1. Binary relevance (BR)

Phương pháp chuyển đổi đơn giản nhất là phương pháp chuyển đổi nhị phân (BR), tức là với mỗi nhãn khác nhau sẽ xây dựng một bộ phân lớp khác nhau. Phương pháp này xây dựng |L| bộ phân lớp nhị phân: Hl: X→ {l; -l} cho mỗi nhãn l khác nhau trong L. Thuật toán chuyển đổi dữ liệu ban đầu trong tập L nhãn. Nhãn là l nếu các nhãn của ví dụ ban đầu gồm l, nhãn là ⌐l trong trường hợp ngược lại. Theo [12], phương pháp này đã được sử dụng bởi Boutell (2004), Goncalves và Quaresma (2003), Lauser và Hotho (2003), Li và Ogihara (2003). Sau đây là ví dụ biểu diễn dữ liệu theo phương pháp này:

Biểu diễn dữ liệu theo phương pháp nhị phân

Example	Label 0	Label 1 (⌐ label 0)	… (⌐ label 0)	Label 99 (⌐ label 0)
1	X			

Example	Label 0(⌐label 1)	Label 1	… (⌐label 1)	Label 99 (⌐ label 1)
2		X		

2. Multi - label k-Nearest Neighbors (MLkNN)

Thuật toán kNN [14] (k-Nearest Neighbors) là phương pháp học máy được sử dụng rộng rãi, thuật toán tìm hàng xóm gần nhất của một đối tượng thử nghiệm trong không gian đặc trưng.

Bộ phân lớp dựa trên thuật toán K người láng giềng gần nhất là một bộ phân lớp dựa trên bộ nhớ, đơn giản vì nó được xây dựng bằng cách lưu trữ tất cả các đối tượng trong tập huấn luyện. Để phân lớp cho một điểm dữ liệu mới x', trước hết bộ phân lớp sẽ tính khoảng cách từ điểm dữ liệu mới tới các điểm dữ liệu trong tập huấn luyện. Qua đó tìm được

tập N (x', D, k) gồm k điểm dữ liệu mẫu có khoảng cách đến x' gần nhất. Ví dụ nếu các dữ liệu mẫu được biểu diễn bởi không gian vector thì chúng ta có thể sử dụng khoảng cách Euclidean để tính khoảng cách giữa các điểm dữ liệu với nhau. Sau khi xác định được tập N (x', D, k), bộ phân lớp sẽ gán nhãn cho điểm dữ liệu x' bằng lớp chiếm đại đa số trong tập N (x', D, k).

Công thức tính Euclidean để tính khoảng cách giữa các điểm dữ liệu: Giả sử có hai phần tử dữ liệu X1=(x11, x12 … x1n) và X2=(x21, x22 ... x2n), độ đo khoảng cách Euclide được tính bằng công thức:

$$Dist(X_1, X_2) = \sqrt{\sum_{i=1}^{n} (x_{1i} - x_{2i})^2}$$

Mô tả thuật toán:

- Đầu vào: tập dữ liệu học D đã có nhãn và đối tượng kiểm tra z.
- Tiến trình:
  - Tính d (x, x') khoảng cách giữa đối tượng kiểm tra và mọi đối tượng (x, y) ϵ D.
  - Lựa chọn tập Dz gồm k đối tượng ϵ
- Đầu ra: nhãn của đối tượng kiểm tra được xác định là

$$y' = \arg\max_{(xi,yi)} \sum I(v = yi)$$

Trong đó:

- v là một nhãn trong tập nhãn
- I () là một hàm số trả lại giá trị 1 khi v có nhãn yi, 0 nếu trong trường hợp ngược lại.
- X là đối tượng xét, y là nhãn của nó.

Nhược điểm của thuật toán k-NN: Đòi hỏi không gian lưu trữ lớn.

Thuật toán MLkNN [13] là thuật toán k-NN áp dụng cho bài toán gán đa nhãn.

Phát biểu bài toán: cho 1 thể hiện x và tập nhãn kết hợp Y ⊆ y, k láng giềng được nhắc tới trong phương pháp ML-KNN. Cho là $\vec{y}_x$ vector phân loại cho x, với l-th là thành phần $\vec{y}_x(l)$ ($l \in Y$) mang giá trị 1 nếu $l \in Y$ và 0 trong trường hợp ngược lại. Thêm vào đó, cho N (x) tập của k láng giềng của x trong tập dữ liệu huấn luyện. Theo đó, nền tảng trên tập nhãn của những người hàng xóm láng giềng, một vector thành viên được định nghĩa như sau:

$$\vec{C_x}(l) = \sum_{a \in N(x)} \vec{y_a}(l), l \in y$$

Với $\vec{C}_x$ tổng số trong láng giềng x tới lớp thứ l.

Trong mỗi trường hợp kiểm tra t, ML-KNN có k hàng xóm N (t) trong mỗi tập huấn luyện. Kí hiệu H^l1 là trường hợp t có nhãn l, H^l0 là trường hợp t không có nhãn l, E^lj (jÎ{0, 1 … K}) biểu thị cho các trường hợp đó, giữa K láng giềng của t, chính xác j thể hiện có l nhãn. Do đó, nền tảng trên vector C_t, phân loại vector y_t sử dụng theo nguyên tắc:

$$\vec{y_t}(l) = argmax_{b \in \{0,1\}} P(H)_b^l \left| E^l_{\vec{C_t}(l)} \right), l \in y$$

Sử dụng luận Bayesian, có thể viết lại:

$$\vec{y_t}(l) = \arg\max_{b \in \{0,1\}} \frac{P(H_b^l) P\left(E^l_{\vec{C_t}(l)} \middle| H_b^l\right)}{P\left(E^l_{\vec{C_t}(l)} \middle| H_b^l\right)}$$

$$= \arg\max_{b \in \{0,1\}} P(E^l_{\vec{C_t}(l)} | H_b^l)$$

Mã giả thuật toán MLkNN được trình bày như sau:

// Tính toán xác suất trước P (H^l_b)

(1) **for** l ∈ Y **do**

(2) P (H^l_1) = (s + $\sum_{i=1}^{m} y_{xi}(l)$) / (s x 2 + m); P(H^l_0) = 1 – P(H^l_1)

// tính xác suất sau P (E^l_j | H^l_b)

(3) Nhận dạng N (xi), i∈ {1, 2 … m};

(4) **for** l ∈ Y **do**

(5) **for** j ∈ {0, 1 … K} **do**

(6) c[j] = 0; c'[j] = 0

(7) **for** i∈{1, 2 … m} **do**

(8) δ = Cxi(l) = $\sum_{a \in N(xi)} ya(l)$;

(9) if ($\vec{y}_{xi}$ (l) == 1) then c[δ] = c[δ] +1;

(10) Else c'[δ] = c'[δ] +1

(11) **for** j ∈ {0, 1 … K} **do**

(12) $P(E^l_j | H^l_1) = (s + c[j]) / (s \times (K + 1) + \sum_{p=0}^{k} c[p])$

(13) $P(E^l_j | H^l_0) = (s + c'[j]) / (s \times (K + 1) + \sum_{p=0}^{k} c'[p])$

// tính toán $\vec{y}_t$ và $\vec{N}_t$

(14) Nhận dạng N (t)

(15) **for** l ∈ γ **do**

(16) $\vec{C_t}(l) = \sum_{a \in N(t)} \vec{y_a}(l)$;

(17) $\vec{y}_t$ (l) = arg max $_{b \in \{0,1\}}$ P (H^l_b) P ($E^l_{\vec{C}t(l)} | H^l_b$);

(18) $\vec{r_t}(l) = P\left(H^l_b \middle| E^l_{\vec{C_t}(l)}\right) = (P(H^l_1)P\left(E^l_{\vec{C_t}(l)} \middle| H^l_1\right)/PE^l_{\vec{C_t}(l)})$

$= (P(H^l_1)P(E^l_{\vec{C_t}(l)}|H^l_1))/(\sum_{b \in \{0,1\}} P(H^l_b)P(E^l_{\vec{C_t}(l)}|H^l_b))$

Hình 2.1 Mã giả thuật toán ML-kNN

3. Random k-labelsets (RAKEL)

Phương pháp Label Powerset (LP) là một phương pháp chuyển đổi của phân lớp dữ liệu đa nhãn mà có xem xét đến sự phụ thuộc của các nhãn lớp. Ý tưởng của phương pháp này là coi một tập con các nhãn như là một nhãn và tiến hành phân lớp như việc phân lớp dữ liệu đơn nhãn. Theo phương pháp này thì số lượng các tập con nhãn được tạo ra là rất lớn, Grigorios và đồng nghiệp [11] đã đề xuất phương pháp RAKEL với mục đích tính đến độ tương quan giữa các nhãn, đồng thời tránh những vấn đề nói trên của LP.

Định nghĩa tập K nhãn, cho tập nhãn L của phân lớp đa nhãn, L= {λi}, với i = 1…|L|. Một tập $Y \subseteq L$ với K = |L| gọi là tập K nhãn. Ta sử dụng giới hạn L^K là tập của tất cả tập nhãn K khác nhau trên L. Kích thước L^K cho bởi công thức: $|L^K| = \binom{|L|}{K}$.

Thuật toán RAKEL là cấu trúc toàn bộ của m phân loại LP, với i = 1 …m, chọn ngẫu nhiên một tập K nhãn, Yi, từ L^k. Sau đó, học phân loại LP $h_i: X \rightarrow P(Y_i)$. Thủ tục của RAKEL:

- **Đầu vào**: số của các mô hình m, kích thước của tập K nhãn, tập của các nhãn L, tập huấn luyện D.
- **Đầu ra**: toàn bộ của phân lớp LP h_i và tương ứng tập K nhãn Y_i; R←L^K;

for i ← 1 đến min (m, $|L^K|$) **do**

  Y_i ← một tập nhãn k ngẫu nhiên chọn từ R;

  Huấn luyện một phân lớp LP $h_i: X \rightarrow P(Y_i)$ trên D

  R ← R\ {Y_i} ;

Hình 2.2 Mã giả thuật toán RAKEL

Số của sự lặp lại (m) là một tham số cụ thể cùng dãy giá trị có thể chấp nhận được từ 1 tới $|L^K|$. Kích cỡ của tập K nhãn là một tham số cụ thể cùng dãy giá trị từ 2 tới |L| - 1. Cho K = 1 và m = |L| ta phân loại

toàn bộ nhị phân của phương pháp Binary Relevance, khi K = |L| (m = 1). Giả thiết việc sử dụng tập nhãn có kích thước nhỏ, số lặp vừa đủ, khi đó RAKEL sẽ quản lý để mô hình nhãn tương quan hiệu quả.

4. ClassifierChain (CC)

Thuật toán này bao gồm chuyển đổi nhị phân L như BR. Thuật toán này khác với thuật toán BR trong không gian thuộc tính cho mỗi mô hình nhị phân, nó được mở rộng cùng nhãn 0/1 cho tất cả phân lớp trước đó [8]. Ví dụ, chuyển đổi giữa BR và CC cho (x, y) với y = [1, 0, 0, 1, 0] và x = [0, 1, 0, 1, 0, 0, 1, 1, 0] (giả sử, cho đơn giản, không gian nhị phân). Mỗi phân loại hj được huấn luyện dự đoán yj ∈ {0, 1}.

Chuyển đổi nhị phân giữa BR và CC [8]

Chuyển đổi của BR		**Chuyển đổi của CC**	
h: x→	y	h: x'→	y
h1: [0, 1, 0, 1, 0, 0, 1, 1, 0]	1	h1: [0, 1, 0, 1, 0, 0, 1, 1, 0]	1
h2: [0, 1, 0, 1, 0, 0, 1, 1, 0]	0	h2: [0, 1, 0, 1, 0, 0, 1, 1, 0, **1**]	0
h3: [0, 1, 0, 1, 0, 0, 1, 1, 0]	0	h3: [0, 1, 0, 1, 0, 0, 1, 1, 0, **1, 0**]	0
h4: [0, 1, 0, 1, 0, 0, 1, 1, 0]	1	h4: [0, 1, 0, 1, 0, 0, 1, 1, 0, **1, 0, 0**]	1
h5: [0, 1, 0, 1, 0, 0, 1, 1, 0]	0	h5: [0, 1, 0, 1, 0, 0, 1, 1, 0, **1, 0, 0, 1**]	0