Equipment with new general education program, Volume 2, Issue 315 (June 2024)
ISSN 1859 - 0810

# Cải thiện kết quả dự đoán điểm thi của sinh viên bằng Máy học với sự hỗ trợ của R

***Nguyễn Thị Lan Anh ****

** Trường ĐH Sư phạm, Đại học Huế*

*Received: 16/04/2024; Accepted: 26/04/2024; Published: 15/5/2024*

***Abstract:*** *Predicting final exam scores of students is important to forecast student accomplishment in the learning process and functioned to help the students who might fail. This paper presents a method that based on machine learning techniques to enhance prediction results by handling the imbalanced data issues. The experimental results achieves the highest performance with 43.70% of Precision, 45.1% of Recall and 44.04% of F1.*

***Keywords:*** *Prediction of student performance, machine learning, imbalanced data*

## 1. Đặt vấn đề

Việc dự đoán sớm kết quả thi cho học sinh, sinh viên dựa trên tình trạng mỗi cá nhân để các em có thể kịp thời điều chỉnh cách thức, thái độ học tập và có phương án ôn tập phù hợp, cũng như điều chỉnh phương pháp giảng dạy của giáo viên, là một trong những vấn đề cần thiết và được quan tâm nhiều trong những năm lại đây. Cùng với sự phát triển của học máy, nhiều chương trình dự đoán, tư vấn đã được nghiên cứu và xây dựng [1] [2] [3]. Tuy nhiên, kết quả thu được vẫn còn khá hạn chế, mà một trong những lý do là vấn đề mất cân bằng dữ liệu. Do dữ liệu mất cân bằng, khi sử dụng các thuật toán phân lớp truyền thống để phân lớp, đa phần các phần tử lớp thiểu số sẽ bị phân lớp nhầm thành lớp đa số [4]. Do vậy, một sinh viên có khả năng thi hỏng nhiều khi bị bỏ qua, dự đoán thành điểm cao, hoặc ngược lại, có khả năng điểm cao lại bị dự đoán thành thi hỏng, sẽ gây ra các hậu quả đáng tiếc, ví dụ như chủ quan hoặc hoang mang, mất phương hướng trong quá trình ôn tập v.v.

Để khắc phục vấn đề mất cân bằng dữ liệu, nhiều phương pháp đã được nghiên cứu, đề xuất và thu được các kết quả khả quan khi áp dụng vào các lĩnh vực khác nhau, như phát hiện gian lận tài chính, dự đoán cấu trúc DNA, dự đoán tương tác giữa protein-protein, phân lớp microRNA, chẩn đoán bệnh trong y học… Theo một số tác giả, trong các nhóm phương pháp xử lý dữ liệu mất cân bằng là phương pháp tiếp cận ở mức độ thuật toán, phương pháp tiếp cận ở mức dữ liệu, thì nhóm các phương pháp xử lý ở mức dữ liệu cho kết quả tốt hơn [5]. Nhóm phương pháp này bao gồm các phương pháp sinh thêm phần tử cho lớp thiểu số, phương pháp giảm bớt phần tử cho lớp đa số và phương pháp kết hợp.

Trong bài báo này, chúng tôi sẽ áp dụng phương pháp sinh thêm phần tử cho lớp thiểu số SCUT [6] để cải thiện hiệu suất bài toán dự đoán kết quả thi của sinh viên.

## 2. Nội dung nghiên cứu

### *2.1 Phương pháp sinh thêm phần tử cho lớp thiểu số*

Xét tập dữ liệu mất cân bằng $S = S_{min} \cup S_{max}$ ($S_{min}$ là lớp thiểu số, $S_{max}$ là lớp đa số), tập các thuộc tính $A \cup \{D\}$, với D là nhãn lớp. Thuật toán SMOTE [6] cho phép các phần tử lớp thiểu số tạo ra phần tử mới, cùng nhãn lớp, nằm giữa nó và một trong những phần tử láng giềng cùng nhãn lớp gần nhất của nó, được mô tả như bên dưới:

Với mỗi phần tử $x_i \in S_{min}$, tìm k láng giềng gần nhất của $x_i$ trong $S_{min}$.

Chọn ngẫu nhiên một phần tử $x_n$ trong số k láng giềng tìm được.

Phần tử mới được sinh ra $x_{new}$ xác định:

$$x_{new}[A_m] = x_i[A_m] + \delta * (x_i[A_m] - x_n[A_m]), \text{ m =1..|A|,}$$

δ là một giá trị ngẫu nhiên thuộc [0,1]

Để cải thiện hiệu suất của bài toán phân lớp dữ liệu mất cân bằng đa lớp, Agrawal A. và cộng sự đã đề xuất thuật toán SCUT [6], trong đó, mỗi lớp sẽ được điều chỉnh số phần tử sao cho kết quả thu được là tất cả các lớp đều có m phần tử, với m là trung bình số phần tử của tất cả các lớp ban đầu. Cụ thể, những lớp ban đầu có ít hơn m phần tử sẽ được sinh thêm phần tử mới bằng thuật toán SMOTE, những lớp ban đầu có nhiều hơn m phần tử sẽ được phân thành các cụm bằng thuật toán EM và chọn ngẫu nhiên từ các cụm này một số phần tử để loại bỏ sao cho tổng số phần tử của các cụm sau loại bỏ sẽ bằng m. Việc làm giảm số lượng phần tử lớp đa số có khả năng làm mất

một số thông tin quan trọng của lớp này, nên có thể làm cho kết quả toàn cục bị ảnh hưởng.

Trong bài báo này, thay vì sử dụng thuật toán SCUT như trong [6] để mỗi lớp sẽ nhận được m phần tử, chúng tôi sẽ điều chỉnh để tất cả các lớp đều nhận được max phần tử, với max là số phần tử của lớp có nhiều phần tử nhất trong tập dữ liệu ban đầu. Do đó, với việc sử dụng phương pháp SCUTM, sẽ không có lớp nào được thực hiện quá trình làm giảm phần tử.

### *2.2. Dữ liệu thực nghiệm*

Chúng tôi sử dụng bộ dữ liệu UCI về kết quả thi của sinh viên khoa Kỹ thuật và khoa Giáo dục, với 31 thuộc tính, sinh viên có Điểm cuối kỳ (Output Grade) thuộc một trong 8 loại khác nhau, như trong [8]. Sau đó, phương pháp Năm số (Five-number method) được sử dụng để phân loại kết quả thi của các sinh viên thành ba nhóm: sinh viên có kết quả bé hơn hoặc bằng 1 được phân loại là Thấp, từ 1 đến 6 là Trung bình, trên 6 là Cao. Như vậy, số sinh viên đạt kết quả Thấp:Trung bình:Cao lần lượt là 43:72:30, và cả tập dữ liệu này mất cân bằng.

### *2.3. Thực nghiệm đánh giá hiệu suất dự đoán kết quả học tập của học sinh*

Quá trình thực nghiệm để đánh giá hiệu suất dự đoán kết quả thi của SV được thực hiện như sau:

1. Trước tiên, 95% đặc trưng được xác định bằng phương pháp lựa chọn đặc trưng dựa vào chỉ số Gini của gói FSinR trong R được xác định. Chúng tôi sử dụng hàm directFeatureSelection, với phương pháp tìm kiếm là selectPercentile, phương pháp xác định tầm quan trọng của đặc trưng là filterEvaluator, thu được 28 đặc trưng quan trọng. Đoạn chương trình giúp xác định các đặc trưng quan trọng như ở Hình 2.1.

```
library(FSinR)
dataset = read.table("higher+education+students+performance+evaluation.vector",sep="\t",head=T,row.names=1)
filter_evaluator <- filterEvaluator('giniIndex')
direct_search_method <- directSearchAlgorithm('selectPercentile', list(percentile=95))
res <- directFeatureSelection(dataset, 'Class', direct_search_method, filter_evaluator)
```

*Hình 2.1: Xác định các đặc trưng quan trọng*

2. Dữ liệu sau xử lý được thực hiện 5 lần kiểm chứng chéo 5-fold (5-fold crossvalidation). Ở mỗi lần lặp, mỗi lớp có số phần tử ít hơn của tập huấn luyện tương ứng được sinh thêm phần tử để đảm bảo có số lượng tương đương với số lượng phần tử của lớp đa số bằng hàm oversample_smote của gói lệnh scutr trong R. Các lớp sau khi sinh thêm phần tử mới được kết hợp với lớp đa số để tạo ra tập huấn luyện mới. Trung bình cộng giá trị độ đo hiệu suất của kết quả phân lớp cho tập kiểm tra trong 5 lần lặp này chính là kết quả để đánh giá hiệu suất của mô hình. Chúng tôi sử dụng máy vector hỗ trợ (KSVM), thuộc gói lệnh kernlab làm bộ phân lớp chính.

```
library(scutr)
dataname="higher+education+students+performance+evaluation.vector"
for(i in 1:5)
{
    train = read.table(paste(dataname,"-train-",i,".txt",sep=""),sep="\t",head=T)
    train = cbind(train[,-ncol(train)],as.factor(train$Class))
    colnames(train)[ncol(train)] = "Class"
    noclasses=unique(train$Class)
    ovspl=max(table(train$Class))
    gendata=matrix(nrow=0,ncol=ncol(train))
    colnames(gendata)=colnames(train)
    for( class_i in noclasses){
        data_i= train[train[,ncol(train)]== class_i,]
        ovsdata_i= oversample_smote(data_i,class_i,"Class",ovspl)
        gendata=rbind(gendata,ovsdata_i)
    }
  write.table(gendata,paste(dataname,"-Smote-train-",i,".txt",sep=""),
```

*Hình 2.2: Sinh thêm phần tử mới*

Để đánh giá hiệu quả của quá trình phân lớp, trong bài báo này, các độ đo Precisiom (Độ chính xác), Recall (Độ hồi tưởng) và F1-score được sử dụng thay cho độ chính xác toàn thể (accuracy) như đối với việc phân lớp tập dữ liệu cân bằng, nhằm tránh tình trạng độ chính xác toàn thể rất cao nhưng rất ít phần tử lớp thiểu số được dự đoán đúng. Các giá trị Precision, Recall, F1 của tập dữ liệu D gồm N nhãn lớp được định nghĩa như sau:

Precision (%) $= \frac{\sum_{i=1}^{N} Precision_i}{N}$, trong đó $Precision_i$ là độ chính xác của nhãn lớp thứ i, được xác định $Precision_i = \frac{TP_i}{TP_i + FP_i} * 100$.

Recall (%) $= \frac{\sum_{i=1}^{N} Recall_i}{N}$, trong đó $Recall_i$ là độ hồi tưởng của nhãn lớp thứ i, được xác định $Recall_i = \frac{TP_i}{TP_i + FN_i} * 100$.

F1 (%) $= \frac{\sum_{i=1}^{N} F1_i}{N}$, trong đó: của nhãn lớp thứ i được xác định: $F1_i(\%) = \frac{2 * Recall_i * Precision_i}{Recall_i + Precision_i}$.

Ở đây, $TP_i$ và $FN_i$ lần lượt là số phần tử lớp thứ i được dự đoán đúng và bị dự đoán sai so với nhãn lớp thực sự của chúng; $TN_i$ và $FP_i$ lần lượt là số phần tử không thuộc lớp thứ i được dự đoán là không thuộc lớp thứ i và thuộc lớp thứ i.

Đoạn chương trình để sinh thêm phần tử mới được trình bày trong Hình 2.2, và Hình 2.3 trình bày cách dự đoán và đánh giá hiệu quả quá trình phân lớp.

```
for(i in 1:5)
{
    train = read.table(paste(datasetname,"-Smote-train-",i,".txt",sep=""),sep="\t",head=T)
    test = read.table(paste(datasetname,"-test-",i,".txt",sep=""),sep="\t",head=T)
    model <- ksvm(Class~.,data=train,type = "spoc-svc",kernel="rbfdot",C=100,scaled=F)
    ypred = predict(model,test[,-ncol(test)])
    for(class_i in noclasses){
        TP = sum(test[,ncol(test)]== class_i & ypred == class_i)
        FP = sum(test[,ncol(test)]!= class_i & ypred == class_i)
        TN = sum(test[,ncol(test)]!= class_i & ypred != class_i)
        FN = sum(test[,ncol(test)]== class_i & ypred != class_i)
        result[class_i,"TP"]= result[class_i,"TP"] +TP
        result[class_i,"FP"]= result[class_i,"FP"] +FP
        result[class_i,"TN"]= result[class_i,"TN"] +TN
        result[class_i,"FN"]= result[class_i,"FN"] +FN
    }
}
for(class_i in noclasses){
    result[class_i,"prec"] = result[class_i,"TP"]*100/(result[class_i,"TP"]+result[class_i,"FP"])
    result[class_i,"recall"] = result[class_i,"TP"]*100/(result[class_i,"TP"]+result[class_i,"FN"])
    result[class_i,"f1"] = 2*result[class_i,"prec"]*result[class_i,"recall"]/(result[class_i,"prec"]
                                        +result[class_i,"recall"])
}
avgresult<-matrix(1,ncol=7)
colnames(avgresult)=c("TP","FP","TN","FN","prec","recall","f1")
avgresult[1,]=c(0)
avgresult[1,"prec"]=sum(result[,"prec"])/length(noclasses)
avgresult[1,"recall"]=sum(result[,"recall"])/length(noclasses)
avgresult[1,"f1"] = sum(result[,"f1"])/length(noclasses)
result=rbind(result,avgresult)
```

*Hình 2.3: Dự đoán và đánh giá hiệu quả quá trình phân lớp*

### *2.4. Kết quả thực nghiệm*

Bằng cách dùng gói FSinR trong R, hai thuộc tính *Regular artistic or sports activity*, *Do you have a partner* được loại bỏ, nghĩa là những thuộc tính này không quan trọng trong việc dự đoán kết quả thi của sinh viên.

Chúng tôi so sánh hiệu quả của việc lựa chọn đặc trưng và thuật toán sinh thêm phần tử bằng cách tính kết quả dự đoán kết quả thi của sinh viên bằng các phương pháp khác nhau, trước khi áp dụng lựa chọn đặc trưng **Org**, **SCUTM** và sau khi lựa chọn đặc trưng: **OrgF**, **SCUTMF**, theo các độ đo Precision, Recall và F1. Kết quả thu được, các giá trị Precision (%), Recall (%) và F1 (%) đối với phương pháp **Org** lần lượt là 37.56, 37.19 và 35.70; đối với phương pháp **SCUTM** là 40.02, 41.67 và 40.15; đối với phương pháp **OrgF** là 34.83, 38.47, 34.49; đối với phương pháp **SCUTMF** là 43.70, 45.1 và 44.04.

Như vậy, khi áp dụng SCUTM, kết quả dự đoán đã cải thiện đáng kể. Cụ thể, Precision tăng 2.46%, Recall tăng 4.48%, F1 tăng với tập dữ liệu gốc, và Precision tăng 8.87%, Recall tăng 6.63%, F1 tăng 9.55% khi có áp dụng lựa chọn đặc trưng. Và khi có áp dụng cả lựa chọn đặc trưng và SCUTM, kết quả thu được đạt giá trị tốt nhất, với Precision là 43.70%, Recall là 45.1 % và F1 là 44.04%.

## 3. Kết luận

Trong bài báo này, chúng tôi đã trình bày một phương pháp làm tăng hiệu quả dự đoán kết quả thi kết thúc học phần cho sinh viên, dựa vào kỹ thuật lựa chọn đặc trưng và sinh thêm phần tử để xử lý vấn đề mất cân bằng dữ liệu trên bài toán phân lớp dữ liệu mất cân bằng đa lớp và thực hiện đánh giá hiệu suất của phương pháp này trên tập dữ liệu mẫu thường được sử dụng. Chúng tôi đã thực nghiệm phương pháp này với sự hỗ trợ của các gói lệnh trong ngôn ngữ R. Kết quả thực nghiệm có so sánh với kết quả trên tập dữ liệu chưa xử lý đã cho thấy, phương pháp được đề xuất có hiệu quả tốt dựa trên các giá trị độ đo đánh giá hiệu suất Precision, Recall và F1. Dù vậy, do kích thước của tập dữ liệu dùng để thực nghiệm vẫn còn hạn chế, trong tương lai, chúng tôi sẽ tìm kiếm các tập dữ liệu thực khác để thực hiện việc kiểm thử đầy đủ hơn, từ đó có thể cho ra những kết quả chính xác và tin cậy hơn. Phương pháp này cũng có thể áp dụng để làm tăng hiệu quả dự đoán kết quả các môn học khác.

**Tài liệu tham khảo**

1. Cortez, Paulo & Silva, Alice, "Using data mining to predict secondary school student performance," *EUROSIS,* 2008.

2. Ying, Dahao & Ma, Jieming, "Student Performance Prediction with Regression Approach and Data Generation," *Applied Sciences,* vol. 14, p. 1148, 2024

3. Hashim, Ali & Akeel, Wid & Khalaf, Alaa, "Student Performance Prediction Model based on Supervised Machine Learning Algorithms," in *IOP Conference Series: Materials Science and Engineering*, 2020.

4. H. Sain and S. W. Purnami, "Combine Sampling Support Vector Machine for Imbalanced Data Classification," in *Procedia Comput. Sci.*, 2015.