

TRƯỜNG ĐẠI HỌC BÁCH KHOA TP HCM
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH



KHAI PHÁ DỮ LIỆU

BÀI TẬP LỚN

Hệ thống dự đoán tình trạng giao thông ở TP. HCM

NHÓM 5

GV hướng dẫn: Trần Minh Quang
SV thực hiện: Đinh Minh Tân – 1613074
Lương Tuấn Kiệt – 1611695
Lê Đức Mạnh – 1611985
Trần Trương Tuấn Phát – 1612541
Cao Nguyên Bình – 1610228
Nguyễn Việt Hưng – 1611441

Tp. Hồ Chí Minh, Tháng 11/2018



Mục lục

1	Giới thiệu	3
2	Kiến thức nền tảng	3
2.1	Hệ thống Open Street Map	3
2.1.1	Giới thiệu	3
2.1.2	Lịch sử hình thành	3
2.1.3	So sánh với Google Maps	3
2.2	Hệ thống giao thông thông minh Trường Đại học Bách Khoa (ITS)	4
2.3	Phân loại dữ liệu	4
2.3.1	Phân loại bằng cây quyết định	4
2.3.1.a	Giải thuật C4.5	5
2.3.1.b	Cây quyết định J48 trong Weka	5
2.3.2	Phân loại bằng mạng nơ-ron ANN	5
2.3.3	Cải thiện độ chính xác của bài toán phân lớp có các lớp dữ liệu không cân bằng	7
2.3.3.a	Oversampling và undersampling	7
2.3.3.b	Di chuyển ngưỡng - Threshold-moving	9
2.3.3.c	Ensemble methods	9
3	Phân tích đề tài	10
3.1	Tình trạng giao thông tại Việt Nam	10
3.2	Giải pháp đề xuất và đề tài bài tập lớn	11
3.3	Phạm vi nghiên cứu	11
3.4	Kỳ vọng đạt được	12
4	Giải quyết vấn đề	13
4.1	Tổng quan về dữ liệu đầu vào	13
4.2	Tiền xử lý dữ liệu	14
4.2.1	Lọc dữ liệu (Data Selection)	14
4.2.2	Biến đổi dữ liệu (Data Transformation)	14
4.2.3	Gắn nhãn dữ liệu	14
4.2.4	Làm sạch dữ liệu (Data Cleaning)	17
4.2.4.a	Loại bỏ outliers, extreme	17
4.2.4.b	Vấn đề: Dữ liệu ở trường speed bằng 0	17
4.2.5	Cân bằng các lớp trong dataset	19
4.3	Khai phá dữ liệu	19
4.3.1	Phân loại bằng cây quyết định	19
4.3.1.a	Model	19
4.3.1.b	Test	21
4.3.2	Phân loại bằng một số phương pháp khác	23
4.4	Đánh giá kết quả	26
4.5	Hiển thị dữ liệu dự đoán lên bản đồ	26
5	Tổng kết và định hướng trong tương lai	26
5.1	Đánh giá kết quả đạt được	26
5.2	Vấn đề tồn tại và định hướng trong tương lai	26
	Tài liệu tham khảo	27

Danh sách hình vẽ

1	Ảnh chụp giao diện web hệ thống ITS	3
2	Ảnh chụp giao diện web hệ thống ITS	4
3	Cây quyết định chỉ ra khả năng mua máy tính của từng loại khách hàng	5
4	Mạng nơ-ron	6
5	Pseudo code của giải thuật SMOTE.	8
6	Pseudo code của giải thuật bagging.	9
7	Đoạn đường của đường Lý Thường Kiệt được phân tích	11
8	Đoạn đường của đường Trường Chinh được phân tích	12
9	Hình ảnh dữ liệu của ngày 6/4/2018	13
10	Thông tin về dữ liệu của đoạn đường Trường Chinh khi lọc dữ liệu, biến đổi dữ liệu và gắn nhãn dữ liệu	15
11	Hình ảnh một số giá trị trong bảng dữ liệu của đoạn đường Trường Chinh khi lọc dữ liệu, biến đổi dữ liệu và gắn nhãn dữ liệu	15
12	Thông tin về dữ liệu của đoạn đường Lý Thường Kiệt khi lọc dữ liệu, biến đổi dữ liệu và gắn nhãn dữ liệu	16
13	Hình ảnh một số giá trị trong bảng dữ liệu của đoạn đường Lý Thường Kiệt khi lọc dữ liệu, biến đổi dữ liệu và gắn nhãn dữ liệu	16
14	Dataset của đoạn đường Trường Chinh sau khi làm sạch dữ liệu	18
15	Dataset của đoạn đường Lý Thường Kiệt sau khi làm sạch dữ liệu	18
16	Dataset của đoạn đường Lý Thường Kiệt sau khi cân bằng các class bằng oversampling	19
17	Dataset của đoạn đường Trường Chinh sau khi cân bằng các class bằng oversampling	20
18	Hyperparameters dùng để training model	21
19	Kết quả training đường Lý Thường Kiệt	22
20	Kết quả training đường Trường Chinh	22
21	Kết quả training đường Lý Thường Kiệt bằng Neural Network	23
22	Kết quả training đường Trường Chinh bằng Neural Network	24
23	Kết quả training đường Lý Thường Kiệt bằng BayesNet	25
24	Kết quả training đường Trường Chinh bằng BayesNet	25

1 Giới thiệu

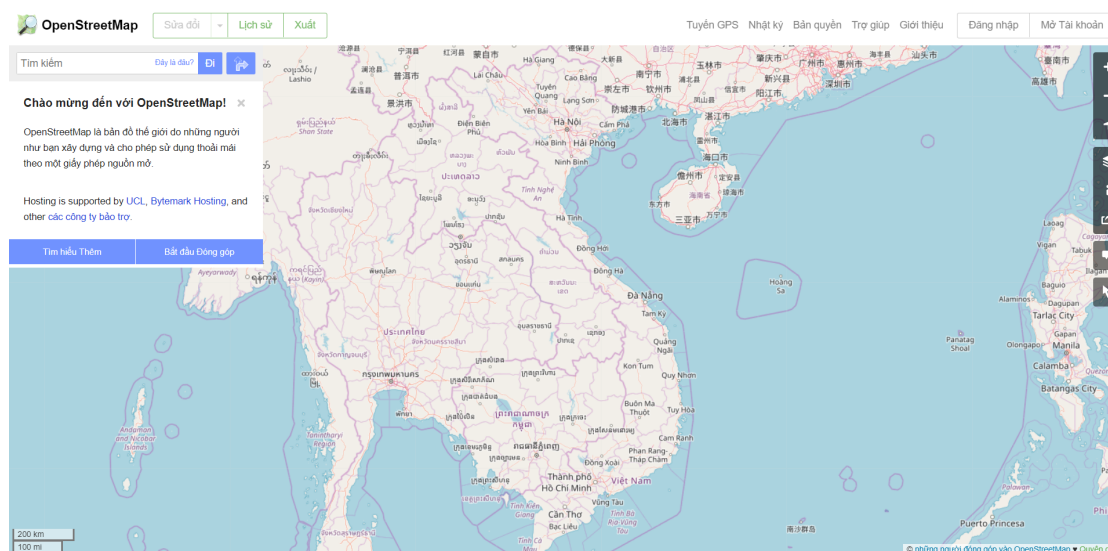
2 Kiến thức nền tảng

2.1 Hệ thống Open Street Map

2.1.1 Giới thiệu

Openstreetmap - OSM: là một dịch vụ bản đồ thế giới trực tuyến có nội dung mở. Tất cả mọi người đều có thể chỉnh sửa bản đồ thế giới cùng nhau nhằm cung cấp và chia sẻ dữ liệu địa lý.

Mô hình của OSM tương đối giống mô hình của bách khoa toàn thư Wikipedia, các cộng tác viên cũng như người dùng dễ dàng sử dụng các công cụ được cung cấp trên OSM để trực tiếp chỉnh sửa bằng cách thao tác trên máy tính để bàn, laptop và cả các thiết bị di động.



Hình 1: Ảnh chụp giao diện web hệ thống ITS

2.1.2 Lịch sử hình thành

OSM được tạo ra bởi Steve Coast ở Anh vào năm 2004, nó được lấy cảm hứng từ sự thành công của Wikipedia và vị thế độc quyền trong việc cung cấp dữ liệu bản đồ của chính phủ các nước. Kể từ đó, OSM đã phát triển đến hơn 1 triệu người dùng đăng ký. Người dùng có thể thu thập và đóng góp dữ liệu bằng cách sử dụng thiết bị GPS, chụp ảnh trên không và các nguồn dữ liệu miễn phí khác. Những dữ liệu này sau đó được phát hành theo giấy phép cơ sở dữ liệu mở - Open Database License. OSM được hỗ trợ bởi OpenStreetMap Foundation, một tổ chức phi lợi nhuận tại Anh.

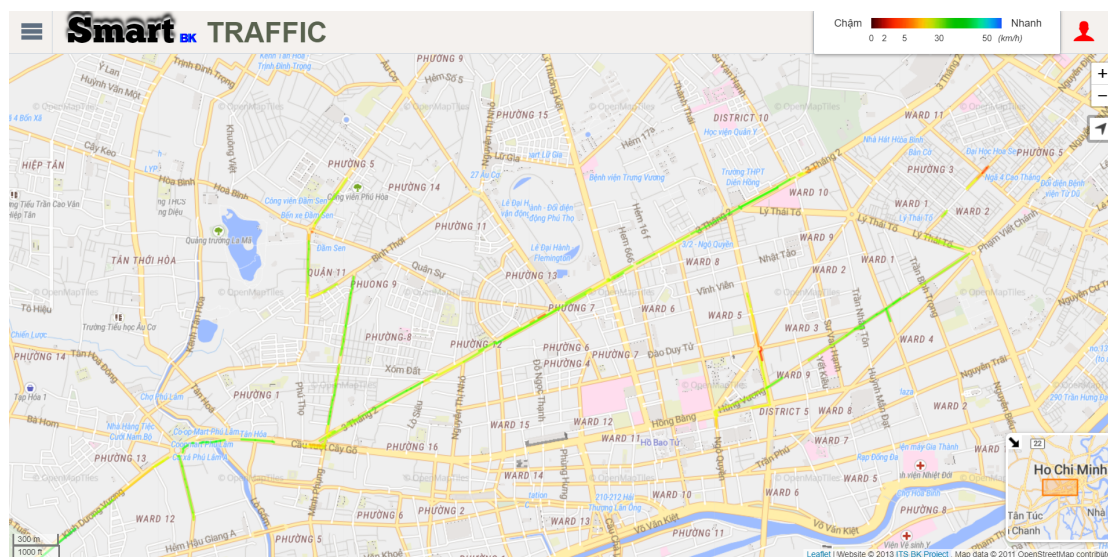
2.1.3 So sánh với Google Maps

So với Google Maps, OSM cho phép người dùng có thể duy trì và chỉnh sửa dữ liệu bản đồ trên toàn thế giới. Vì ai cũng có thể chỉnh sửa nên người ta gọi là "Wikipedia của bản đồ". Còn Google Maps tuy miễn phí với đại đa số người dùng nhưng nếu người sử dụng và các nhà phát triển sử dụng bộ API của họ để đưa bản đồ lên sản phẩm của mình thì sẽ bị tính phí. Cụ thể, nếu bản đồ API mà

bạn sử dụng vượt quá 25.000 lượt xem trong một ngày thì sẽ bị tính phí (với phí là 0.05\$/1000 lượt xem, và không quá 100.000 lượt xem trong vòng 24 giờ).

2.2 Hệ thống giao thông thông minh Trường Đại học Bách Khoa (ITS)

Hệ thống giao thông thông minh Trường Đại học Bách Khoa (Intelligent Transportation Systems - ITS) là hệ thống được nhóm Intelligent Transportation Systems Group (ITSG) phát triển. Hệ thống được xây dựng để thu thập và xử lý tín hiệu GPS từ xe hơi, xe taxi, xe buýt, thiết bị di động; đồng thời cung cấp dữ liệu đã được xử lý đến các ứng dụng web và điện thoại thông minh theo thời gian thực. Từ đó phát triển các ứng dụng nâng cao như dự đoán, cảnh báo tình trạng giao thông hay tìm đường đi tốt nhất ít thời gian nhất. Hệ thống đã và đang được triển khai trên toàn Việt Nam nhưng tập trung chủ yếu ở miền Nam. Hệ thống cũng góp phần đáp ứng nhu cầu về phát triển giao thông thông minh - thành phố thông minh (smart city) của thành phố Hồ Chí Minh và Việt Nam.



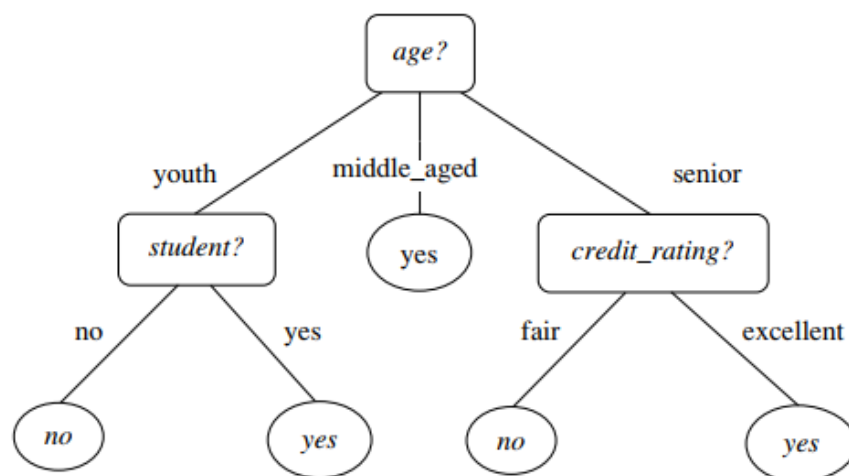
Hình 2: Ảnh chụp giao diện web hệ thống ITS

2.3 Phân loại dữ liệu

2.3.1 Phân loại bằng cây quyết định

Cây quyết định (decision tree) là một cây phân cấp có cấu trúc được dùng để phân lớp các đối tượng dựa vào dãy các luật, đồng thời là một công cụ phổ biến trong khai phá và phân lớp dữ liệu:

- Node nội: phép kiểm thử (test) trên một thuộc tính.
- Node lá: nhãn/mô tả của một lớp (class label)
- Nhánh từ một node nội: kết quả của một phép thử trên thuộc tính tương ứng



9

Hình 3: Cây quyết định chỉ ra khả năng mua máy tính của từng loại khách hàng

2.3.1.a Giải thuật C4.5

C4.5 là thuật toán phân lớp dữ liệu dựa trên cây quyết định hiệu quả và phổ biến trong những ứng dụng khai phá cơ sở dữ liệu có kích thước nhỏ. C4.5 sử dụng cơ chế lưu trữ dữ liệu thường trú trong bộ nhớ, chính đặc điểm này làm C4.5 chỉ thích hợp với những cơ sở dữ liệu nhỏ, và cơ chế sắp xếp lại dữ liệu tại mỗi node trong quá trình phát triển cây quyết định. C4.5 còn chứa một kỹ thuật cho phép biểu diễn lại cây quyết định dưới dạng một danh sách sắp thứ tự các luật if-then (một dạng quy tắc phân lớp dễ hiểu). Kỹ thuật này cho phép làm giảm bớt kích thước tập luật và đơn giản hóa các luật mà độ chính xác so với nhánh tương ứng cây quyết định là tương đương.

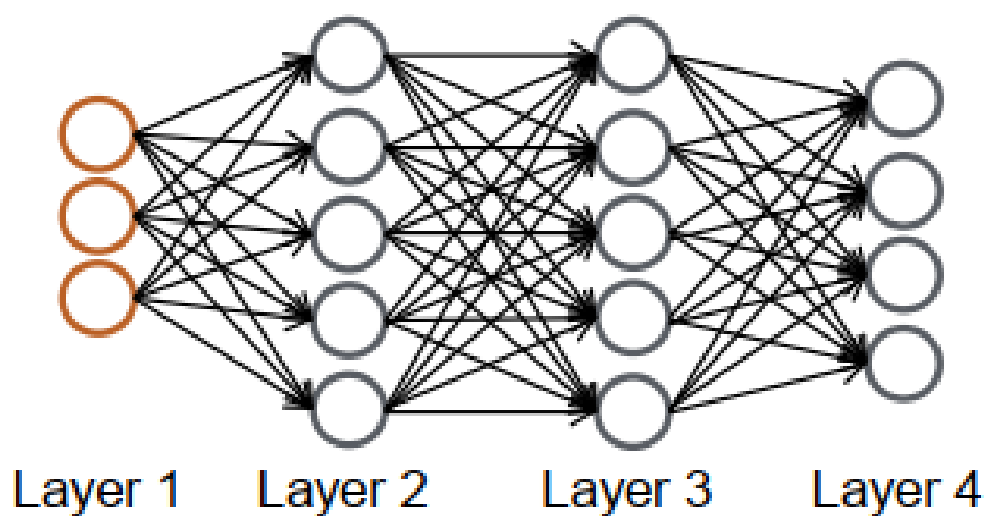
2.3.1.b Cây quyết định J48 trong Weka

Giải thuật C4.5 xây dựng cây quyết định khi hiện thực trên Weka thì được gọi là J48. Công cụ phân lớp này đóng vai trò như bộ lọc và được tổ chức theo một hệ thống cấp bậc. J48 có tên lớp đầy đủ `weka.classifiers.trees.J48`.

2.3.2 Phân loại bằng mạng nơ-ron ANN

Mạng Nơ-ron nhân tạo (Artificial Neural Network- ANN) là mô hình xử lý thông tin được mô phỏng dựa trên hoạt động của hệ thống thần kinh của sinh vật, bao gồm số lượng lớn các Nơ-ron được gắn kết để xử lý thông tin. ANN giống như bộ não con người, được học bởi kinh nghiệm (thông qua huấn luyện), có khả năng lưu giữ những kinh nghiệm hiểu biết (tri thức) và sử dụng những tri thức đó trong việc dự đoán các dữ liệu chưa biết (unseen data).

Kiến trúc chung của một mạng nơ-ron nhân tạo (ANN) gồm 3 thành phần đó là: Input Layer, Hidden Layer và Output Layer. Trong đó, lớp ẩn (Hidden Layer) gồm các Nơ-ron nhận dữ liệu input từ các Nơ-ron ở lớp (Layer) trước đó và chuyển đổi các input này cho các lớp xử lý tiếp theo. Trong một ANN có thể có nhiều lớp ẩn.



Hình 4: Mạng nơ-ron

2.3.3 Cải thiện độ chính xác của bài toán phân lớp có các lớp dữ liệu không cân bằng

Với dữ liệu hai lớp, dữ liệu là loại không cân bằng nếu lớp chính quan tâm (lớp tích cực) được biểu thị bằng một vài bộ dữ liệu, trong khi phần lớn các bộ dữ liệu đại diện cho lớp phủ định, ít quan tâm hay hiển nhiên.

Đối với dữ liệu đa chiều không cân bằng, phân phối dữ liệu của mỗi lớp khác nhau đáng kể ở bộ dữ liệu đại diện cho lớp phủ định, ít quan tâm hay hiển nhiên, lớp chính hoặc các lớp quan tâm thường là hiếm.

Ví dụ: Trong chẩn đoán y tế chẩn đoán sai một bệnh nhân ung thư là khỏe mạnh và ngược lại bệnh nhân khỏe mạnh bị ung thư thường rất ít và hiếm làm do phần trăm không đáng kể nên không ảnh hưởng nhiều độ chính xác so với phần trăm chuẩn đoán đúng nhưng hai lớp này là hai lớp chúng ta quan tâm.

Vấn đề mất cân bằng trong lớp học liên quan chặt chẽ đến tính nhạy cảm của việc học tập với chi phí, trong đó chi phí của các lỗi, mỗi lớp, không bằng nhau. Ví dụ, nó là tổn kém hơn nhiều để chẩn đoán sai một bệnh nhân ung thư là khỏe mạnh (một tiêu cực sai - false negative) hơn để chẩn đoán sai một bệnh nhân khỏe mạnh bị ung thư (một dương tính giả - false positive).

False negative error có thể dẫn đến chết người và do đó đắt hơn nhiều so với false positive error. Các ứng dụng khác liên quan đến dữ liệu không cân bằng trong lớp bao gồm phát hiện gian lận, phát hiện sự cố tràn dầu từ các hình ảnh radar vệ tinh và theo dõi lỗi.

Các độ đo chính xác (accuracy measure) có thể dùng thay thế nhau tùy theo mục đích sử dụng để đánh giá class theo yêu cầu: sensitivity hay recall (the true positive rate); specificity (the true negative rate); F_score; ROC curves (Receiver Operating Characteristic curve - đường cong đặc trưng hoạt động của bộ thu nhận).

2.3.3.a Oversampling và undersampling

Oversampling hoạt động bằng cách lấy lại các tuple tích cực sao cho tập huấn luyện kết quả chứa số lượng tuple dương và âm bằng nhau. **Undersampling** hoạt động bằng cách giảm số lượng các tuple âm. Nó loại bỏ ngẫu nhiên các bộ dữ liệu từ lớp đa số (tiêu cực) cho đến khi có số lượng các bộ tích cực và tiêu cực bằng nhau.

Ví dụ: Giả sử tập huấn luyện ban đầu chứa 100 tuples positive và 1000 negative. Trong oversampling, chúng ta sao chép các bộ dữ liệu của lớp hiếm hơn để tạo thành một tập huấn luyện mới chứa 1000 tuple tích cực và 1000 bộ dữ liệu âm.

Undersampling: Trong quá trình lấy mẫu, loại bỏ ngẫu nhiên các bộ tiêu cực để tập huấn luyện mới chứa 100 tuple positive và 100 tuple negative.

Một số biến thể để oversampling và undersampling tồn tại. Chúng có thể khác nhau, cho ví dụ, trong cách các tuple được thêm vào hoặc loại bỏ. Ví dụ: **thuật toán SMOTE (Synthetic Minority Over-sampling Technique)** sử dụng oversampling, thêm vào tuple tổng hợp - "gần" tích cực tuple trong không gian tuple.

Algorithm *SMOTE*(T, N, k)

Input: Number of minority class samples T ; Amount of SMOTE $N\%$;
Number of nearest neighbors k

Output: $(N/100) * T$ synthetic minority class samples

```
1. (* If  $N$  is less than 100%, randomize the minority class samples as  
   only a random percent of them will be SMOTEd. *)  
2. if  $N < 100$   
3.   then Randomize the  $T$  minority class samples  
4.      $T = (N/100) * T$   
5.      $N = 100$   
6.   endif  
7.  $N = (int)(N/100)$  (* The amount of SMOTE is assumed to be in  
   integral multiples of 100. *)  
8.  $k$  = Number of nearest neighbors  
9.  $numattrs$  = Number of attributes  
10.  $Sample[ ][ ]$ : array for original minority class samples  
11.  $newindex$ : keeps a count of number of synthetic samples generated,  
    initialized to 0  
12.  $Synthetic[ ][ ]$ : array for synthetic samples  
    (* Compute  $k$  nearest neighbors for each minority class sample only. *)  
13. for  $i \leftarrow 1$  to  $T$   
14.   Compute  $k$  nearest neighbors for  $i$ , and save the indices in  
    the  $nnarray$   
15.    $Populate(N, i, nnarray)$   
16. endfor  
  
     $Populate(N, i, nnarray)$  (* Function to generate the synthetic sam-  
    ples. *)  
17. while  $N \neq 0$   
18.   Choose a random number between 1 and  $k$ , call it  $nn$ . This  
    step chooses one of the  $k$  nearest neighbors of  $i$ .  
19.   for  $attr \leftarrow 1$  to  $numattrs$   
20.     Compute:  $dif = Sample[nnarray[nn]][attr] - Sample[i][attr]$   
21.     Compute:  $gap =$  random number between 0 and 1  
22.      $Synthetic[newindex][attr] = Sample[i][attr] + gap * dif$   
23.   endfor  
24.    $newindex++$   
25.    $N = N - 1$   
26. endwhile  
27. return (* End of  $Populate$ . *)  
End of Pseudo-Code.
```

Hình 5: Pseudo code của giải thuật SMOTE.

2.3.3.b Di chuyển ngưỡng - Threshold-moving

Tiếp cận với vấn đề mất cân bằng lớp không liên quan đến việc lấy mẫu. Nó áp dụng cho các bộ phân loại, được đưa ra một bộ dữ liệu đầu vào, trả về một giá trị đầu ra liên tục. Đó là, đối với một bộ tuple đầu vào, \mathbf{X} , một trình phân loại (classifier) như vậy trả về như là một mapping: $f(\mathbf{X}) \rightarrow [0, 1]$. Thay vì chỉnh sửa các bộ dữ liệu training, phương thức này trả về một quyết định phân loại dựa trên các giá trị đầu ra.

Trong cách tiếp cận đơn giản nhất, các bộ dữ liệu mà $f(\mathbf{X}) \geq t$, đối với một số ngưỡng, t , được coi là dương, trong khi tất cả các bộ dữ liệu khác được coi là âm. Các cách tiếp cận khác có thể liên quan đến thao tác các đầu ra theo trọng số.

Nói chung, ngưỡng di chuyển di chuyển ngưỡng, t , sao cho các phân lớp hiếm có dễ phân loại hơn (và do đó, ít có khả năng sai số âm sai hơn).

Ví dụ: những bộ phân loại (classifiers) bao gồm naive Bayesian classifiers và neural network classifiers(backpropagation,...)

Phương pháp di chuyển ngưỡng, mặc dù không phổ biến như quá mức và rút gọn, rất đơn giản và đã cho thấy một số thành công cho dữ liệu không cân bằng hai lớp.

2.3.3.c Ensemble methods

Các bộ phân loại (clasifiers) cá nhân tạo nên bộ quần thể có thể bao gồm các phiên bản của các phương pháp được mô tả ở đây như oversampling và ngưỡng di chuyển.

Algorithm: Bagging. The bagging algorithm—create an ensemble of classification models for a learning scheme where each model gives an equally weighted prediction.

Input:

- D , a set of d training tuples;
- k , the number of models in the ensemble;
- a classification learning scheme (decision tree algorithm, naïve Bayesian, etc.).

Output: The ensemble—a composite model, M^* .

Method:

- (1) **for** $i = 1$ to k **do** // create k models:
- (2) create bootstrap sample, D_i , by sampling D with replacement;
- (3) use D_i and the learning scheme to derive a model, M_i ;
- (4) **endfor**

To use the ensemble to classify a tuple, X :

let each of the k models classify X and return the majority vote;

Hình 6: Pseudo code của giải thuật bagging.

3 Phân tích đề tài

3.1 Tình trạng giao thông tại Việt Nam

Tình trạng kẹt xe đang là vấn đề nghiêm trọng và nhức nhối tại TP.HCM – thành phố đông dân nhất cả nước. Kẹt xe liên tiếp xảy ra ngay trên những đường vốn vẫn lưu thông dễ dàng, và cả vào những giờ không phải cao điểm. Hầu hết các tuyến đường trong thành phố đều đông nghẹt ô tô, xe máy... thêm vào đó là những công trường xây dựng các cao ốc văn phòng, công trình đào đường lấp cống... làm “lô cốt” mọc lên liên tục... càng làm cho việc ùn tắc giao thông dễ xảy ra. Và tình trạng này cứ kéo dài với mức độ ngày càng trầm trọng. Kẹt xe thực tế là một vấn đề “đau đầu” không chỉ riêng ở TP.HCM mà còn là của cả nước, và là vấn đề không thể giải quyết một sớm một chiều.

Các yếu tố ảnh hưởng đến vấn nạn kẹt xe tại TP.HCM:

Mật độ dân cư: mật độ dân số TP.HCM rất cao. Với quy mô dân số như vậy, hệ thống hạ tầng giao thông TP.HCM và nhất là hạ tầng giao thông ở các quận nội ô hiện đã không đáp ứng được yêu cầu giao thông của xã hội là điều không thể tránh khỏi. Mật độ phương tiện giao thông: Mật độ động phương tiện giao thông cao trong khi hệ thống cầu đường đang giảm chất lượng. Sự mất cân đối trên đã dẫn đến hệ quả là kẹt xe diễn ra ngày càng nghiêm trọng.

Hạ tầng giao thông TP.HCM đã trở nên xuống cấp nghiêm trọng, quá tải, thường xuyên ùn tắc. Hệ thống giao thông công cộng kém hiệu quả. Hệ thống đường xá chật hẹp, hư hỏng. Tuy nhiên, việc tổ chức, quản lý kết cấu hạ tầng đường bộ vẫn chưa được quan tâm đúng mức nên đường hư hỏng, xuống cấp nhanh. Trong khi đó, tình trạng con đường vừa xây dựng xong lại phải đào lên để đặt đường ống cấp thoát nước, điện... làm giảm chất lượng cũng như gây nhiều phiền phức cho người dân. Hệ thống cống thoát nước của thành phố xuống cấp, không đáp ứng được yêu cầu thoát nước.

Hệ thống quản lý, thiết bị hỗ trợ: thời gian gần đây, có thể nói chưa bao giờ TP.HCM lại mạnh tay đầu tư cho các hệ thống quản lý và thiết bị hỗ trợ giao thông như vậy. Hàng loạt những dự án nâng cấp, đổi mới, lắp đặt hệ thống đèn tín hiệu, hệ thống quản lý thông tin giao thông mới đang được thành phố quan tâm rất nhiều điển hình như dự án đầu tư bổ sung, lắp đặt các đèn báo lùi thời gian đi kèm với hệ thống đèn tín hiệu giao thông tại các giao lộ trên địa bàn thành phố, dự án tăng cường năng lực quản lý giao thông đô thị cho TP.HCM theo nghị định thư Pháp - Việt, đầu tư áp dụng hệ thống quản lý giao thông ITS... Các hệ thống này được kỳ vọng sẽ góp phần đáng kể giải quyết tình trạng kẹt xe tại TP.HCM. Tuy nhiên, mặc dù được nghiên cứu, đầu tư lớn nhưng hiện nay hệ thống thiết bị hỗ trợ và quản lý giao thông TP.HCM vẫn chưa phát huy được tính năng, vẫn chưa góp phần giải quyết được kẹt xe ngày càng nghiêm trọng.

Ý thức của người tham gia giao thông: vấn đề ý thức của người dân đang trở thành một thực trạng đáng báo động. Hiện nay, tại các thành phố, thị trấn, tình trạng lách luật, vượt đèn đỏ, đi sai phần đường vẫn diễn ra khá phổ biến. Đường chật, phương tiện giao thông quá đông là những lý do được đưa ra, nhưng ý thức của người tham gia giao thông cũng là một yếu tố ảnh hưởng rất đáng kể. Ý thức chấp hành Luật Giao thông đường bộ của người tham gia giao thông hiện nay rất kém, đây chính là nguyên nhân hàng đầu khiến tình hình tai nạn giao thông và kẹt xe tăng đột biến. Một thực tế là tình hình lưu thông tại TP.HCM khá lộn xộn, mạnh ai nấy chạy, bất chấp quy định của pháp luật. Nhất là ở những tuyến đường, giao lộ vắng bóng cảnh sát giao thông, tình trạng người điều khiển phương tiện ngang nhiên vi phạm giao thông ngày càng phổ biến, như một hiện tượng xã hội đáng báo động.

Tổ chức, quản lý, điều hành giao thông: tổ chức, quản lý, điều hành giao thông đóng góp một phần không nhỏ trong giảm bớt tình trạng kẹt xe tại TP.HCM hiện nay. Bên cạnh các yếu tố khác thì việc tổ chức, quản lý, điều hành giao thông bất cập là yếu tố ảnh hưởng không ít đến tình trạng kẹt xe. Thời gian qua, việc thí điểm tổ chức phân làn cho các phương tiện cơ bản đã giải quyết được tình trạng kẹt xe, tê liệt giao thông trên một số tuyến đường thành phố nhưng dường như công tác

tổ chức, điều hành giao thông tại TP.HCM vẫn chưa đạt kết quả như kỳ vọng của nhiều người như việc bố trí, phân làn đường vẫn còn nhiều bất hợp lý, việc xử phạt vi phạm giao thông vẫn còn nhẹ, chưa đạt hiệu quả răn đe người dân, công tác phối hợp giữa các cơ quan hữu quan để tổ chức, nghiên cứu và khảo sát điều tra cơ bản chưa được ngành giao thông công chính làm tới nơi tới chốn.

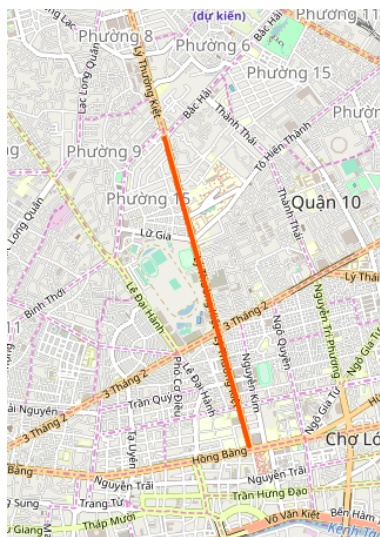
3.2 Giải pháp đề xuất và đề tài bài tập lớn

Phát triển các loại xe công cộng, trong đó chú ý ưu tiên các loại hình vận tải tận dụng không gian trên cao và trong lòng đất (như hệ thống Metro). Nâng cấp, hiện đại hóa hệ thống hạ tầng kỹ thuật, xây dựng gấp các trục đường lớn (xuyên tâm, hướng tâm, tiếp tuyến, vành đai, đường nối các đô thị vệ tinh), tăng cường các cầu vượt sông tại Hà Nội và TP.HCM, triển khai các biện pháp nhằm hạn chế số điểm “giao cắt” giữa các dòng phương tiện, như: lập thể hoá và xây dựng cầu vượt ở các ngã tư lớn (thay vì các vòng xoay), phân luồng hợp lý cho các dòng xe. . . . Nâng cao ý thức người tham gia giao thông cũng là một nhiệm vụ quan trọng giúp hạn chế tình trạng kẹt xe. Ngoài ra, sự phát triển nhanh chóng của công nghệ cũng có thể đóng góp một phần rất quan trọng trong việc giải quyết tình trạng kẹt xe. Bằng cách áp dụng công nghệ vào quản lý giao thông, chúng ta có thể xây dựng các ứng dụng giúp dự đoán tình trạng giao thông và đưa ra các hướng dẫn hợp lý giúp người tham gia có thể chọn đường đi một cách thông minh, sáng suốt. Từ đó, tình trạng kẹt xe có thể được hạn chế và khắc phục.

Bằng cách áp dụng data mining, nhóm đã thực hiện việc dự đoán tình trạng kẹt xe tại một số tuyến đường trong thành phố với dữ liệu lấy từ trang traffic.hcmut.edu.vn. Nhóm đã sử dụng giải thuật để phân loại dữ liệu bằng cây quyết định, từ đó dự đoán tình trạng tắc đường và hiển thị trên bản đồ trực quan.

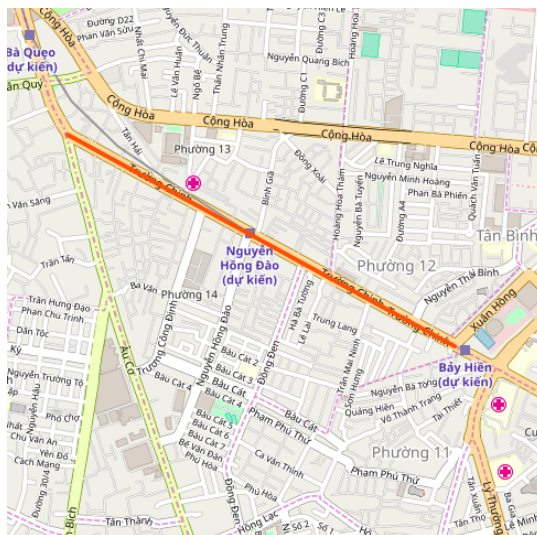
3.3 Phạm vi nghiên cứu

Trong báo cáo này nhóm nghiên cứu dự đoán tình trạng tắc đường tại hai đoạn đường chính. Đoạn đường thứ 1 đó là đường **Lý Thường Kiệt**, đoạn đường từ đoạn cắt từ Bắc Hải đến giao với Hồng Bàng, làn đường có chiều từ Bắc Hải đến Hồng Bàng.



Hình 7: Đoạn đường của đường Lý Thường Kiệt được phân tích

Đoạn đường thứ 2 đó là đường **Trường Chinh**, đoạn đường từ ngã ba với đường Âu Cơ đến ngã tư Bảy Hiền, làn đường có chiều từ Âu Cơ đến ngã tư Bảy Hiền.



Hình 8: Đoạn đường của đường Trường Chinh được phân tích

3.4 Kỳ vọng đạt được

Nhóm hi vọng có thể đưa ra được dự đoán một cách chính xác về tình trạng kẹt xe của TP.HCM. Từ đó có thể áp dụng vào các ứng dụng sử dụng rộng rãi khắp cả nước và đưa ra những hướng dẫn hợp lý giúp hạn chế tình trạng kẹt xe xảy ra và người tham gia giao thông có thể tránh được tình trạng này.

4 Giải quyết vấn đề

4.1 Tổng quan về dữ liệu đầu vào

Dữ liệu được thu thập từ trang traffic.hcmut.edu.vn.

Mô tả các trường trong dữ liệu:

- **segmentId**: Mã định dạng một segment
- **speed**: tốc độ trung bình của một segment tại thời điểm nhất định.
- **density**: mật độ phương tiện lưu thông trên segment. Trường này hiện không có dữ liệu.
- **start_longitude**: kinh độ của điểm bắt đầu một segment.
- **start_latitude**: vĩ độ của điểm bắt đầu một segment.
- **end_longitude**: kinh độ của điểm kết thúc một segment.
- **end_latitude**: vĩ độ của điểm kết thúc một segment.
- **update**: thời gian dưới dạng timestamp, ví dụ 1507263295110 tương ứng với thứ sáu, 6 tháng 10 năm 2017 11:14:55.110 GMT+07:00
- **date**: thời gian dưới dạng chuỗi theo định dạng năm-tháng-ngày.

Dữ liệu được lưu trữ trong từng file theo ngày. Bên dưới là hình ảnh của một file dữ liệu trong một ngày, cụ thể ở đây là ngày 6/4/2018.

	A	B	C	D	E	F	G	H	I	J
1	segmentId	speed	density	frame	start_longitude	start_latitude	end_longitude	end_latitude	update	date
2	32592320135183	13.333333333333333	Thông thoáng	95	106.2481895	21.1197766	106.2484304	21.1197749	1522947562577	2018-04-06
3	26646920364032		38 Thông thoáng	95	105.7687747	21.0148991	105.7689987	21.0146021	1522947562577	2018-04-06
4	4504415043590	32.833333333333333	Thông thoáng	95	105.9536896	9.6319194	105.9629624	9.6201296	1522947562577	2018-04-06
5	4504415043588	56.16666666666667	Thông thoáng	95	105.9493035	9.6374511	105.9535894	9.6320476	1522947562577	2018-04-06
6	4504415043587	59.66666666666667	Thông thoáng	95	105.9470191	9.6403311	105.9493035	9.6374511	1522947562577	2018-04-06
7	4504415043586	55.583333333333333	Thông thoáng	95	105.9426199	9.6459157	105.9470191	9.6403311	1522947562577	2018-04-06
8	4504415043584	41.5	Thông thoáng	95	105.9382945	9.6514036	105.9405059	9.6486572	1522947562577	2018-04-06
9	32592320135190		15 Thông thoáng	95	106.2504843	21.1196231	106.2507364	21.119619	1522947562577	2018-04-06
10	32592320135191	18.333333333333333	Thông thoáng	95	106.2507364	21.119619	106.2509885	21.1196247	1522947562577	2018-04-06
11	32592320135189		17.5 Thông thoáng	95	106.2502173	21.1196321	106.2504843	21.1196231	1522947562577	2018-04-06
12	32592320135184	6.666666666666667	Thông thoáng	95	106.2484304	21.1197749	106.2488293	21.11976	1522947562577	2018-04-06
13	32592320135185		20.5 Thông thoáng	95	106.2488293	21.11976	106.2492815	21.1197238	1522947562577	2018-04-06
14	8297468067840	42.66666666666667	Thông thoáng	95	108.0757584	10.9316696	108.0749149	10.9309436	1522947562577	2018-04-06
15	8297468067841	43.333333333333333	Thông thoáng	95	108.0749149	10.9309436	108.0747414	10.9308239	1522947562577	2018-04-06
16	8297468067842	43.66666666666667	Thông thoáng	95	108.0747414	10.9308239	108.0745662	10.9307122	1522947562577	2018-04-06
17	28588278415365		30 Thông thoáng	95	106.6162768	10.7672673	106.6161353	10.7670699	1522947562577	2018-04-06
18	8297468067844		44 Thông thoáng	95	108.0742194	10.9305239	108.0733248	10.9301273	1522947562577	2018-04-06
19	8297468067845		44 Thông thoáng	95	108.0733248	10.9301273	108.0692567	10.9283969	1522947562577	2018-04-06
20	8297468264448	42.66666666666667	Thông thoáng	95	108.0766457	10.9324436	108.0757584	10.9316696	1522947562577	2018-04-06
21	11507465060359	25.354166666666667	Thông thoáng	95	106.3916682	10.3041607	106.3997382	10.3025065	1522947562577	2018-04-06
22	29922424127489		20 Thông thoáng	95	106.6225387	10.7673066	106.62224	10.7673305	1522947562577	2018-04-06

Hình 9: Hình ảnh dữ liệu của ngày 6/4/2018

Nhóm thu thập dữ liệu của 7 tuần ngẫu nhiên để làm tập dữ liệu ban đầu trong nghiên cứu này.

4.2 Tiền xử lí dữ liệu

4.2.1 Lọc dữ liệu (Data Selection)

Tập dữ liệu mà chúng ta có được thu thập từ nhiều đường trong thành phố Hồ Chí Minh, do đó, đầu tiên chúng ta cần lọc ra những segment tương ứng nằm trên 2 đoạn đường mà chúng ta phân tích. Street ID của 2 đoạn đường đó là **219861105** đối với đoạn đường Trường Chinh và **220860894** đối với đoạn đường Lí Thường Kiệt. Kết quả của việc lọc dữ liệu, biến đổi dữ liệu và gắn nhãn được thực hiện trong một đoạn code sử dụng thư viện Pandas của Python nên kết quả sẽ được trình bày bên dưới sau bước biến đổi dữ liệu và gắn nhãn dữ liệu.

4.2.2 Biến đổi dữ liệu (Data Transformation)

Bước đầu tiên ta làm trong giai đoạn này là đưa khoảng giá trị của trường segmentID về khoảng hợp lí hơn cho việc tính toán cũng như quan sát. Sau khi đã lọc ra được tất cả các segment của mỗi đoạn đường, ta đánh số lại nó dạng 1,2,3,... Trường timestamp của tập dữ liệu sẽ được biến đổi thành 2 thuộc tính mới là hour tương ứng với giờ trong ngày và weekday tương ứng với ngày trong tuần. Trường hour có giá trị trong khoảng 0-24, trường weekday có giá trị nguyên từ 0 đến 6, với 0 tương ứng là thứ 2, 1 là thứ ba, ...

Nhận thấy lưu lượng lưu thông vào những thời điểm khác nhau là không giống nhau. Cụ thể vào giờ cao điểm thì lưu lượng lưu thông sẽ cao hơn nhiều so với những giờ thấp điểm. Để thể hiện điều này thì ta sẽ thêm một trường mới vào dữ liệu đó là isPeakedTime. Theo nghiên cứu thực tế và dựa vào sự phân bố của dữ liệu thì nhóm gán các khoảng thời gian 6h-8h, 11h-14h, 16h-19h30 trong 1 ngày là những khoảng thời gian cao điểm. Trường isPeakedTime sẽ có giá trị 1 nếu là giờ cao điểm, ngược lại là 0.

Bên cạnh đó, lưu lượng lưu thông vào các ngày cuối tuần và ngày bình thường cũng khác nhau, nên ta sẽ tạo ra một thuộc tính mới là isWeekend. Với các ngày cuối tuần là thứ 7 và chủ nhật thì trường này sẽ có giá trị là 1, còn những ngày bình thường thì trường này là 0.

4.2.3 Gắn nhãn dữ liệu

Công việc tiếp theo cần phải làm trước khi khai phá dữ liệu là cần phải gắn nhãn dữ liệu. Một số phương pháp để gắn nhãn có thể đề cập như:

- Watching video real time
- Ngưỡng (Threshold)
- Gom cụm (Clustering)

Trong nghiên cứu này, nhóm sử dụng phương pháp threshold để gắn nhãn cho dữ liệu. Trường speed sẽ được dùng để gắn nhãn cho dữ liệu. Bảng dưới đây là các giá trị ngưỡng và các nhãn được gán tương ứng:

Tốc độ	Nhãn
$0 < v < 5$	Red
$5 \leq v < 15$	Yellow
$v \geq 15$	Green

Nhóm sử dụng thư viện Pandas của Python để thực hiện việc này. Kết quả qua 2 bước này ta có được 2 tập dữ liệu của 2 đoạn đường như sau.



Current relation
Relation: train tc
Instances: 22385
Attributes: 8

Attributes

No.	Name
1	<input type="checkbox"/> segmentId
2	<input type="checkbox"/> speed
3	<input type="checkbox"/> timestamp
4	<input type="checkbox"/> weekday
5	<input type="checkbox"/> hour
6	<input type="checkbox"/> isPeakedTime
7	<input type="checkbox"/> isWeekend
8	<input checked="" type="checkbox"/> congestion

Hình 10: Thông tin về dữ liệu của đoạn đường Trường Chinh khi lọc dữ liệu, biến đổi dữ liệu và gắn nhãn dữ liệu

Relation: train tc								
No.	segmentId	speed	timestamp	weekday	hour	isPeakedTime	isWeekend	congestion
	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Nominal
1	1.0	31.0	1.52286...	3.0	0.23...	0.0	0.0	Green
2	1.0	11.6...	1.52290...	3.0	11.2...	1.0	0.0	Yellow
3	1.0	14.3...	1.52291...	3.0	13.9...	1.0	0.0	Yellow
4	1.0	9.33...	1.52292...	3.0	17.9...	1.0	0.0	Yellow
5	1.0	4.5	1.52295...	4.0	1.23...	0.0	0.0	Red
6	1.0	14.1...	1.52297...	4.0	8.48...	0.0	0.0	Yellow
7	1.0	19.0	1.52298...	4.0	9.73...	0.0	0.0	Green
8	1.0	17.0	1.52299...	4.0	13.2...	1.0	0.0	Green
9	1.0	28.5	1.52300...	4.0	15.2...	0.0	0.0	Green
10	1.0	10.7...	1.52302...	4.0	20.2...	0.0	0.0	Yellow
11	1.0	7.97...	1.52302...	4.0	20.9...	0.0	0.0	Yellow
12	1.0	5.5	1.52302...	4.0	21.2...	0.0	0.0	Yellow
13	1.0	7.83...	1.52302...	4.0	22.4...	0.0	0.0	Yellow
14	1.0	20.0...	1.52303...	4.0	23.2...	0.0	0.0	Green
15	1.0	18.0	1.52306...	5.0	8.98...	0.0	1.0	Green
16	1.0	23.0	1.52307...	5.0	10.4...	0.0	1.0	Green
17	1.0	15.9...	1.52308...	5.0	13.2...	1.0	1.0	Green
18	1.0	38.0	1.52308...	5.0	13.9...	1.0	1.0	Green

Hình 11: Hình ảnh một số giá trị trong bảng dữ liệu của đoạn đường Trường Chinh khi lọc dữ liệu, biến đổi dữ liệu và gắn nhãn dữ liệu



Current relation

Relation: train_ltk

Instances: 18114

Attributes: 8

Attributes

All

None

Invert

Pattern

No.		Name
1	<input type="checkbox"/>	segmentId
2	<input type="checkbox"/>	speed
3	<input type="checkbox"/>	timestamp
4	<input type="checkbox"/>	weekday
5	<input type="checkbox"/>	hour
6	<input type="checkbox"/>	isPeakedTime
7	<input type="checkbox"/>	isWeekend
8	<input type="checkbox"/>	congestion

Hình 12: Thông tin về dữ liệu của đoạn đường Lý Thường Kiệt khi lọc dữ liệu, biến đổi dữ liệu và gán nhãn dữ liệu

Relation: train_ltk								
No.	segmentId Numeric	speed Numeric	timestamp Numeric	weekday Numeric	hour Numeric	isPeakedTime Numeric	isWeekend Numeric	congestion Nominal
1	1.0	27.0	1.52288...	3.0	5.48...	0.0	0.0	Green
2	1.0	26.0	1.52288...	3.0	7.98...	1.0	0.0	Green
3	1.0	20.6...	1.52289...	3.0	8.73...	0.0	0.0	Green
4	1.0	21.75	1.52289...	3.0	8.98...	0.0	0.0	Green
5	1.0	24.0	1.52289...	3.0	9.23...	0.0	0.0	Green
6	1.0	29.0	1.52289...	3.0	9.48...	0.0	0.0	Green
7	1.0	26.5	1.52289...	3.0	9.98...	0.0	0.0	Green
8	1.0	24.0	1.52289...	3.0	10.2...	0.0	0.0	Green
9	1.0	26.0	1.52290...	3.0	12.2...	1.0	0.0	Green
10	1.0	23.3...	1.52291...	3.0	13.9...	1.0	0.0	Green
11	1.0	31.6...	1.52291...	3.0	14.4...	0.0	0.0	Green
12	1.0	25.0	1.52291...	3.0	14.9...	0.0	0.0	Green
13	1.0	27.0	1.52291...	3.0	15.2...	0.0	0.0	Green
14	1.0	5.0	1.52293...	3.0	20.4...	0.0	0.0	Yellow
15	1.0	28.8...	1.52293...	3.0	21.4...	0.0	0.0	Green
16	1.0	27.0	1.52296...	4.0	5.73...	0.0	0.0	Green
17	1.0	26.0	1.52297...	4.0	8.23...	0.0	0.0	Green
18	1.0	31.0	1.52297...	4.0	8.73...	0.0	0.0	Green
19	1.0	25.5	1.52298...	4.0	9.48...	0.0	0.0	Green
20	1.0	24.25	1.52298...	4.0	9.73...	0.0	0.0	Green

Hình 13: Hình ảnh một số giá trị trong bảng dữ liệu của đoạn đường Lý Thường Kiệt khi lọc dữ liệu, biến đổi dữ liệu và gán nhãn dữ liệu

12/11/2018 10:21:33 pm

4.2.4 Làm sạch dữ liệu (Data Cleaning)

4.2.4.a Loại bỏ outliers, extreme

Xoay quanh vấn đề này, nổi bật hơn hết là nhận diện phần tử biên (outliers và extreme). Nhắc lại một ít về cách xác định phần tử biên:

Với $IQR = Q3 - Q1$, ta có

- Outliers: $\geq Q3 + 1.5 \times IQR$ hoặc $\leq Q1 - 1.5 \times IQR$
- Extreme: $\geq Q3 + 3 \times IQR$ hoặc $\leq Q1 - 3 \times IQR$

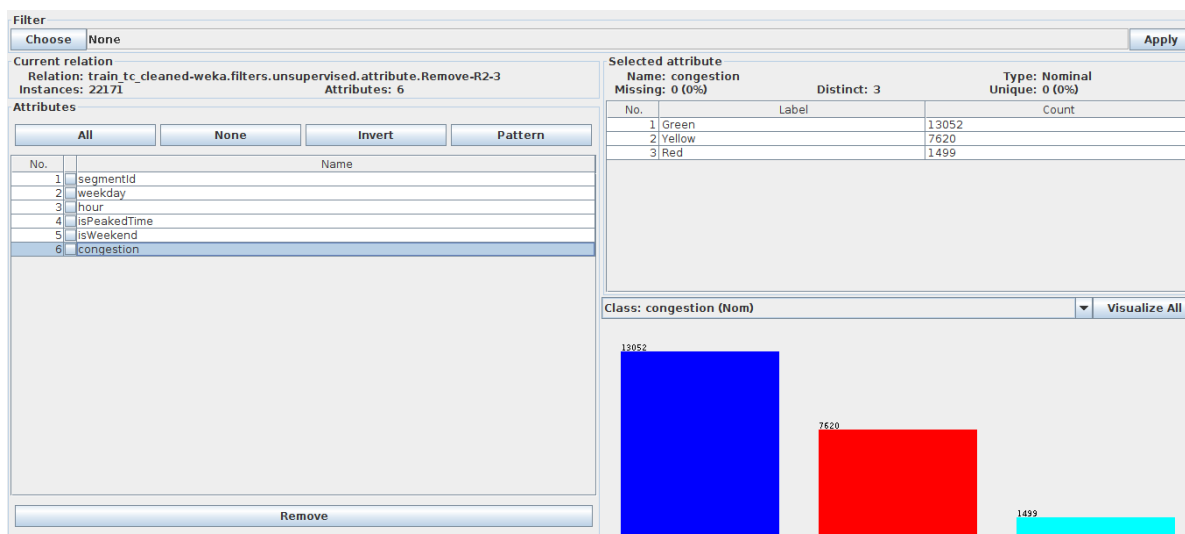
Trong tập dữ liệu mà chúng ta đang thao tác, trường speed là trường được tính toán từ các tập dữ liệu khác được thu thập từ hệ thống ITS nên vấn đề dữ liệu nhiễu có thể xảy ra nên để đảm bảo có một tập dataset tốt, ta cần phát hiện và loại bỏ các phần tử này.

4.2.4.b Vấn đề: Dữ liệu ở trường speed bằng 0

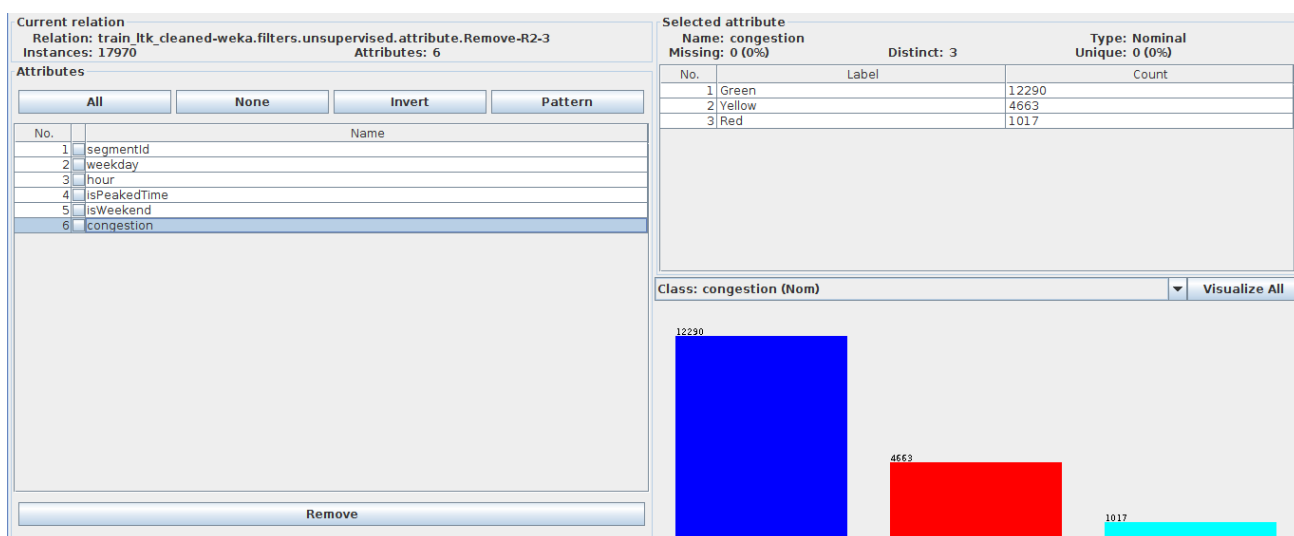
Theo mô tả ở trên về trường speed thì speed ở đây là tốc độ lưu thông trung bình của tất cả các phương tiện trong đoạn segment đang xét. Segment ở đây là một đoạn nhỏ trên đoạn đường đang dự đoán tình trạng giao thông. Nếu speed lớn hơn 0 thì không có vấn đề gì. Theo thực tế ta có thể nhận ra rằng khi trường speed nhỏ thì khả năng segment đang xét hiện tại đang tắc đường là rất cao. Tuy nhiên khi speed bằng 0 thì ta cần xem xét một số vấn đề sau:

- **Không có phương tiện nào đang lưu thông**
Khi không có phương tiện nào đang lưu thông thì trường speed của đoạn segment đó sẽ bằng 0. Điều này dẫn đến dữ liệu nhiễu, ở đây tốc độ bằng 0 nhưng thực chất không có tắc đường xảy ra.
- **Segment này nằm ở ngã ba, ngã tư,... có đèn giao thông và tại thời điểm đang xét tín hiệu đèn giao thông là đèn đỏ**
Một trường hợp khác trường speed của segment có thể bằng 0 mà không phải tắc đường là đoạn segment đó nằm tại ngã ba, ngã tư có đèn giao thông và lúc đang xét đang đèn đỏ. Giả sử mọi người đều chấp hành tốt luật an toàn giao thông thì lúc này tốc độ của các phương tiện giao thông tại segment này sẽ bằng 0. Có thể thấy được đây cũng là một trường hợp cần xem xét bởi tốc độ bằng 0 nhưng không phải tắc đường.

Hiện tại với những tuple có trường speed bằng 0, chúng ta sẽ loại bỏ khỏi dataset. Kết quả sau khi làm sạch dữ liệu ta có 2 tập dataset tương ứng với 2 đường như sau:



Hình 14: Dataset của đoạn đường Trường Chinh sau khi làm sạch dữ liệu



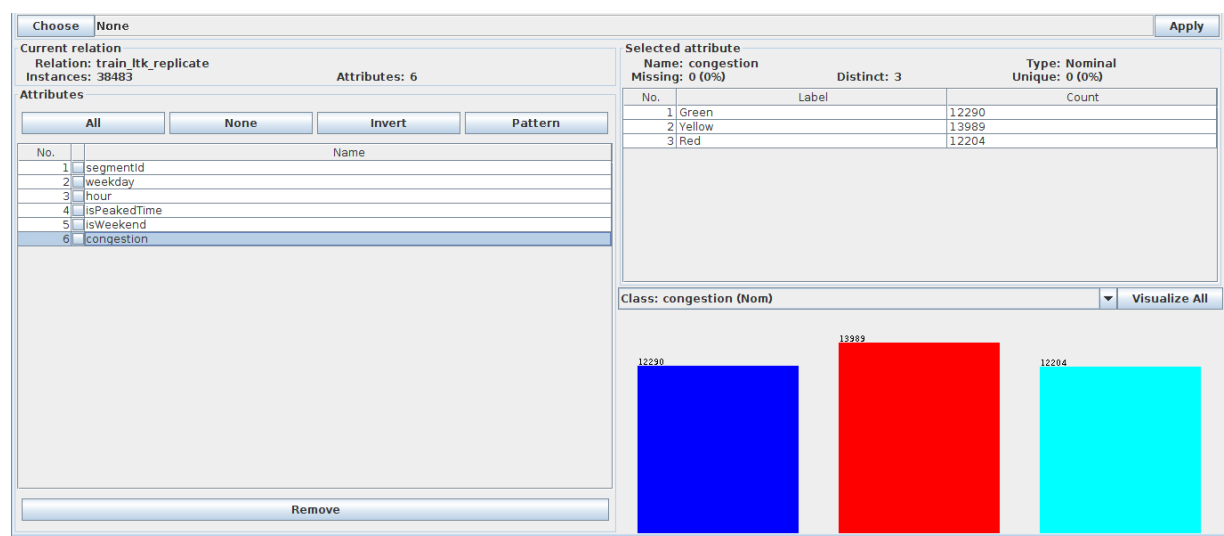
Hình 15: Dataset của đoạn đường Lý Thường Kiệt sau khi làm sạch dữ liệu

4.2.5 Cân bằng các lớp trong dataset

Nhìn vào 2 dataset ở trên ta nhận thấy lớp mà ta đang quan tâm là lớp Red có rất ít dữ liệu. Cụ thể như sau:

Tập dữ liệu	Class Red	Class Yellow	Class Green
Lý Thường Kiệt	1017	4663	12290
Trường Chinh	1499	7620	13052

Với tập dữ liệu mất cân bằng như vậy thì kết quả phân loại sẽ không tốt. Do đó nhóm sẽ dùng một số kỹ thuật cân bằng lại dataset đã nêu ở mục kiến thức cần thiết. Sau khi cân bằng dataset ta có được tập dữ liệu cuối cùng như sau với 3 class có số lượng tuple xấp xỉ nhau như sau:



Hình 16: Dataset của đoạn đường Lý Thường Kiệt sau khi cân bằng các class bằng oversampling

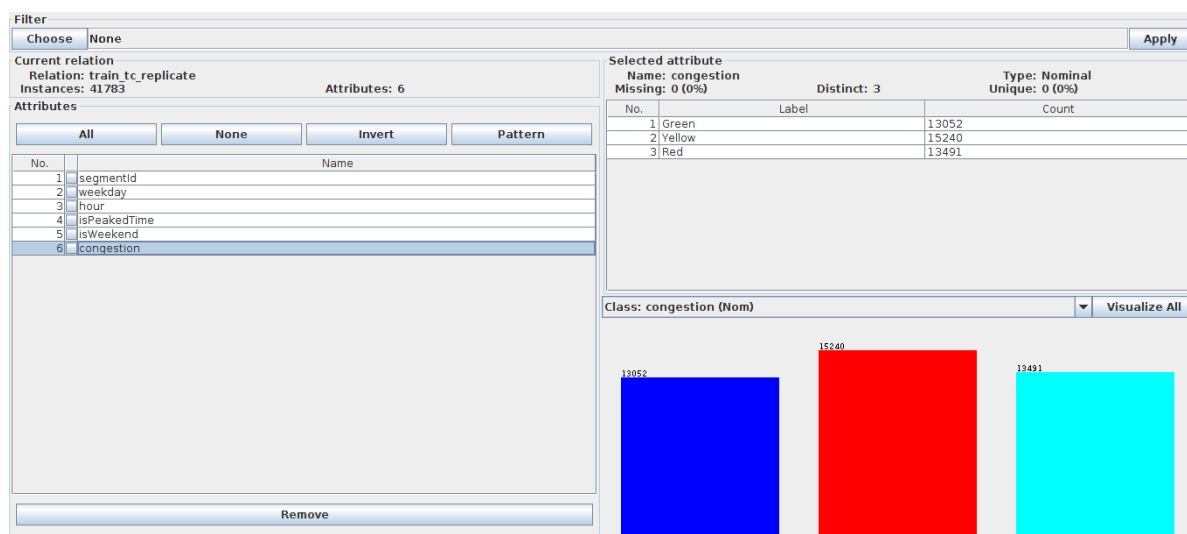
Tập dữ liệu	Class Red	Class Yellow	Class Green
Lý Thường Kiệt	12204	13989	12290
Trường Chinh	13491	15240	13052

4.3 Khai phá dữ liệu

4.3.1 Phân loại bằng cây quyết định

4.3.1.a Model

Do dữ liệu gồm nhiều thuộc tính có dạng category (segmentId, speed, isPeakedTime, isWeekend) nên việc lựa chọn mô hình cây quyết định để phân loại dữ liệu là phù hợp hơn cả. Mô hình dự đoán tắc nghẽn của nhóm sẽ dựa vào dữ liệu cung cấp từ người dùng segmentId (đoạn đường muốn dự đoán), weekday (thời gian trong tuần từ thứ 2 đến chủ nhật), hour (thời điểm trong ngày) và dựa vào dữ liệu học từ quá khứ để cho ra kết quả dự đoán. Như đã trình bày ở trên, việc dự đoán tình trạng tắc nghẽn giao thông còn phụ thuộc vào nhiều yếu tố khác như khi vào giờ cao điểm thì hiện tượng này sẽ xảy ra thường xuyên hơn hay những ngày cuối tuần lượng giao thông sẽ tăng lên. Để tăng độ chính xác cho thuật toán và phản ánh tình trạng giao thông của Việt Nam, nhóm đã tạo thêm 2 thuộc tính

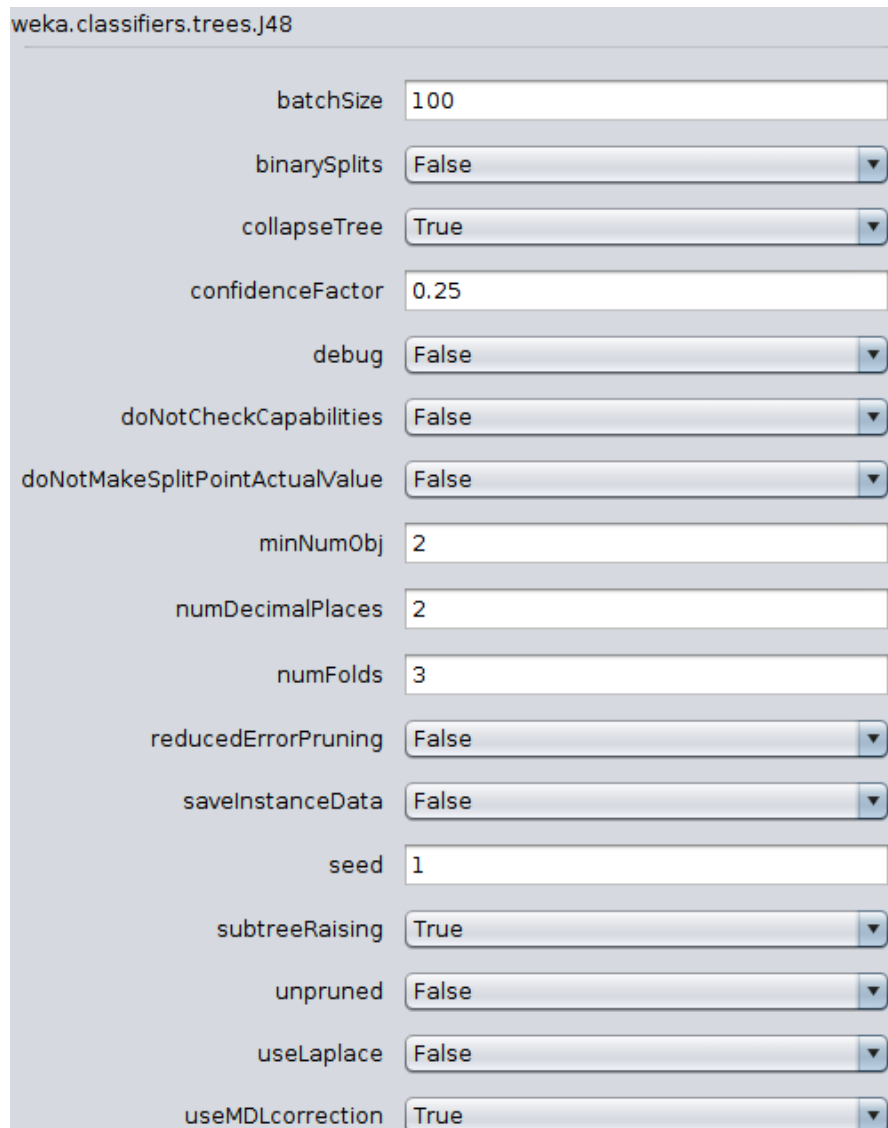


Hình 17: Dataset của đoạn đường Trường Chinh sau khi cân bằng các class bằng oversampling

bổ sung là: isPeakedTime(có là giờ cao điểm) và isWeekend(có là ngày cuối tuần). Vì vậy, model của nhóm đề xuất là: (segmentId, weekday, hour) → (segmentId, weekday, hour, isPeakedTime, isWeekend) → Decision Tree. Do hiện tượng tắc nghẽn giao thông chỉ xảy ra một số thời điểm trong ngày và thời gian xảy ra quá dài nên số lượng dữ liệu được gắn nhãn congestion là rất nhỏ so với lượng dữ liệu được gắn nhãn congestion. Các thuật toán machine learning phân loại chỉ áp dụng tốt với các loại dữ liệu mà ở đó, các lớp có lượng dữ liệu tương đương nhau. Để giải quyết vấn đề này, có 2 cách giải quyết:

- Oversample: Đó là lặp lại dữ liệu của lớp có số lượng dữ liệu ít hơn để cân bằng với lớp có dữ liệu chiếm đa số.
- Undersample: Ngẫu nhiên xóa một lượng dữ liệu của lớp lớn hơn để tạo ra sự cân bằng giữa các lớp dữ liệu. Tuy nhiên, điều này có thể làm mất đi dữ liệu quan trọng.

Vì vậy, nhóm chọn cách oversample để cân bằng dữ liệu. Training dữ liệu, nhóm dùng thuật toán decision tree C4.5 được hiện thực trong Weka là J48 với các hyperparameters như sau:



The image shows the Weka GUI for the J48 classifier. The title bar reads 'weka.classifiers.trees.J48'. The settings are as follows:

Parameter	Value
batchSize	100
binarySplits	False
collapseTree	True
confidenceFactor	0.25
debug	False
doNotCheckCapabilities	False
doNotMakeSplitPointActualValue	False
minNumObj	2
numDecimalPlaces	2
numFolds	3
reducedErrorPruning	False
saveInstanceData	False
seed	1
subtreeRaising	True
unpruned	False
useLaplace	False
useMDLcorrection	True

Hình 18: Hyperparameters dùng để training model

4.3.1.b Test

Dùng phương pháp **cross validation** với **fold = 10**, ta được kết quả như sau:
Đối với dataset trên đoạn đường Lý Thường Kiệt, kết quả training và test như sau:



Time taken to build model: 5.13 seconds

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	34084	88.569 %
Incorrectly Classified Instances	4399	11.431 %
Kappa statistic	0.8279	
Mean absolute error	0.0988	
Root mean squared error	0.2535	
Relative absolute error	22.2793 %	
Root relative squared error	53.8362 %	
Total Number of Instances	38483	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.759	0.040	0.899	0.759	0.823	0.756	0.908	0.886	Green
	0.897	0.101	0.836	0.897	0.865	0.785	0.945	0.856	Yellow
	1.000	0.034	0.932	1.000	0.965	0.949	0.990	0.964	Red
Weighted Avg.	0.886	0.060	0.887	0.886	0.884	0.828	0.948	0.900	

=== Confusion Matrix ===

a	b	c	<-- classified as
9332	2466	492	a = Green
1047	12548	394	b = Yellow
0	0	12204	c = Red

Hình 19: Kết quả training đường Lý Thường Kiệt

Đối với đường Trường Chinh:

Time taken to build model: 6.29 seconds

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	33364	79.8507 %
Incorrectly Classified Instances	8419	20.1493 %
Kappa statistic	0.6969	
Mean absolute error	0.1568	
Root mean squared error	0.3213	
Relative absolute error	35.3709 %	
Root relative squared error	68.2262 %	
Total Number of Instances	41783	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.655	0.101	0.747	0.655	0.698	0.576	0.856	0.774	Green
	0.743	0.144	0.748	0.743	0.745	0.600	0.886	0.760	Yellow
	1.000	0.060	0.888	1.000	0.941	0.913	0.982	0.938	Red
Weighted Avg.	0.799	0.103	0.793	0.799	0.794	0.694	0.907	0.822	

=== Confusion Matrix ===

a	b	c	<-- classified as
8555	3809	688	a = Green
2904	11318	1018	b = Yellow
0	0	13491	c = Red

Hình 20: Kết quả training đường Trường Chinh

4.3.2 Phân loại bằng một số phương pháp khác

Bên cạnh dùng cây quyết định J48 ở trên, nhóm có dùng một số phương pháp phân loại khác, BayesNet được kết quả như sau:

- **ANN** Mô hình mạng neuron network trong Weka có tên là MultilayerPerception, cũng được nhóm sử dụng với các thông số mặc định của Weka, cho kết quả như sau:

Time taken to build model: 47.38 seconds

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	19374	50.3443 %
Incorrectly Classified Instances	19109	49.6557 %
Kappa statistic	0.2511	
Mean absolute error	0.3763	
Root mean squared error	0.4367	
Relative absolute error	84.8336 %	
Root relative squared error	92.7249 %	
Total Number of Instances	38483	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.628	0.149	0.664	0.628	0.645	0.486	0.804	0.671	Green
	0.482	0.374	0.424	0.482	0.451	0.106	0.603	0.431	Yellow
	0.403	0.231	0.448	0.403	0.424	0.178	0.690	0.446	Red
Weighted Avg.	0.503	0.256	0.508	0.503	0.505	0.250	0.695	0.513	

=== Confusion Matrix ===

a	b	c	<-- classified as
7714	3212	1364	a = Green
2553	6739	4697	b = Yellow
1345	5938	4921	c = Red

Hình 21: Kết quả training đường Lý Thường Kiệt bằng Neural Network

Time taken to build model: 50.09 seconds

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	18788	44.9657 %
Incorrectly Classified Instances	22995	55.0343 %
Kappa statistic	0.1647	
Mean absolute error	0.4121	
Root mean squared error	0.4566	
Relative absolute error	92.9323 %	
Root relative squared error	96.9743 %	
Total Number of Instances	41783	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.477	0.177	0.551	0.477	0.511	0.313	0.698	0.535	Green
	0.561	0.472	0.405	0.561	0.471	0.085	0.561	0.402	Yellow
	0.297	0.190	0.427	0.297	0.351	0.120	0.634	0.423	Red
Weighted Avg.	0.450	0.289	0.458	0.450	0.445	0.168	0.627	0.450	

=== Confusion Matrix ===

a	b	c	<-- classified as
6230	5085	1737	a = Green
3048	8545	3647	b = Yellow
2035	7443	4013	c = Red

Hình 22: Kết quả training đường Trường Chinh bằng Neural Network

- **BayesNet** Mạng Bayes phân loại sử dụng phương pháp xác suất cũng được nhóm sử dụng, thuật toán tương ứng trong Weka là BayesNet, cho kết quả như sau(thông số mặc định của Weka):



Time taken to build model: 0.78 seconds

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	22630	58.8052 %
Incorrectly Classified Instances	15853	41.1948 %
Kappa statistic	0.3817	
Mean absolute error	0.344	
Root mean squared error	0.4141	
Relative absolute error	77.5671 %	
Root relative squared error	87.9239 %	
Total Number of Instances	38483	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.731	0.147	0.699	0.731	0.715	0.577	0.858	0.761	Green
	0.491	0.255	0.523	0.491	0.507	0.239	0.696	0.546	Yellow
	0.556	0.218	0.542	0.556	0.549	0.335	0.766	0.566	Red
Weighted Avg.	0.588	0.209	0.585	0.588	0.586	0.377	0.770	0.621	

=== Confusion Matrix ===

a	b	c	<-- classified as
8978	2120	1192	a = Green
2578	6870	4541	b = Yellow
1284	4138	6782	c = Red

Hình 23: Kết quả training đường Lý Thường Kiệt bằng BayesNet

Time taken to build model: 0.24 seconds

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	21475	51.3965 %
Incorrectly Classified Instances	20308	48.6035 %
Kappa statistic	0.2682	
Mean absolute error	0.3838	
Root mean squared error	0.4371	
Relative absolute error	86.5654 %	
Root relative squared error	92.8387 %	
Total Number of Instances	41783	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.591	0.166	0.618	0.591	0.604	0.430	0.794	0.646	Green
	0.474	0.332	0.450	0.474	0.462	0.140	0.633	0.489	Yellow
	0.484	0.237	0.493	0.484	0.489	0.248	0.718	0.519	Red
Weighted Avg.	0.514	0.250	0.516	0.514	0.515	0.266	0.711	0.548	

=== Confusion Matrix ===

a	b	c	<-- classified as
7717	3534	1801	a = Green
3105	7223	4912	b = Yellow
1672	5284	6535	c = Red

Hình 24: Kết quả training đường Trường Chinh bằng BayesNet

4.4 Đánh giá kết quả

Đối với dữ liệu imbalance, độ chính xác Accuracy thường không phản ánh tính đúng đắn của model. Bởi vì chỉ cần dự đoán tất cả trường hợp luôn thuộc về lớp chiếm đa số thì ta luôn có một độ chính xác Accuracy rất cao. Để đánh giá những mô hình thế này, chúng ta thường sử dụng các phương pháp đánh giá khác như Precision/Recall, F-score, Confusion Matrix,... Một trong những đánh giá độ hiệu quả model hay được sử dụng là F-score.

F-score được tính như sau:

$$F_1 = 2 \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

Một model hiệu quả thường có precision/recall cao. F-score cao theo công thức trên đồng nghĩa với việc precision và recall của model cũng cao.

Sau đây là kết quả, F-score của các model trên:

Algorithm	Lý Thường Kiệt	Trường Chinh
Decision Tree	0.884	0.794
Neural Network	0.505	0.445
BayesNet	0.586	0.515

Chúng ta thấy, sử dụng Decision Tree cho kết quả tốt hơn các thuật toán còn lại rất nhiều.

4.5 Hiện thị dữ liệu dự đoán lên bản đồ

5 Tổng kết và định hướng trong tương lai

5.1 Đánh giá kết quả đạt được

5.2 Vấn đề tồn tại và định hướng trong tương lai



Tài liệu