# **I) Data Mining - Classification: the Basic**

# **II) Classification with Imbalanced Dataset**

## Định nghĩa

Tập dữ liệu liệu mất cân bằng (Imbalanced Dataset- ID) là tập dữ liệu mà trong đó một hoặc một vài lớp có lượng mẫu (instances) ít hơn hơn các lớp còn lại. Về lý thuyết, tập dữ liệu có phân phối không đồng đều giữa các lớp đã có thể được xem là mất căn bằng. Nhưng trong thực tế, ta thường tìm hiểu Imbalanced Dataset qua những tập có tỉ lệ chênh lệch rất lớn [[[1]](#footnote-0)]. Cụ thể, tỉ lệ giữa lớp nhỏ (minority classes) và lớp lớn (majority classes) có thể lên đến 1:100, 1:1000 hoặc 1:10000 . Trong các trường hợp đó, những luật phân loại (classfication rules) dự đoán lớp thiểu số thường ít hơn, yếu hơn hoặc có những luật không được tìm thấy [[[2]](#footnote-1)]. Do đó, việc phân loại chính xác cho mẫu thuộc lớp thiểu số rất khó khăn: tỷ lệ mất cân bằng của tập dữ liệu càng lớn thì việc phát hiện mẫu thuộc lớp thiểu số càng khó. Trong nghiên cứu Machine Learning và Data Mining, người ta thường gọi tình huống phải xử lý Imbalanced Dataset là Class Imbalance Problem [[3]](#footnote-2) .

## 2) Class Imbalance trong bài toán thực tế và tác động:

### 2.1) Các bài toán thực tế:

Class Imbalance Problem có thể được xem là một trong những vấn đề lớn nhất của Data Mining và Machine Learning nói chung. Bởi vì vấn đề này xuất hiện phổ biến trong nhiều lĩnh vực thực tế và các kỹ thuật học máy truyền thống thưởng tỏ ra kém hiệu quả. Ta xem xét một số ứng dụng thực tế có xuất hiện vấn đề này:

1. Medical Diagnosis. Cơ sở dữ liệu y tế lưu trữ lượng lớn thông tin về bệnh nhân và bệnh án của họ. Việc khai phá dữ liệu này có thể tìm ra các đặc trưng (features) của một số chứng bệnh và dự đoán chúng. Thông qua hệ thống như computer aided decision (CAD), các thuật toán có thể hỗ trợ chuyên viên y tế trong việc chẩn đoán[[4]](#footnote-3) . Cụ thể như: chẩn đoán bệnh nhân có mắc ung thư hay không. Số mẫu bệnh nhân mắc ung thư nhỏ hơn nhiều so với tổng số mẫu lưu trữ hồ sơ. Đây chính là Class Imbalance Problem. Thực tế, nếu kết luận người bị ung thư là không bị thì mang lại hậu quả lớn.
2. **Phát hiện tràn dầu trên bề mặt đại dương.** [[5]](#footnote-4) Tràn dầu thường do sự cố hoặc xả thải từ các tàu. Hệ thống phát hiện tràn dầu này sử dụng tập dữ liệu đến từ ảnh chụp vệ tinh, từ đó phát hiện các sự cố hoặc hành vi xả thải trái phép. Tuy nhiên, số ảnh có vệt dầu là rất nhỏ so với tổng số ảnh chụp, chính là chênh lệch giữa các lớp trong tập dữ liệu đầu vào.
3. **Phát hiện giao dịch gian lận trong thanh toán**. Ví dụ như gian lận thẻ tín dụng hoặc gian lận thanh toán thuê bao di động. Ở Mỹ, gian lận thanh toán thuê bao làm nền công nghiệp viễn thông tổn thất hàng trăm triệu USD mỗi năm [[6]](#footnote-5). Trong các ghi chép giao dịch, số giao dịch hợp lệ lại chiếm phần lớn, vì vậy việc phân tích và phát hiện giao dịch gian lận gặp khó khăn.
4. Và nhiều bài toán thuộc các lĩnh vực khác, như: Phát hiện xâm nhập hệ thống mạng, quản lý rủi ro, phân loại văn bản,v..v…

### 2.2 Tác động:

Learning = representation + evaluation + search

Pedro Domingos, CACM’12. “A few useful things to know about machine learning”.

Trong đó:

* Representation là Target function f(x;
* Evaluation là Objective/cost/loss function P(f(x;,y)
* Search chính là việc optimize hàm P với tham số. Trong quá trình này, y trở thành tham số và trở thành biến.

Ta xét bài toán phân loại chẩn đoán bệnh ung thư với input là một hồ sơ y tế, output là phân lớp: “dương tính” (có bệnh) và “âm tính” (không có bệnh). Trong bài toán 2-class với 2 lớp được định nghĩa như trên, ta có 4 phân loại đánh giá như sau:

* true positives (TP) - được phân loại đúng vào lớp “dương tính”
* true negatives (TN) - được phân loại đúng vào lớp “âm tính”
* false positives (FP) - được phân loại sai vào lớp “dương tính”
* false negatives (FN) - được phân loại sai vào lớp “âm tính”

Một trong các tiêu chí đánh giá truyền thống là Error Rate: số lượng trường hợp phân loại sai (FN và FP) trên tổng số trường hợp.

Giả sử ung thư có mặt trong 3% trong test dataset, một classifier đánh giá 100% trường hợp là không ung thư (âm tính) thì có Error Rate là 3%. Đây là trường hợp có FN chiếm 3%. Mặc dù con số này có vẻ thấp, từ góc nhìn y tế, việc bỏ qua 3% người mắc ung thư không được chẩn đoán đúng và điều trị là không chấp nhận được. Tương tự, các trường hợp FP, việc chẩn đoán nhầm người không bệnh thành có bệnh gây thiệt hại về tiền, tinh thần và rủi ro phẫu thuật. Tuy nhiên, các classifier tối ưu theo Error Rate hay Accuracy = 1- Error Rate) thường sẽ “ưu ái” lớp lớn: một sample sẽ dễ bị dự đoán là lớp lớn trong khi lớp nhỏ thường là lớp được quan tâm hơn trong các bài toán xuất hiện Class Imbalance[[7]](#footnote-6). Như đã nói ở mục 1, việc phân loại cho lớp nhỏ - ở đây là dương tính với ung thư - là rất khó khăn mặc dù hậu quả của việc không phát hiện là rất lớn.

### 2.3) Kết luận

Tóm lại, ta có thể thấy việc sử dụng hàm/tiêu chí đánh giá truyền thống cũng như kĩ thuật/mô hình truyền thống thường không có hiệu quả trong việc phân loại đúng các mẫu thuộc lớp nhỏ. Mà trong nhiều bài toán thực tế, các lớp nhỏ lại lại là lớp được quan tâm hơn, và cần được phân loại chính xác hơn là lớp lớn[[8]](#footnote-7).

## 3)Phương pháp/ kĩ thuật xử lý:

### 3.1) Thay đổi tiêu chuẩn đánh giá:

Xem xét bài toán 2 lớp với Confusion Matrix:

|  |  |  |
| --- | --- | --- |
|  | Dự đoán: Positive | Dự đoán: Negative |
| Positive class | True Positive (TP) | False Negative (FN) |
| Negative class | False Positive (FP) | True Negative (TN) |

Như đã nói ở trên, Accuracy hay Error rate không còn là phép đo phù hợp đối với bài toán Class Imbalance. Người ta cần dùng phép đo mà trong đó bao hàm phân phối các lớp, trước tiên ta xét các phép đo sau:

* True positive rate: TPrate = là số phần trăm các mẫu Positive được phân loại đúng.
* True negative rate: TNrate = là số phần trăm các mẫu Negative được phân loại đúng.
* False positive rate: FPrate = là số phần trăm các mẫu Positive được phân loại sai.
* False negative rate: FNrate = là số phần trăm các mẫu Negative được phân loại sai.

Với mục tiêu phân loại tốt nhất cho tất cả các lớp, ta không thể dùng 4 phép đo trên một cách tách biệt. Một cách thường được dùng trong bài toán Class Imbalance dung hòa 4 phép đo trên chính là phân tích biểu đồ Receiver Operating Characteristic (ROC) hay thường gọi là ROC Curve. ROC curve là đường thể hiện tương quan giữa hai đại lượng TP rate và FP rate, nói cách khác, là đánh đổi giữa benefits (TP rate) và costs (FP rate).

Đối với các classifier mà đầu ra là xác suất mẫu thuộc một class, người ta cần quyết định Decision Threshold (gọi tắt DT): xác xuất đạt trên mức thì thuộc về class ấy. Việc đưa ra Decision Threshold một cách cảm tính dẫn đến hai hậu quả: 1) Ảnh hưởng đến quá trình đánh giá mô hình nếu dùng phép đo truyền thống như Accuracy; 2) Việc so sánh các mô hình có DT khác nhau là không thể.

Để giải quyết vấn đề trên, ROC Curve là đường đồ thị biểu diễn tập tất cả các DT có thể dùng, được vẽ trong đồ thị có trục X là FP và trục Y là TP (miền giá trị của 2 trục là từ 0 đến 1). Trong đó, điểm (0,0) và (1,1) thể hiện những classifier vô dụng luôn phân loại các mẫu là Negative và Positive, điểm (0,1) thể hiện classifier lý tưởng nhất. Sau khi có ROC curve, người ta đưa ra phép đó AUC - Area Under Curve: diện tích vùng nằm bên dưới ROC curve tạo ra bởi đường này và các cạnh đồ thị.

Chỉ số AUC dành cho classifier tốt thường nằm từ mức 0.5 ( dự đoán hoàn toàn ngẫu nhiên 50/50) và 1.0 (classifier lý tưởng). AUC chính là chỉ số thể hiện năng lực phân biệt (distinguishing) giữa 2 class. Tương tự, có nhiều phương pháp dựa trên graphic được đánh giá là hiệu quả trong đánh giá hiệu quả của classifier[[9]](#footnote-8) như sử dụng precision-recall graphs[[10]](#footnote-9), cost curve[[11]](#footnote-10),v..v..

Một metric khác thường được sử dụng như F-measure:

F-measure =

Trong đó Precision = , Recall =(= TPrate) và là hệ số điều chỉnh tương quan giữa Precision và Recall, với lựa chọn phổ biến là 1: thể hiện độ quan trọng ngang nhau giữa hai giá trị này.

Hoặc Geometric Mean (G-mean):

G-mean =

Với Sensitivity = TPrate và Specificity = TNrate.

### 3.2) Các hướng tiếp cận dữ liệu và tiếp cận thuật toán

Nhiều phương pháp giải quyết vấn đề Class Imbalance đã được đưa ra, đi theo các hướng tiếp cận về dữ liệu, về thuật toán hoặc cả hai.

Ở hướng tiếp cận về dữ liệu (Data-level Approaches), mục tiêu chính là cố gắng cân bằng lại phân phối các lớp (class distribution) bằng cách resampling bộ dữ liệu. Người ta đưa ra nhiều phương pháp resampling sử dụng phối hợp với nhiều hệ thống phân lớp (Classification System) khác nhau.

Ở hướng tiếp cận về thuật toán (Algorithm-level approaches), người ta dựa trên hiểu biết về một thuật toán learning có sẵn, hiểu biết về lĩnh vực thực tế mà thuật toán đó áp dụng, tại sao thuật toán learning đó thất bại với tình huống Imbalance Class để phát triển giải pháp cải tiến. Một hướng tiếp cận khác trong nhóm này là Recognition-based One-class learning: người ta tạo ra model chỉ học các mẫu từ 1 class được quan tâm (cần nhận diện). Mục tiêu của model này là với 1 mẫu mới được đưa ra, nó nhận diện độ tương tự (similarity) của mẫu này với class mục tiêu. Hai thuật toán được đưa vào nghiên cứu theo hướng tiếp cận này là SVM[[12]](#footnote-11) và Neural Network training[[13]](#footnote-12).

### 3.3) Cost-sensitive learning

Giải pháp Cost-sensitive learning phối hợp cả hướng tiếp cận về dữ liệu và thuật toán. Bằng cách áp đặt trọng số chi phí cho việc đánh giá sai 1 mẫu, phương pháp này tạo ra tình huống: việc phân loại sai cho mẫu của lớp nhỏ có chi phí lỗi lớn hơn. Sau đó bằng cách hướng đến giảm chi phí này cũng như tổng chi phí lỗi chung, việc dự đoán mẫu thuộc lớp nhỏ sẽ dần chính xác hơn. Trọng số chi phí có thể được gán theo lớp, hoặc gán khác nhau theo mỗi mẫu; có thể được tích hợp vào một thuật toán learning, hoặc vào một phân đoán của hệ thống phân lớp.

Một kĩ thuật cost-sensitive classification tạo ra ma trận chi phí và kết hợp vào quá trình xây dựng mô hình, cuối cùng hướng đến mô hình có chi phí thấp nhất có thể. Hai hướng đi chính trong phương pháp này:

* **Direct Methods**: Trực tiếp đưa trọng số chi phí vào trong các thuật toán có sẵn và tối ưu hóa các chi phí này. Ví dụ trong các giải thuật di truyền, chi phí lỗi cho việc phân loại sai có thể được đưa vào fitness function.
* **Meta-learning**: Không làm thay đổi thuật đoán gốc mà tích hợp trọng số chi phi vào các cơ chế tiền xử lý của training data (sampling) hoặc sau khi có output (threshold).

### 3.3) Ensemble learning và AdaBoost

**3.3.1) Ensemble learning**

Còn được gọi là Multiple Classifier System, ý tưởng cơ bản của Emsemble learning là xây dựng nhiều classifier từ dữ liệu gốc và tổng hợp dự đoán của chúng khi phân loại mẫu mới chưa biết. Động lực chính của phương pháp này là thông qua tổng hợp để gia tăng năng lực tổng quát hóa (generalization ability) của các classifier: bởi vì lượng dữ liệu sử dụng để train là có hạn (đặc biệt đối với lớp nhỏ), các classifier thành phần có thể mắc lỗi, nhưng các pattern bị phân loại sai đối với mỗi classifier thành phần không nhất thiết giống nhau[[14]](#footnote-13).

Phương pháp ensemble được xem là một giải pháp mạnh mẽ với bài toán Class Imbalance Problem[[15]](#footnote-14) [[16]](#footnote-15) [[17]](#footnote-16). Phương pháp này dựa trên sự tổng hợp nhiều thuật toán learning, giải pháp cost-sensitive learning và các kĩ thuật khác[[18]](#footnote-17):

* **Cost-sensitive Emsemble:**
  + **Cost-sensitive Boosting**: AdaCost16 [[19]](#footnote-18), AdaC115, AdaC215,v..v..
* **Data Preprocessing + Emsemble Learning:**
  + **Boosting-based**: SMOTEboost[[20]](#footnote-19), RUSBoost14,v...v..
  + **Bagging-based**[[21]](#footnote-20): OverBagging, UnderBagging, Random Forest[[22]](#footnote-21) v..v..
  + **Hybrid**: EasyEnsemble13, BalanceCascade13,...

**3.3.2) Adaboost:**

Thuật toán Adaboost gán 1 trọng số cho mỗi mẫu thể hiện độ quan tâm đối với mẫu đó, những mẫu thường bị phân loại sai bởi các classifier trước sẽ được gán trọng số lớn nhất. Từ đó, classifier sau để tối ưu hóa Objective/Loss/Cost function sẽ hướng đến việc phân loại các mẫu đó đúng hơn: trong quá trình train, trọng số của các mẫu đã bị phân loại sai sẽ tăng dần trong khi trọng số của các mẫu đã được phân loại đúng sẽ giảm dần. Sản phẩm cuối cùng là tổng hợp của các classifier thành phần, có thể có trọng số đi kèm với mỗi classifier.

Thuật toán Adaboost (mã giả):

\_\_\_\_

Input: S=(x1,y1), (x2,y2),... (xn,yn) với xiX, yi Y.

Initialize: D1(i) = 1/n

For t=1,...T

1. Train base classifier htY với phân phối Dt.
2. Tính toán trọng số của classifier hiện tại.
3. Cập nhật lại phân phối trọng số các sample Dt+1.

Output:

Final classifier H(x):= sign(tht(x) )

\_\_\_\_

Chú thích:T là số round thực hiện training,là 1 hệ số chuẩn hóa cho các classifier thành phần khi tham gia tông hợp, Di là phân phối trọng số các sample ở round thứ i (với ý nghĩa như đã đề cập ở phần lý thuyết).

# III) Các phương pháp Sampling:

Như đã nói ở phần hướng tiếp cận dữ liệu, các lợi thế của phương pháp Sampling có thể kể đến như: 1) Có thể phối hợp với nhiều kỹ thuật đã nêu ở trên, tuy nhiên lại hoàn toàn có thể độc lập với các thuật toán learning mà chúng phối hợp; 2) Phương pháp này trực tiếp, dễ áp dụng cũng như thường xuyên được tích hợp trong các hệ thống phân lớp. Các khuyết điểm dễ thấy là: 1)Ta không biết chính xác phân phối lớp nào là tối ưu; 2) Tiêu chí chọn mẫu để sampling rất quan trọng nhưng phụ thuộc vào bài toán và người quyết định; 3) Các rủi ro tác động đến thông tin quan trọng trong dữ liệu.

Có thể kể ra 3 hướng đi chính trong phương pháp Sampling:

* **Oversampling**: Nhân số lượng dữ liệu của lớp nhỏ lên nhiều lần để có một bộ dữ liệu cân xứng hơn. Điều cần quan tâm là cách chọn các mẫu để nhân lên (tránh mất đi các features quan trọng), rủi ro của việc tăng nhiễu có sẵn trong lớp nhỏ, hoặc tăng khả năng overfit của mô hình.
* **Undersampling**: Giảm số lượng dữ liệu chênh lệch xuống, bằng cách chọn mẫu của lớp lớn để loại bỏ/giữ lại cho đến khi bộ dữ liệu cân bằng. Phương pháp này thường áp dụng với trường hợp dữ liệu lớn, khuyết điểm là dễ làm mất thông tin quan trọng mà có thể ảnh hưởng đến quá trình learning.
* **Hybrid**: Sử dụng cả 2 phương pháp nêu trên. Ví dụ: ta có 60 mẫu thuộc lớp nhỏ (~3%) và 1918 mẫu thuộc lớp lớn (~97%), ta có thể tạo thêm số mẫu thuộc lớn nhỏ cho đến khi đạt 120 mẫu, đồng thời lấy ngẫu nhiên 120 mẫu từ lớp lớn, sau đó học trên tập đã cân bằng này.

## Random:

Hướng đi cơ bản nhất chính là chọn ngẫu nhiên các mẫu để thực hiện Oversampling và Undersampling. Khuyết điểm của Random Over-Sampling (ROS) là dễ tạo ra quá nhiều mẫu giống nhau, dẫn đến tình trạng overfit cho mô hình. Đối với Random Under-Sampling (RUS), ta có thể vô tình loại bỏ các thông tin quan trọng đối trong dataset.

## SMOTE :

Được đưa ra bởi N.V.Chawla, K.W.Bowyer, L.O.Hall, W. Philip Kegelmeyer vào năm 2002, là viết tắt của Synthetic Minority Over-sampling TEchnique. Hướng tiếp cận này được truyền cảm hứng bởi một kĩ thuật trong lĩnh vực nhận dạng kí tự viết tay (Ha & Bunker, 1997): tạo thêm mẫu ký tự viết tay mới từ các mẫu cũ bằng các phép biến đổi xoay, nghiêng. Đối với SMOTE, ý tưởng là tạo thêm các mẫu tổng hợp mới (synthetic) trên không gian đặc tính (feature space) thay vì không gian dữ liệu (data space). Cách thực hiện là, đối với mỗi mẫu của lớp nhỏ đang được xem xét, tạo ra thêm mẫu tổng hợp trên đoạn thẳng giữa mẫu này và 1 trong số k láng giềng cùng lớp gần nhất.

Cụ thể, với mỗi mẫu được xem xét, ta có các bước sau:

1. Tìm k láng giềng gần nhất (Nearest Neighbor-NN) thuộc cùng lớp (trong feature space)
2. Chọn 1 láng giềng trong đó để lấy vector từ mẫu đang xét đến láng giềng ấy.
3. Tạo ngẫu nhiên một số giữa 0 và 1 và nhân với vector vừa chọn, ta tìm được một điểm trên vector. Tạo một mẫu mới cùng lớp trên điểm ấy.
4. Ở một vài biến thể, ta có thể tạo mẫu mới với tất cả các láng giềng tìm được, hoặc có thể chọn ngẫu nhiên 1 láng giềng như trên.

Nhiều biến thể, cũng như phát triển của phương pháp SMOTE đã được nghiên cứu, ví dụ như phối hợp với kĩ thuật boost để tạo ra SMOTEboost20, định nghĩa và tích hợp trọng số để tăng tính chọn lọc như Safe-Level-SMOTE[[23]](#footnote-22) hay ADASYN (Adaptive Synthetic Sampling)[[24]](#footnote-23)

## Informed Undersampling:

Một phương pháp tiêu biểu là phối hợp sử dụng K-nearest neighbor (KNN) classifier trong quá trình undersampling. Dựa trên tính chất mất cân bằng của dữ liệu, các phương pháp KNN undersampling đã được đưa ra[[25]](#footnote-24): NearMiss-1, NearMiss-2, NearMiss-3. Phương pháp NearMiss-1 giữ lại những mẫu thuộc lớp lớn có khoảng cách trung bình đến k láng giềng gần nhất thuộc lớp nhỏ là bé nhất. Phương pháp NearMiss-2 giữ lại những mẫu thuộc lớp lớn có khoảng cách trung bình đến k láng giềng xa nhất thuộc lớp nhỏ là bé nhất. NearMiss-3, với mỗi mẫu thuộc lớp nhỏ, chọn giữ lại k mẫu thuộc lớp lớn là láng giềng gần nhất với chúng. Các biến thể khác có thể kể đến như Edited Nearest Neighbor (ENN): Chọn 1 lớp chiếm số đông trong k-NN của một mẫu, nếu mẫu đó không thuộc “lớp đa số” này thì loại bỏ mẫu đó; hoặc Condensed Nearest Neighbor: chọn ra 1 subset sao cho với mỗi mẫu, k-NN chúng đều thuộc cùng lớp với mẫu ấy.

## Ensemble-based Method:

Trong phương pháp RUS truyền thống, việc mất thông tin (information loss) là một bất lợi lớn, người ta đưa ra cách kết hợp với Ensemble Learning[[26]](#footnote-25). Hai ví dụ tiêu biểu là phương pháp EasyEnsemble và BalanceCascade.

Hiện thực của EasyEnsemble rất trực tiếp: phát triển một hệ thống learning bằng cách thực hiện RUS nhiều lần trên lớp lớn độc lập với nhau, để tạo ra các subset của lớp lớn (có kích thước cân bằng với lớp nhỏ), mỗi subset này kết hợp với lớp nhỏ tạo thành 1 bộ dữ liệu, sau đó xây dựng hệ thống multiple classifiers mà mỗi classifier học trên 1 bộ dữ liệu kết hợp riêng trên.

Đối với BalanceCascade, ý tưởng chính là tạo ra hệ thống ensemble learning có thể tự chọn loại bỏ các mẫu thuộc lớp lớn qua mỗi round: các mẫu thuộc lớp lớn đã được phân loại đúng sẽ được chọn để loại bỏ khỏi learning set của round tiếp theo. (Ta có thể kết hợp xây dựng mô hình giống EasyEnsemble cho phương pháp này).

Ta đều có thể dùng AdaBoost cho 2 phương pháp trên.

## Các phương pháp khác:

### 5.1) Tomek Link Removal:

Một cặp 2 mẫu được gọi là Tomek Link nếu chúng thuộc khác class và đều là NN của nhau. Phương pháp này loại bỏ tất cả Tomek link từ Dataset (hoặc trong một biến thể khác, chỉ loại bỏ trong tập lớp lớn).

### 5.2) Cluster centroids:

Phương pháp này thay thế các mẫu của một cụm (cluster) bằng trung tâm của cụm ấy (cluster centroid) do một thuật toán K-means tìm ra. Tùy theo mức độ undersampling mà người ta chọn số mẫu để thực hiện việc này.

1. H. He, E.A. Garcia, Learning from imbalanced data, IEEE Transactions on Knowledge and Data Engineering 21 (9) (2009) - 1264. [doi: 10.1109/TKDE.2008.239](http://dx.doi.org/10.1109/TKDE.2008.239) [↑](#footnote-ref-0)
2. Y. Sun, A.K.C. Wong, M.S. Kamel, Classification of imbalanced data: a review, International Journal of Pattern Recognition and Artificial Intelligence 23 (4) (2009) - 687. [doi: 10.1142/S0218001409007326](http://dx.doi.org/10.1142/S0218001409007326)) [↑](#footnote-ref-1)
3. N.V. Chawla, N. Japkowicz, A. Kotcz, Editorial: special issue on learning from imbalanced data sets, SIGKDD Explorations 6 (1) (2004) - 1. [doi: 10.1145/1007730.1007733](http://dx.doi.org/10.1145/1007730.1007733) [↑](#footnote-ref-2)
4. M.A. Mazurowski, P.A.Habas, J.M.Zurada, J.Y.Lo, J.A. Baker, G.D. Tourassi. Training Neural Network Classifiers for Medical Decision Making: The Effects of Imbalanced Datasets on Classification (2007). doi: [10.1016/j.neunet.2007.12.031](https://dx.doi.org/10.1016%2Fj.neunet.2007.12.031) [↑](#footnote-ref-3)
5. M. Kubat, R. Holte and S. Matwin, Machine learning for the detection of oil spills in

   satellite radar images, Mach. Learn. (1998) 195–215. [↑](#footnote-ref-4)
6. D. Walters and W. Wilkinson, Wireless fraud, now and in the future: a view of the

   problem and some solutions, Mobile Phone News (1994), pp. 4–7. [↑](#footnote-ref-5)
7. Weiss GM, Provost F. Technical report. Department of Computer Science, Rutgers University (2001): The effect of class distribution on classifier learning: an empirical study. [↑](#footnote-ref-6)
8. C. Elkan, The foundations of cost–sensitive learning, in: Proceedings of the 17th IEEE International Joint Conference on Artificial Intelligence (IJCAI’01) (2001) [↑](#footnote-ref-7)
9. R.C. Prati, G.E.A.P.A. Batista, M.C. Monard, A survey on graphical methods for classification predictive performance evaluation, IEEE Transactions on Knowledge and Data Engineering 23 (11) (2011) 1601–1618. [↑](#footnote-ref-8)
10. J. Davis, M. Goadrich, The relationship between precisionrecall and ROC curves, in: Proceedings of the 23th International Conference on Machine Learning (ICML’06), ACM (2006). 233–240 [↑](#footnote-ref-9)
11. C. Drummond, R.C. Holte, Cost curves: an improved method for visualizing classifier performance, Machine Learning 65 (1) (2006) 95–130 [↑](#footnote-ref-10)
12. L.M. Manevitz, M. Yousef, One-class svms for document classification, J. Mach. Learn. Res. 2 (2001) [↑](#footnote-ref-11)
13. N. Japkowicz, Supervised versus unsupervised binary-learning by feedforward neural networks, Mach. Learn. 41 (1) (2001). [↑](#footnote-ref-12)
14. J. Kittler, M. Katef, R. Duin, J. Matas, On combining classifiers, IEEE Trans. Pattern Anal. Mach. Intell. 20 (3) (1998) [↑](#footnote-ref-13)
15. X.-Y. Liu, J. Wu, Z.-H. Zhou, Exploratory undersampling for class-imbalance learning, IEEE Transactions on System, Man and Cybernetics B 39 (2) (2009) 539–550. [↑](#footnote-ref-14)
16. C. Seiffert, T.M. Khoshgoftaar, J. Van Hulse, A. Napolitano, RUSBoost: a hybrid approach to alleviating class imbalance, IEEE Transactions on System, Man and Cybernetics A 40 (1) (2010) 185–197. [↑](#footnote-ref-15)
17. Y. Sun, M.S. Kamel, A.K.C. Wong, Y. Wang, Cost-sensitive boosting for classification of imbalanced data, Pattern Recognition 40 (12) (2007) [↑](#footnote-ref-16)
18. V.López , A. Fernández , S. García, V. Palade, F. Herrera, An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. Information Sciences 250 (2013) [↑](#footnote-ref-17)
19. W. Fan, S.J. Stolfo, J. Zhang, P.K. Chan, Adacost: misclassification cost-sensitive boosting, in: Proceedings of the 16th International Conference on Machine Learning (ICML’96), Morgan Kaufmann Publishers Inc., San Francisco, CA, USA. (1999). [↑](#footnote-ref-18)
20. N.V. Chawla, A. Lazarevic, L.O. Hall, K.W. Bowyer, SMOTEBoost: Improving prediction of the minority class in boosting, in: Proceedings of 7th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD’03). (2003) [↑](#footnote-ref-19)
21. L. Breiman, Bagging predictors, Machine Learning 24 (2) (1996) [↑](#footnote-ref-20)
22. L. Breiman, Random forests, Machine Learning 45 (2001) [↑](#footnote-ref-21)
23. Chumphol Bunkhumpornpat, Krung Sinapiromsaran, Chidchanok Lursinsap: Safe-Level-SMOTE: Safe-Level-Synthetic Minority Over-Sampling TEchnique for Handling the Class Imbalanced Problem, Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD) (2009) [↑](#footnote-ref-22)
24. Haibo He, Yang Bai, Edwardo A. Garcia, and Shutao Li: ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning. Link: <https://sci2s.ugr.es/keel/pdf/algorithm/congreso/2008-He-ieee.pdf> [↑](#footnote-ref-23)
25. J. Zhang and I. Mani, “KNN Approach to Unbalanced Data Distributions: A Case Study Involving Information Extraction,” Proc. Int’l Conf. Machine Learning (ICML ’2003), Workshop Learning from Imbalanced Data Sets, 2003. [↑](#footnote-ref-24)
26. X.Y. Liu, J. Wu, and Z.H. Zhou, “Exploratory Under Sampling for Class Imbalance Learning,” Proc. Int’l Conf. Data Mining, (2006) 965- 969. [↑](#footnote-ref-25)