ISSN 2354-0575

NGHIÊN CỨU MÔ HÌNH ENSEMBLES VÀ ÁP DỤNG DỰ ĐOÁN BỆNH THẬN TẠI BỆNH VIỆN ĐA KHOA ĐIỆN BIÊN

Nguyễn Văn Hậu[1], Nguyễn Thị Hải Năng[1], Nguyễn Tiến Tự[2], Nguyễn Ngọc Tiến[2]
1 Trường Đại học Sư phạm Kỹ thuật Hưng Yên
2 Bệnh viện Đa khoa tỉnh Điện Biên

Ngày tòa soạn nhận được bài báo: 20/10/2017
Ngày phản biện đánh giá và sửa chữa: 25/11/2017
Ngày bài báo được chấp nhận đăng: 05/12/2017

Tóm tắt:

Máy học hiện nay được áp dụng rộng rãi trong nhiều ứng dụng, bao gồm chẩn đoán y khoa, phát hiện thẻ tín dụng giả, phân tích thị trường chứng khoán, phân loại các chuỗi DNA, nhận dạng tiếng nói và chữ viết, dịch tự động, chơi trò chơi và cử động rô-bốt (robot locomotion). Hàng năm, cộng đồng nghiên cứu và cộng đồng công nghiệp đã có những cuộc hội thảo về chăm sóc sức khỏe sử dụng kiến thức của Máy học, Trí tuệ nhân tạo [7]. Vic Gundotra, cựu giám đốc tại Google và Microsoft, nhận định rằng trong vòng 5 năm tới, Máy học sẽ là trợ thủ đắc lực cho các bác sĩ [11]. Trong bài báo này chúng tôi dùng mô hình Cây quyết định (Decision Trees) của Máy học để dự đoán bệnh thận tại bệnh viên đa khoa tỉnh Điện Biên. Để cải thiện khả năng dự đoán, Chúng tôi tìm hiểu và cài đặt hai mô hình ensembles thường sử dụng và là những mô hình hiệu quả nhất trong Máy học: Random Forests và Gradient Boosted Trees.

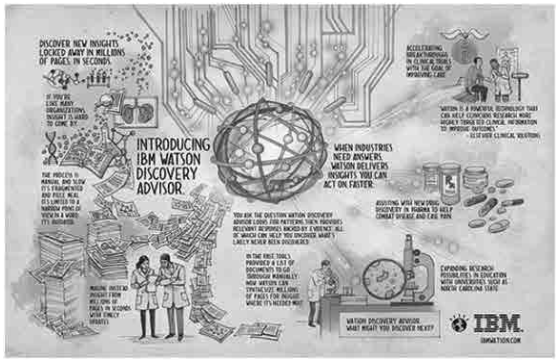
Từ khóa: *Cây quyết định, Random Forests, Gradient Boosted Trees, Mô hình Ensembles, Máy học.*

1. Giới thiệu

Lịch sử Máy học đã có từ lâu, nhưng nó thực sự có nhiều đột phá cho tới khi các nhà khoa học máy tính áp dụng kỹ thuật Deep Learning (học nhiều tầng) vào nhiều sản phẩm có tính ứng dụng hiệu quả trong thương mại và công nghiệp. Hiện nay, những công ty hàng đầu về công nghệ đều có những đội ngũ nghiên cứu và phát triển các sản phẩm Máy học: Google, Facebook, IBM, Intel, Amazon, Microsoft, Apple, v.v...

Enlitic là một công ty dùng deep learning, mang lại nhiều thành công nhất hiện nay cho Máy học, nhằm giúp bác sĩ khám bệnh nhanh hơn và chính xác hơn [8]. Mỗi khi một bác sĩ chẩn đoán cho bệnh nhân, họ đang giải quyết một tập dữ liệu phức tạp. Mục đích của mỗi trường hợp là đưa ra quyết định điều trị tối ưu dựa trên nhiều hình thức thông tin lâm sàng, như lịch sử bệnh nhân, triệu chứng, xét nghiệm và hình ảnh y khoa. Chất lượng và số lượng của dữ liệu này đang được cải thiện nhanh chóng - ước tính sẽ phát triển hơn 50 lần trong thập kỷ này, lên đến 25.000 petabyte trên toàn thế giới vào năm 2020. Đội ngũ chuyên gia y tế và các nhà khoa học dữ liệu hàng đầu thế giới muốn cải thiện kết quả dự báo bệnh nhân bằng sử dụng những dữ liệu nhằm khai thác thông tin của dự liệu. Enlitic sử dụng deep learning để tìm ra những tri thức từ hàng tỉ trường hợp lâm sàng. Enlitic xây dựng các giải pháp đểgiúp các bác sĩ tận dụng kiến thức chuyên sâu của cả cộng đồng y tế cho mỗi bệnh nhân.

Jensen Huang, giám đốc điều hành của Nvidia – công ty công nghệ nổi tiếng ở California, dự đoán rằng vấn đề chăm sóc sức khỏe và xe ô tô tự hành sẽ sớm được đảm nhiệm bởi Trí tuệ Nhân tạo [14]. Nhiều nhóm nghiên cứu Máy học cũng đang đầu tư vào lĩnh vực y tế và chăm sóc sức khỏe [9, 10]. IBM đang dần hiện thực một "ván cược lớn nhất": Thay vì mất nhiều ngày tra cứu hàng mớ hồ sơ bệnh án và tài liệu chuyên ngành để đưa ra kết luận chẩn đoán và trị liệu cho một ca bệnh với một xác suất sai không tránh khỏi, các bác sỹ chỉ cần nhập dữ liệu bệnh nhân cho Watson, được coi là cuộc cách mạng của IBM dùng Máy học, phân tích, so sánh với hàng trăm ngàn tài liệu trong kho kiến thức khổng lồ của nó rồi đưa ra gợi ý hướng điều trị chính xác chỉ sau vài giây [12]. Cuối cùng, chúng ta phải kể tới Watson, được kỳ vọng sẽ mang lại những chuyển biến tích cực cho ngành y tại Việt Nam [13].



Hình 1. *Sử dụng Máy học cho bài toán dữ liệu lớn làm đòn bẩy cho những cuộc cách mạng trong các lĩnh vực thiết yếu điển hình như chăm sóc sức khỏe chính là sứ mệnh và tầm nhìn của IBM hiện nay [13]*

Bài báo này sẽ nghiên cứu giải thuật Cây quyết định (Decision Trees), một giải thuật thông dụng trong Máy học, và áp dụng vào dự đoán bệnh nhân mắc bệnh thận tại tại bệnh viện Đa khoa tỉnh Điện Biên. Chúng tôi chọn Decision Trees vì nó có những ưu điểm như:

- Không cần tiền xử lý dữ liệu (normalization, standardization);

- Thuật toán làm việc hiệu quả khi các dữ liệu có scale hoàn toàn khác nhau, hoặc ngay cả sự pha trộn giữa các đặc tính nhị phân (binary) và liên tục (continuous);

- Thuật toán cây quyết định dễ dàng hình ảnh hóa và dễ hiểu cho cả những người không am hiểu về Máy học;

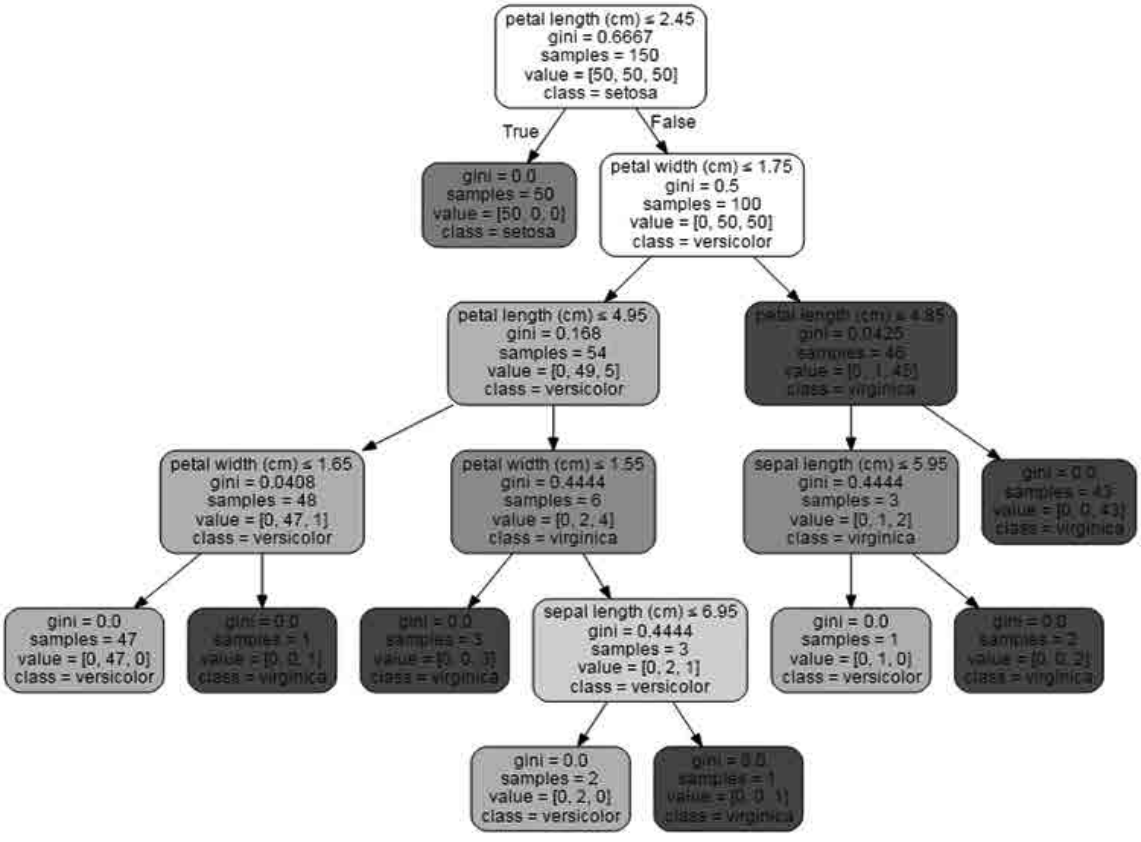
- Thuật toán cũng không thay đổi khi dữ liệu được mở rộng.

Những ưu điểm của Cây quyết định rất phù hợp với tính chất của bài toán: Khi bổ sung thêm các bệnh án, thuật toán không cần thay đổi nhiều. Quan trọng hơn cả là sự lý giải của mô hình có khả năng thuyết phục cả những người không phải là chuyên gia về Máy học. Đây chính là ưu điểm rất lớn của Cây quyết định, so với các mô hình khác. Để cải thiện khả năng dự đoán chúng tôi sử dụng mô hình Random Forests, là những phương pháp thường được dùng nhất trong Máy học hiện nay.

Phần còn lại của bài báo có cấu trúc như sau. Phần 2 sẽ giới thiệu về mô hình Cây quyết định (Decision Trees). Phần 3 sẽ trình bày mô hình Ensembles, trong khi phần 4 sẽ mô tả dữ liệu thu thập được. Phần 5 sẽ trình bày kết quả. Phần cuối cùng là kết luận.

2. Mô hình cây quyết định (Decision Trees)

Cây quyết định là một thuật toán Máy học có giám sát dùng cho cả bài toán phân lớp (classification) và hồi qui (regression). Cây quyết định rất hiệu quả trong nhiều lớp bài toán. Nó được sử dụng nhiều một phần vì nó dễ hiểu với mọi người, một phần vì nó có thể đưa ra lời giải thích chính xác cách thức mô hình đưa ra sự phân loại hay dự đoán đối với từng trường hợp. Chúng ta có thể theo dõi quá trình học và đưa ra dự đoán của cây quyết định thông qua các nhánh cây, và nó thực chất là một chuỗi (rất nhiều) các câu lệnh *if – then*. Để biết thêm, người đọc nên tham khảo chương 3 ở cuốn sách của Mitchell [1].



Hình 2. *Ví dụ cây quyết định cho bài toán hoa Ailen [3]*

Cây quyết định là một cấu trúc dạng cây, ở đó mỗi nút phía trong (internal node) biểu thị cho một câu hỏi (kiểm tra đặc trưng), mỗi nhánh biểu diễn một câu trả lời của câu hỏi đó, và mỗi lá (leaf node) biểu diễn một nhãn (class label). Mỗi một quyết định (decision) được xác định bằng một đường đi từ gốc tới lá (thông qua các thứ tự kiểm tra đặc trưng). Mỗi đường đi (từ gốc tới lá) thể hiện cho quyết định như vậy tương đương với một luật phân lớp.

Hình 2 minh họa việc áp dụng cây quyết định cho việc phân lớp 3 loài hoa Ailen (setosa, versicolor, và virginica). Mỗi nút lá là một nhãn (một trong 3 loài hoa). Mỗi một đường đi từ gốc tới lá là một sự giải thích (luật) cho việc định nhãn. Như

chúng ta thấy, các quyết định trên cây rất dễ hiểu và dễ giải thích, vì chúng có thể hình ảnh hóa được.

Ý tưởng của quá trình xây dựng cây quyết định là việc tìm ra các câu hỏi (đặc trưng) để câu trả lời sẽ cho nhiều thông tin liên quan tới dự đoán nhất. Giả sử, nếu một câu hỏi *yes/no* mà kết quả luôn cho đúng khi trả lời "*yes*", và sai khi câu là lời là "*no*" (hoặc ngược lại), thì đây là một câu hỏi tuyệt vời, vì nó cho chúng ta rất nhiều thông tin. Ngược lại, nếu câu hỏi *yes/no* mà kết quả không chắc đúng khi trả lời "*yes*", và cũng không chắc đúng khi trả lời "*no*", thì câu hỏi đó không cho chúng ta nhiều thông tin.

Vậy, làm thế nào để chúng ta có thể đo được "thông tin"? Để định lượng được thông tin, người ta dùng entropy, được giới thiệu bởi nhà toán học Shannon. Entropy thường dùng với nghĩa hỗn loạn (hay không chắc chắn). Trong phần này, chúng ta dùng entropy để đánh giá độ không chắc chắn liên quan tới dữ liệu. Mục đích của chúng ta là chia ra thành các tập con mới có độ entropy nhỏ dần. Có một tiêu chí khác để tách hai tập dữ liệu, đó là dựa vào chỉ số Gini. Để biết thêm, người đọc có thể đọc chương 3 ở [1] và chương 7 ở [2].

3. Mô hình Ensembles cho cây quyết định

3.1. Mô hình Ensembles

Tại sao chúng ta không thể huấn luyện thuật toán Máy học trên tập dữ liệu và sử dụng các dự đoán từ cùng một tập dữ liệu này để đánh giá các thuật toán Máy học? Câu trả lời đơn giản là overfitting, một hiện tượng thường gặp trong Máy học khi thuật toán thực thi tốt trên tập dữ liệu huấn luyện, nhưng lại kém trên tập dữ liệu mới. Hãy tưởng tượng một thuật toán ghi nhớ mọi quan sát trong quá trình huấn luyện. nếu bạn đánh giá thuật toán học máy của bạn trên cùng một bộ dữ liệu được sử dụng để huấn luyện thuật toán, một thuật toán như thế này sẽ có một điểm số hoàn hảo trên tập dữ liệu huấn luyện. Nhưng thuật toán đó khi dự đoán trên dữ liệu mới (unseen data) lại rất kém. Tóm lại, overfitting xảy ra khi nó thực hiện rất tốt trên mô hình huấn luyện (training set) nhưng lại cho dự đoán kém trên tập dữ liệu mới (test sets). Mà mục tiêu của Máy học là cần tạo ra các mô hình có khả năng dự đoán tốt cho những dữ liệu mới (unseen data).

Một nhược điểm rất lớn của cây quyết định là mô hình dễ bị rơi vào trạng thái overfitting. Chính vì vậy, trong hầu hết các ứng dụng, các phương pháp ensembles thường được dùng thay cho việc dùng đơn lẻ mô hình Cây quyết định.

Ensembles là phương pháp kết hợp nhiều mô hình Máy học nhằm tạo ra một mô hình mạnh hơn. Có nhiều mô hình ensembles trong Máy học, tuy nhiên có hai mô hình ensembles dùng cho nhiều loại ứng dụng khác nhau và cùng dùng Cây quyết định đã chứng tỏ được tính hiệu quả cao: Random Forests và Gradient Boosted Decision Trees.

3.2. Random Forests

Random Forests dùng để khắc phục trạng thái overfitting, một nhược điểm của Decision Trees. Thuật toán tạo ra một tập các Cây quyết định (Decision Tree), trong đó các cây có sự khác nhau. Sự ra đời của thuật toán xuất phát từ ý tưởng mỗi cây sẽ có thể đưa ra những dự đoán rất tốt, nhưng nó lại dễ bị overfitting. Do vậy, nếu chúng ta tạo ra nhiều cây, tất cả các cây này đều dự đoán tốt và nếu bị overfitting thì nó sẽ overfitiing theo nhiều hướng khác nhau; chúng ta có thể giảm tổng overfitting bằng cách lấy trung bình của tập các cây đó. Điều chú ý ở đây là, Random Forests vẫn tận dụng được tính hiệu quả của mô hình Decsision Tree, trong khi việc giảm overfitiing có thể được tính toán/chỉ ra bằng toán học.

Để xây dựng Random Forests, chúng ta cần tạo ra rất nhiều cây quyết định. Mỗi cây ngoài nhiệm vụ đảm nhận nhiệm vụ dự đoán, nó còn phải khác so với các cây còn lại. Random Forests lấy tên các cây từ việc trích ngẫu nhiên trong quá trình xây dựng cây để đảm bảo mỗi cây là khác nhau. Có hai cách tạo ra cây trong Random Forests: 1) lựa chọn các quan sát để tạo cây; 2) lựa chọn đặc tính trong quá trình tách (split).

Trong bài báo này, chúng tôi sẽ sử dụng thư viện Scikit – learn [4]. Scikit-learn (viết tắt là sklearn) là một thư viện mã nguồn mở dành cho học máy - một ngành trong trí tuệ nhân tạo, rất mạnh mẽ và thông dụng với cộng đồng Python, được thiết kế trên nền NumPy và SciPy. Scikit-learn chứa hầu hết các thuật toán machine learning hiện đại nhất.

Để xây dựng một cây, người dùng cần dùng một bootstrap mẫu cho tập dữ liệu. Số cây (*n_samples* cây) được tạo ra một cách ngẫu nhiên. Dữ liệu cho mỗi cây sẽ lớn như dữ liệu ban đầu, nhưng một số quan sát có thể thiếu, trong khi một số khác có thể bị lặp lại. Sau đó mỗi cây quyết định sẽ được tạo từ mỗi tập dữ liệu mới đó. Tuy nhiên, so với thuật toán cây quyết định, thuật toán sẽ có sự biến đổi một chút. Cụ thể, thay cho việc tìm kiếm một nút tốt nhất, thuật toán sẽ lựa chọn ngẫu nhiên một tập con các đặc tính, và tìm ra đặc tính tốt nhất trong trong tập con đó. Tổng các đặc tính được lựa chọn sẽ được điều chỉnh thông qua tham số *max_features*. Như vậy, việc lựa chọn tập con đặc tính được lặp lại tách biệt trên mỗi nút, nên mỗi nút trên một cây sẽ tạo ra quyết định dùng một tập con khác nhau của các đặc tính, cùng với việc sử dụng bootstrap mẫu sẽ tạo cho các cây quyết định khác nhau.

Một vấn đề được đặt ra là việc lựa chọn tham số *max_features*. Nếu chúng ta thiết lập *max_features = n_features*, thì điều đó đồng nghĩa với việc tại một nút (để tách) chúng ta sẽ lựa chọn tất cả các thuộc tính trong tập dữ liệu, cộng thêm với việc không lựa chọn ngẫu nhiên đặc tính (mà dựa vào thuật toán tính độ hỗn loạn giống như trong mô hình Decision Trees). Nếu chúng ta thiết lập *max_features* = 1, khi đó các phép tách sẽ không có sự lựa chọn. Do vậy, nếu thiết lập tham số *max_features* lớn, Random Forests sẽ có nhiều cây cùng sự tương đồng, và chúng có thể thỏa mãn dữ liệu

dễ dàng, bằng việc dùng những đặc tính khác biệt nhất. Trong khi nếu thiết lập tham số *max_features* nhỏ, Random Forestscác cây sẽ ít tương đồng hơn, và mỗi cây sẽ có độ sâu đủ lớn thể thỏa mãn dữ liệu. Người đọc có thể tham khảo thêm ở [5,6].

Để có một dự đoán dùng Random Forests, đầu tiên thuật toán sẽ phải quan tâm tới mọi dự đoán của các cây trong rừng cây đó. Tiếp đó, sẽ có chiến lược khác nhau, tùy thuộc vào kiểu dự đoán:

- Với bài toán regression, chúng ta có thể tính trung bình các kết quả để đưa ra dự đoán cuối cùng.
- Với bài toán classification, chiến lược "soft voting" được áp dụng. Trong đó mỗi cây sẽ đưa ra một "soft" prediction, tức là xác suất cho mỗi kết quả được đưa ra. Xác suất dự đoán sẽ được tính bằng trung bình tất cả các cây, và lớp có xác suất cao nhất sẽ được.

3.3. Gradient Boosted Trees (Gradient Boosting Machines)

Mô hình Gradient Boosted Trees (đôi khi còn có tên Stochastic Gradient Boosting hay Gradient Boosting Machines) là một trong những thuật toán phức tạp nhất và hiệu quả của kĩ thuật ensembles.

Khác với Random Forests, *Gradient Boosted Trees* tạo ra các cây một cách tuần tự, trong đó mỗi cây sau sẽ cố gắng khắc phục những lỗi của các cây trước. Sẽ không có sự ngẫu nhiên trong quá trình tạo cây ở *Gradient Boosted Trees*; thay vào đó, kĩ thuật pre-pruning sẽ được dùng. Cây trong *Gradient Boosted Trees* thường có độ cao thấp (từ 1 tới 5), điều này làm mô hình chiếm ít bộ nhớ và cho kết quả nhanh hơn.

Ý tưởng chính của *Gradient Boosted Trees* là kết hợp nhiều mô hình đơn giản (weak learners), những cây thấp (shallow trees). Mỗi cây có thể dự đoán tốt cho từng phần dữ liệu, và sau khi kết hợp nhiều cây lại sẽ tăng khả năng dự đoán cho mô hình. Một điểm đáng chú ý là *Gradient Boosted Trees* thường nổi trội hơn các thuật toán Máy học khác và thường được dùng rộng rãi trong ứng dụng thực tế. Tuy nhiên, việc thiết lập các tham số sẽ yêu cầu chặt chẽ hơn so với Random Forests. Một tham số quan trọng là *learning_rate*, nó sẽ kiểm soát mức độ mỗi cây sẽ sửa lỗi ra sao. Việc tăng *learning_rate* và *n_estimators* sẽ làm tăng độ phức tạp của mô hình, vì mô hình sẽ có nhiều việc phải làm hơn nhằm sửa lỗi cho các cây lần lượt được tạo ra.

4. Dữ liệu và chương trình

4.1. Dữ liệu

Dữ liệu trong bài này được thu thập từ các bệnh án tại bệnh viện đa khoa tỉnh Điện Biên. Số bệnh nhân này được làm xét nghiệm chẩn đoán bệnh trong 2 năm 2015 – 2016 với tổng số lần xét nghiệm là 166.823 lượt trong đó mẫu đạt tiêu chuẩn là 3.648 lần. Mặc dù có rất nhiều thông tin, nhưng chúng tôi lựa chọn 15 đặc tính đều ở dạng số, và cột cuối cùng là đặc tính cần dự đoán:

Bảng 1. *Dữ liệu thu thập từ bệnh viện đa khoa Điện Biên*

Số TT	Đặc tính	Giải thích
1	age	Tuổi
2	sex	Gới tính
3	wbc	White blood cell (bạch cầu máu)
4	ly	Lymphocytes (bạch cầu Lympho)
5	ne	Newtrophylia (bạch cầu đoạn trung tính)
6	rbc	Red blood cell (hồng cầu máu)
7	hgb	Hemoglobin (HGB - huyết sắc tố)
8	hct	Hematocrit (Hct – thể tích khối hồng cầu)
9	plt	Platelet (tiểu cầu)
10	na	Natri máu
11	kl	Kali máu
12	prtp	Protein máu toàn phần
13	al	Albumin
14	ur	Urê máu
15	cr	Creatinin
16	absence	1: mắc bệnh; 0: không mắc bệnh

4.2. Chương trình

Trong phần này, chúng tôi sẽ cài đặt chương trình, chạy thử và tổng hợp các kết quả. Chú ý rằng Chương trình 1 cần cho các chương trình phía sau (2, 3, 4), và Chương trình 5 cần cả chương trình 4. Chương trình 1 sau đây sẽ hiển thị những thông tin cơ bản về dữ liệu (kích cỡ của số hàng, số cột) và 5 dòng dữ liệu đầu tiên:

```
from pandas import read_csv
import os

duongDan = os.getcwd() + '\data\\
than_final.csv'
tenCot = ['age','sex','WBC','LY',
'NE','RBC','HGB','HCT','PLT','Na'
,'KL', 'Protein','Albumin','Ure'
,'Creatinin','absence']
duLieu = read_csv(duongDan,
names=tenCot)
from sklearn import preprocessing
print (duLieu.shape) # (3648, 16):
Dữ liệu có 3648 hàng và 16 cột
print (duLieu.head())
# Hiển thị 5 hàng đầu tiên
from sklearn import preprocessing

maTran= duLieu.values
X = maTran[:,:-1]
y = maTran[:,-1]
dieuChinh = preprocessing.
MinMaxScaler(feature_range= (0,1))
```

```
X_dieuChinh = dieuChinh.fit_
transform(X)
```

Chương trình 1: Kết nối và hiển thị thông tin dữ liệu.

Kết quả của Chương trình 1 trên sẽ cho ra kích cỡ của dữ liệu và 5 hàng đầu tiên:

```
(3648, 16)
age  sex    WBC    LY    NE   RBC  HGB   HCT
PLT      Na     K  Protein
0   78    1   6.13  14.4  77.7  2.98   88
25.5   98.0  139.80  3.70    71.28
1   16    0   7.69  13.6  73.5  3.64   81
25.5  249.0  141.90  3.70    60.77
2   51    0  10.13  14.8  82.2  3.74  127
35.6  179.0  138.95  3.42    74.10
3   79    0   4.33  25.5  62.6  3.34  101
30.8  260.0  134.35  3.06    73.90
4   42    1   3.53  13.0  75.0  1.58   47
13.0   52.0  125.50  6.82    66.50
   Albumin     Ure  Creatinin  absence
0     36.3   2.988      55.43        1
1     28.2  18.003     566.34        1
2     38.9   4.200      94.00        1
3     33.7   6.400     476.00        1
4     32.6  50.400    2246.00        1
```

Sau đây là mô hình Decision Trees cho bài toán dự báo bệnh:

```
from sklearn.model_selection import
train_test_split
from sklearn.tree import
DecisionTreeClassifier
tree = DecisionTreeClassifier()

X_train, X_test, y_train, y_test =
train_test_split(X_dieuChinh, y,
test_size=0.33, random_state=0)
tree.fit(X_train, y_train)
print("accuracy on training set: %f"
% tree.score(X_train, y_train))
print("accuracy on test set: %f" %
tree.score(X_test, y_test))
```

Chương trình 2: Chương trình dự đoán bệnh Thận ở bệnh viện đa khoa Điện Biên sử dụng mô hình Decision Trees.

Chương trình 2 cho kết quả:

```
accuracy on training set: 1.000000
accuracy on test set: 0.930921
```

Chương trình 2 cho ta thấy rõ hiện tượng overfitting xảy ra. Độ chính xác trên tập huấn luyện (training set) là 100%, trong khi trên tập kiểm tra (test set) là 93%. Chương trình 3 sau đây sẽ tạo ra một Random Forests, như đã trình bày ở phần trước, nhằm tránh overfitting để tăng độ chính xác cho dự đoán.

Trong thư viện sklearn có sử dụng rất nhiều các tham số trong mô hình *RandomForestClassifier()*. Tuy nhiên chúng ta chỉ quan tâm 4 tham số:

- *bootstrap*: boolean, optional (default=True), để xác định xem có dùng các mẫu bootstrap khi dựng cây hay không.

- *max_features*: int, float, string or None, optional (default="auto"), để xác định số đặc tính được dùng để chọn phép tách tốt nhất:

+ nếu 'auto', khi đó *max_features* = sqrt(*n_features*)

+ nếu float, khi đó *max_features* = int(*n_features* * *n_features*)

- n_estimators: integer, optional (default=10), số cây trong Random Forests.

- criterion: string, optional (default="gini"), dùng để xác định chất lượng của phép tách, chúng ta có thể có 2 lựa chọn "gini" hoặc "entropy".

```
from sklearn.ensemble import
RandomForestClassifier
for i in (100,500,1000,1500,2000):
# tách dữ liệu ra thành 2 tập: huấn
luyện (training) và kiểm tra (test)
X_train, X_test, y_train, y_test =
train_test_split(X_dieuChinh, y,
test_size=0.25, random_state=1)
# tạo forest If "auto", then max_
features=sqrt(n_features).
forest =
RandomForestClassifier(bootstrap=True,
n_estimators=i, criterion='gini',
max_features= 'auto')

# huấn luyện mô hình trên tập huấn luyện
forest.fit(X_train, y_train)
print("accuracy on training set: %f"
% forest.score(X_train, y_train))
print("accuracy on test set: %f" %
forest.score(X_test, y_test))
```

Chương trình 3: Chương trình dự đoán bệnh Thận ở bệnh viện đa khoa Điện Biên sử dụng mô hình Random Forests.

Bảng 2 sau đây tổng hợp kết quả của Chương trình 3 sau khi thay đổi hai tham số quan trọng n_estimators (là số cây mà mô hình Random Forests tạo ra) và *max_features* (số đặc tính được tham gia vào quá trình tách tại các nút).

Bảng 2. *Kết quả khi chạy chương trình dùng mô hình ensembles*

n_estimators/ *max_features*	'auto' Training| Test	0.5 Training| Test	0.75 Training| Test	1.0 Training| Test
100	0.9997 | 0.9616	0.9996 | 0.9616	0.9996 | 0.9509	0.9996 | 0.9510
500	0.9997 | 0.9638	0.9996 | 0.9627	0.9996 | 0.9518	0.9996 | 0.9518
1000	0.9997 | 0.9638	0.9996 | 0.9616	0.9996 | 0.9509	0.9996 | 0.9510

1500	0.9997 | 0.9649	0.9996 | 0.9627	0.9996 | 0.9509	0.9996 | 0.9510
2000	0.9997 | 0.9649	0.9996 | 0.9627	0.9996 | 0.9594	0.9996 | 0.9518

Theo kết quả của Bảng 2, Random Forests sẽ cho dự đoán tốt nhất là 96.49% khi số cây tạo ra là 1500 hoặc 2000 cây.

Chương trình sau đây cài đặt mô hình Gradient Boosted Trees cho bài toán dự báo bệnh thận:

```
from sklearn.ensemble import
GradientBoostingClassifier
gbrt = GradientBoostingClassifier(n_
estimators = 1000, learning_rate=
0.07, max_features= 'auto', random_
state=0, max_depth=4)
X_train, X_test, y_train, y_test =
train_test_split(X_dieuChinh, y,
test_size=0.25,  random_state=0)
gbrt.fit(X_train, y_train)
print("accuracy on training set: %f"
% gbrt.score(X_train, y_train))
print("accuracy on test set: %f" %
gbrt.score(X_test, y_test))
```

Chương trình 4: Chương trình dự đoán bệnh Thận ở bệnh viện đa khoa Điện Biên sử dụng mô hình Gradient Boosted Trees.

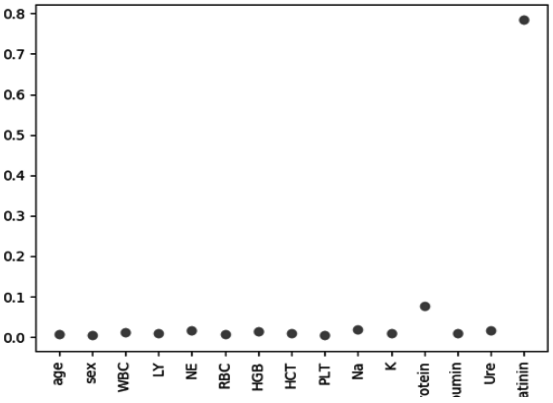
Chương trình 4 cho kết quả:

```
accuracy on training set: 1.000000
accuracy on test set: 0.963816
```

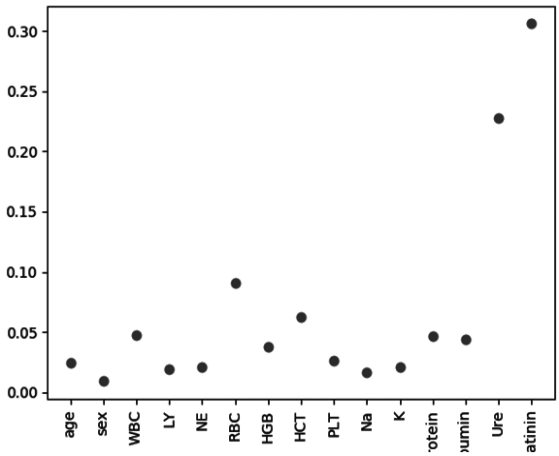
Để xem và đánh giá mức độ quan trọng của các đặc tính ở các mô hình, chúng tôi có dựng đồ thị quan sát. Chương trình 5 xây dựng đồ thị (Hình 4) cho mô hình Gradient Boosted Trees.

```
import matplotlib.pyplot as plt
plt.plot(gbrt.feature_importances_, 'o')
plt.xticks(range(X.shape[1]),
tenCot, rotation=90)
plt.show()
```

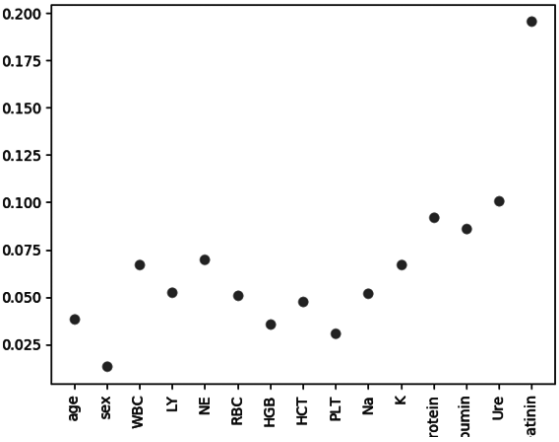
Chương trình 5: Chương trình hiển thị mức độ quan trọng của các đặc tính trong mô hình Gradient Boosted Trees.



Hình 3. *Mức độ quan trọng của các đặc tính trong mô hình Decision Trees*



Hình 4. *Mức độ quan trọng của các đặc tính trong mô hình Random Forests*



Hình 5. *Mức độ quan trọng của các đặc tính trong mô hình Gradient Boosted Trees*

Quan sát Hình 3, 4, và 5, chúng ta nhận thấy rằng đặc tính Creatinin được đánh giá quan trọng nhất ở cả 3 mô hình, và đặc biệt quan trọng ở mô hình Decision Trees (xấp xỉ 0.8). Ngoài đặc tính Creatinin ra, các đặc tính ở mô hình Decision Trees không có sự chênh lệch nhiều. Trong khi đó ở mô hình Random Forests có sự khác biệt hơn; đáng chú ý là đặc tính Ure rất được coi trọng (gần 0.25 so với 0.3 của Creatinin). Với mô hình Gradient Boosted Trees thì hơi khác, Ure được coi trọng sau Creatinin, nhưng chỉ nhỉnh hơn Protein và Albumin không nhiều.

5. Kết luận

Bài báo tiến hành tìm hiểu mô hình cây quyết định (Decision Trees). Cùng với những ưu điểm như dễ hiểu và không cần tiền xử lý dữ liệu, Decision Trees có một nhược điểm quan trọng là overfitting. Để xử lý vấn đề này, chúng tôi tìm hiểu và sử dụng mô hình ensembles. Cụ thể chúng tôi đã tìm hiểu và cài đặt hai mô hình ensembles được coi là hiệu quả nhất: Random Forests và Gradient Boosted Trees.

Bài báo có hai đóng góp. Thứ nhất, chúng tôi đã tiến hành thu thập các dữ liệu từ các bệnh án của bệnh viên đa khoa Điện Biên. Chúng tôi đã lọc bỏ những bệnh án không đủ dữ liệu và những cột đặc tính không cần thiết. Thêm nữa, chúng tôi cũng phải xử lý một số thông tin thiếu. Cuối cùng dữ liệu gồm 3648 bệnh án với 15 đặc tính độc lập và 1 đặc tính dự đoán (bị hay không bị). Thứ hai, chúng tôi đã cài đặt và có được kết quả dự đoán với mô hình *Decision Trees* với độ chính xác là *93.09%*; với hai mô hình ensembles cho ra kết quả tốt hơn với độ chính xác tương ứng là: *96.49%* và *96.38%*.

Công việc sắp tới của chúng tôi là mở rộng kết quả đã đạt được của bài báo này bằng hai công việc. Thứ nhất, chúng tôi muốn tìm hiểu thêm các mô hình khác trong Máy học (machine learning algorithms), ngoài hai mô hình mà bài báo này đã bàn tới. Cụ thể chúng tôi muốn tìm hiểu và so sánh kết quả với một số mô hình học có giám sát: Decision Trees, K – nearest neighbor, và Neuron Network. Thứ hai, chúng tôi sẽ rà soát lại và trao đổi với các bác sĩ chuyên khoa để thực thi thêm các bệnh án mới, nhằm kiểm định lại chương trình của chúng tôi. Một hướng khác chúng tôi cũng muốn áp dụng mô hình đã xây dựng vào dữ liệu mới (ở bệnh viện khác, ở vùng khác) để có thêm những thông tin về tính hiệu quả của mô hình.

Sau khi đã tìm hiểu và so sánh như vậy, chúng tôi tin rằng sẽ tìm được mô hình thực sự phù hợp với bài toán dự đoán bệnh thận. Chúng tôi nhận thức được rằng việc chuẩn đoán bệnh là một công việc khó, đòi hỏi tính chuyên môn cao và rất quan trọng, vì nó liên quan tới sức khỏe của con người. Tuy nhiên, chúng tôi cũng nhận thấy rằng với sự phát triển của khoa học máy tính gần đây, đặc biệt là ngành Trí tuệ Nhân tạo và Máy học, đã có rất nhiều ứng dụng quan trọng đã có hiệu quả cao. Do vậy, chúng tôi tin rằng nghiên cứu có thể được mở rộng và có thể ứng dụng vào hỗ trợ cả bệnh nhân và bác sĩ của Bệnh viện Đa khoa tỉnh Điện Biên.

Lời cảm ơn

Bài báo này được tài trợ bởi trung tâm Nghiên cứu ứng dụng Khoa học và Công nghệ, trường Đại học Sư phạm Kỹ thuật Hưng Yên, mã số UTEHY.T026.P1718.01. Các tác giả cũng cảm ơn những bác sĩ tại bệnh viện đa khoa Điện Biên đã tận tình giải thích những kiến thức chuyên ngành.

Tài liệu tham khảo

[1]. Tom M. Mitchell, *Machine Learning*, McGrawHill, 1997, 432 pages, ISBN: 0070428077.
[2]. Toby Segaran, *Programming Collective Intelligence*, O'Reilly August, 2007, 362 pages, ISBN-10: 0596529325.
[3]. http://archive.ics.uci.edu/ml/
[4]. http://scikit-learn.org/
[5]. http://scikit.learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
[6]. Andreas C. Müller, Sarah Guido, *Introduction to Machine Learning with Python: A Guide for Data Scientists*, O'Reilly Media; 1st (October 21, 2016) 394 pages, ISBN-10: 1449369413
[7]. http://mucmd.org/
[8]. http://www.enlitic.com/
[9]. http://staging.csml.ucl.ac.uk/clinics/
[10]. https://www.data-service-alliance.ch/activities
[11].https://www.recode.net/2016/12/5/13837908/machine-learning-doctors-vic-gundotra-recode-podcast
[12].http//genk.vn/ibm-va-cuoc-cach-mang-tri-tue-nhan-tao-mang-ten-watson-20160830152953753.chn
[13].http://genk.vn/tri-tue-nhan-tao-ibm-watson-se-duoc-trien-khai-voi-ngan-hang-benh-vien-va-truyen-hinh-cap-tai-viet-nam-20170420183145415.chn
[14]. https://www.technologyreview.com/s/607831/nvidia-ceo-software-is-eating-the-world-but-ai-is-going-to-eat-software/

STUDY ENSEMBLE METHODS AND PREDICT KIDNEY DISEASE FOR DIEN BIEN GENERAL HOSPITAL

Abstract:

Machine learning has now been widely applied in the modern life, ranging from medical diagnostics to counterfeit credit cards, stock market analysis, DNA sequencing, speech and writing recognition, auto translate, play games and robot locomotion. Every year, the research community and the industrial community have conducted health care seminars using the knowledge of Machine Learning, Artificial Intelligence [7]. Vic Gundotra, a former director at Google and Microsoft, said that within next five years, Machine Learning would be a powerful assistant to doctors. In this paper, we use Decision Trees model to predict kidney disease for the general hospital Dien Bien. In order to improve the accuracy, we study and install two models of ensembles which commonly used and are the most effective models in machine learning: Random Forests and Gradient Boosted Trees.

Keywords: *Decision Trees, Random Forests, Gradient Boosted Trees, Ensemble methods, Machine learning.*