thỏa (2019). KSI - Phương pháp phân cụm với bộ lọc ngẫu nhiên để loại bỏ nhiễu trong dữ liệu mất cân bằng*, Tạp chí Khoa học công nghệ thông tin và truyền thông, Học viện Công nghệ thông tin và truyền thông, 01, 55-60.*

[4] Phạm Thị Hường, Phạm Văn Kiên, Đỗ Ngọc Quỳnh (2017)- Phương pháp DEC-SVM phân lớp dữ liệu mất cân bằng

[5] Bùi Minh Quân, Phạm Xuân Hiền, Huỳnh Xuân Diệp (2013). Nâng cao độ chính xác phân loại lớp ít mẫu từ tập dữ liệu mất cân bằng*, Tạp chí Khoa học Trường đại học Cần Thơ.*

[6] Nguyễn Mai Phương, Trần Thị Ánh Tuyết, Nguyễn Thị Hồng, Đặng Xuân Thọ (2015), Random Border Undersampling: Thuật toán mới giảm phần tử ngẫu nhiên trên đường biên trong dữ liệu mất cân bằng, Kỷ yếu FAIR, 612-619.

TÀI LIỆU TIẾNG ANH

[7] Han J., Kamber M. (2011) – "Data mining: Concepts and Techniques" - 3nd Edition, Morgan Kaufman Publishers.

[8] Sain, H. & Purnami, S. W. (2015). Combine Sampling Support Vector Machine for Imbalanced Data Classification. *Procedia Comput. Sci.* **72,** 59–66.

[9] Leichen Chen, Zhihua Cai, Lu Chen (2010), A Novel Different Evolution- Clustering Hybrid Resampling Algorithm on Imbalanced Datasets", in: Knowledge Discovery and Data Mining, 2010. WKDD

'10. Third International Conference, 81-85.

[10] Chumphol Bunkhumpornpat, Krung Sinapiromsaran, Chidchanok Lursinsap (2009), "*Safe-Level-SMOTE: Safe-Level- Synthetic Minority Over Sampling Technique for Handling the Class Imbalanced Problem*", in Advances in Knowledge Discovery and Data Mining: Springer-Verlag Berlin Heidelberg, vol. 5476, pp. 475-482

[11] Mikel Galar, Alberto Fernandez, Edurne Barrenechea, Humberto Bustince (2011), "*A Review on Ensembles for the Class Imbalance Problem: Bagging – Boosting, and Hybrid-Based Approaches*", IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, vol. 42, no. 4, pp.463-484.

[12] Han Hui, Wang Wen-Yuan, and Mao Bing- Huan (2005), "*Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning*," in ICIC 2005, pp. 878-887.

[13] Sotiris Kotsiantis, Dimitris Kanellopoulos, Panayiotis Pintelas (2006), "*Handling imbalanced datasets: A review*", GESTS International Transactions on Computer Science and Engineering, vol.30.

[14] Xu - Ying Liu, Jianxin Wu, and Zhi-Hua Zhou (2006), Exploratory Undersampling for Class-Imbalance Learning, 6th IEEE International Conference onData Mining (ICDM'06), 965-969.

[15] T. M. Mitchell [1997] – "Machine Learning", McGraw-Hill.

[16] Sun Yanmin, Wong Andrew K. C., and Kamel Mohamed S.(2009), "*Classification of imbalanced data: A review*", International Journal of Pattern Recognition and Artificial Intelligence, vol. 23, pp. 687–719.

Trang WEB

[17] https://archive.ics.uci.edu/ml/datasets/Diabetes

[18] https://en.wikipedia.org/wiki/Precision_and_recal

[19] https://sourceforge.net/projects/weka/