ТУБ 4

Многообразие задач, в которых необходимо распознавать образы путем их классификации, требует различных подходов к их решению. Рассмотрим несколько возможных типов задач, представления исходных данных в них и применения понятия «расстояния» для решения задачи классификации объектов.

4.1 Классификация объектов с бинарными признаками

Пусть некоторый образ X_i представляется в виде вектора $\overline{X}_i = \{x_{i1}, x_{i2}, ... x_{ik} ... x_{in}\}$, где первый индекс – это номер объекта, а второй индекс – номер признака. Образ можно записать в виде последовательности двоичных символов в соответствии с правилом: если i-ый объект обладает κ-ым признаком, то x_{ik}=1, иначе x_{ik}=0.

Рассмотрим пример, в котором исходные данные для классификации представлены в следующей таблице

Объект	Вектор	Желтый	Красный	Есть семечки	Есть косточки
Вишня	X_1	$x_{11}=0$	$x_{12}=1$	$x_{13}=0$	$x_{14}=1$
Яблоко	X_2	$x_{21}=1$	$x_{22}=1$	$x_{23}=1$	$x_{24}=0$
Банан	X_3	$x_{31}=1$	$x_{32}=0$	$x_{33}=0$	$x_{34}=0$

Эту таблицу можно представить как класс C, а множество векторов – $X=\{X_1, X_2, X_3\}$. В данном случае класс фруктов состоит из объединения непересекающихся подмножеств, каждое из которых включает в себя единственный объект. Поэтому такая классификация является тривиальной. Можно провести более тонкую классификацию, если для каждой пары объектов последовательно установить степень их сходства и различия. Тогда таблицу соответствия двух объектов X_i и X_j представим следующим образом

X_j	X_i	
	1	0
1	a	h
0	g	b

где a – число случаев, когда X_i и X_j имеют один и тот же признак

$$a = \sum_{k=1}^{n} x_{ik} x_{jk},$$

b – число случаев, когда X_i и X_j не имеют никаких общих признаков

$$b = \sum_{k=1}^{n} (1 - x_{ik})(1 - x_{jk}),$$

h – число случаев, когда X_i не имеет часть признаков, присущих X_j

$$h = \sum_{k=1}^{n} (1 - x_{ik}) x_{jk},$$

g – число случаев, когда X_i имеет признаки, отсутствующие у X_j

$$g = \sum_{k=1}^{n} x_{ik} (1 - x_{jk}).$$

Следовательно, чем больше сходств между объектами X_i и X_j, тем больше должен быть коэффициент a и тем больше отличаться от других коэффициентов. Можно ввести функцию сходства с набором свойств.

1. Функция возрастает в зависимости от a.
2. Функция симметрична относительно g и h.
3. Функция убывает в зависимости от b.

Подобную функцию, для которой, в зависимости от требований в поставленной задаче, известно несколько различных вариантов вычисления, называют двоичным расстоянием. Ниже приведено несколько способов вычисления функции сходства.

$$S_1(X_i, X_j) = \frac{a}{n}, \quad S_2(X_i, X_j) = \frac{a}{n-b}, \qquad S_3(X_i, X_j) = \frac{a}{2a+h+g},$$

$$S_4(X_i, X_j) = \frac{a}{a + 2(g + h)}, \qquad S_5(X_i, X_j) = \frac{a + b}{n},$$

$$S_6(X_i, X_j) = \frac{a}{g + h}, \quad S_7(X_i, X_j) = \frac{1}{2}\left[\frac{a}{a + g} + \frac{a}{a + h}\right],$$

$$S_8(X_i, X_j) = \frac{a}{\sqrt{(a + h)(a + b)}}, S_9(X_i, X_j) = \frac{ab - gh}{ab + gh}.$$

n – число признаков объектов.

В зависимости от используемых коэффициентов каждая из функций отражает те или иные признаки распознаваемых объектов. Так, функция $S_6 \to \infty$; $S_2, S_4, S_5, S_8, S_9 = 1$; $S_3 = 0.5$.

Введя функцию сходства, применим ее для классификации фруктов.

X_1=(0,1,0,1); X_2=(1,1,1,0); X_3=(1,0,0,0). Найдем сходства между каждой парой объектов: X_1 и X_2, X_1 и X_3, X_2 и X_3. Ниже приведены таблицы с вычисленными значениями переменных a, b, h, g.

X_2	X_1	
	1	0
1	1	2
0	1	0

X_3	X_1	
	1	0
1	0	1
0	2	1

X_3	X_2	
	1	0
1	1	0
0	2	1

Поскольку в данной задаче наобходимо определить максимальное сходство между объектами, воспользуемся для его определения функцией S_6, так как в ней не используется переменная b, отражающая различия объектов.

$$S_6(X_1, X_2) = \frac{a}{g+h} = \frac{1}{1+2} = 0{,}333,$$

$$S_6(X_1, X_3) = \frac{a}{g+h} = \frac{0}{1+2} = 0,$$

$$S_6(X_2, X_3) = \frac{a}{g+h} = \frac{1}{0+2} = 0{,}5.$$

В соответствии с выбранными признаками объектов и критериями отбора получаем, что объекты X_2 и X_3 больше всего схоже между собой.

4.2 Расстояние между списками

В некоторых задачах сходство между объектами базируется на мере сравнения порядка следования элементов в объектах-списках.

Пусть имеется два объекта, представленные следующими списками:

$$\bar{X}_i = (x_{i1}, x_{i2} \ldots x_{in});\ \bar{X}_j = (x_{j1}, x_{j2} \ldots x_{jn}).$$

Для их сравнения используются коэффициенты, которые определяются следующим образом:

$$\Delta_{lk}^{i} = \begin{cases} 1, \text{ при } x_{il} > x_{ik} \\ -1, \text{ при } x_{il} < x_{ik} \\ 0, \text{ при } x_{il} = x_{ik} \end{cases} \quad l<k.$$

Расстояние в этом случае вычисляется по формуле:

$$d(\bar{X}_i, \bar{X}_j) = 1 - \frac{2}{n(n-1)} \sum_{l<k} \Delta_{lk}^{j} \Delta_{lk}^{i}.$$

Если компоненты обоих списков упорядочены однотипно, то $\Delta_{lk}^{i} = \Delta_{lk}^{j}\ \forall l,k$. Расстояние между объектами в этом случае равно 0, что соответствует максимальному сходству между ними.

Рассмотрим пример. Даны два вектора X_1={0.5, 1, 2}, X_2={1, 3, 8}, требуется определить расстояние между ними предложенным выше способом. Вычисляются коэффициенты Δ_{lk}^{i} для каждого из векторов.

$\Delta_{1,2}^{1} = \Delta_{1,3}^{1} = \Delta_{2,3}^{1} = -1;\ \Delta_{1,2}^{2} = \Delta_{1,3}^{2} = \Delta_{2,3}^{2} = -1.$

Тогда расстояние между объектами будет равно

$$d(\bar{X}_1, \bar{X}_2) = 1 - \frac{2}{3(3-1)}[1 \times 1 + 1 \times 1 + 1 \times 1] = 0.$$

Максимальное расстояние между списками достигается, когда они упорядочены противоположно.

Рассмотри еще один способ определения расстояния между списками, связанный со «стоимостью» суммарного преобразования одного списка в другой. Чем меньше стоимость, тем меньше расстояние между списками и больше сходств между ними.

Пусть заданы два списка: $\bar{X}_i = (x_{i1}, x_{i2} \ldots x_{in})$; $\bar{X}_j = \left(x_{j1}, x_{j2} \ldots x_{jm}\right)$, m и n могут быть не равны. Задача состоит в нахождении функции, которая выражала бы степень сходства двух списков. Преобразование одного списка в другой выполняется с помощью трех операций, приведенных ниже. Также в преобразованиях используется пустой символ λ.

SUB (подстановка) $x_i \to x_j\ SUB\left(x_i, x_j\right)$.

DES (уничтожение) $x \to \lambda\ DES(x, \lambda)$.

CRE (создание) $\lambda\ \to x\ CRE(\lambda, x)$.

Каждой операции соответствует своя стоимость C. Для оценки расстояния между двумя списками вводят понятие полной стоимости последовательности преобразований от исходного списка к конечному. Процесс перехода от одного списка к другому может включать в себя все три операции.

Пусть имеется два списка $\overline{X}_i = (x_{i1}, x_{i2} ... x_{i5}); \overline{X}_j = (x_{j1}, x_{j2,} x_{j3})$ и установлены цены на каждую операцию преобразования одного элемента в другой:

$c(\lambda, x_{jk}) = 1\ \forall k$;

$c(x_{il}, \lambda) = 0.5\ \forall l$;

$c(x_{il}, x_{jk}) = 0\ \forall l,k$ при $x_{il} =\ x_{jk}$;

$c(x_{il}, x_{jk}) = 1\ \forall l,k$ при $x_{il} \neq\ x_{jk}$.

Рассмотрим пример определения сходства двух списков, выполнив преобразование первого списка во второй и оценив суммарную стоимость преобразования. X_i = {*a*,*a*,*b*,*a*,*c*}; X_j = {*a*,*b*,*d*}. Тогда один из возможных путей преобразования следующий:

1. $a \to a = 0.$
2. $a \to \lambda = 0.5$
3. $b \to b = 0.$
4. $a \to d = 1.$
5. $c \to \lambda = 0.5$

Суммарная стоимость преобразований равна 2.

4.3 Метод динамического программирования

Отличие данного метода от способов нахождения расстояния между списками заключается в том, что в этой процедуре одному элементу первого списка могут соответствовать несколько элементов второго списка. Пустые места в списках отсутствуют. Задача состоит в оценке сходств двух списков посредством нахождения расстояния между ними.

Даны два списка X и Y, состоящие из элементов:
$X(l) = a_1, a_2, \dots, a_l \dots;\ Y(k) = b_1, b_2, \dots, b_k \dots.$

Список X называется опорным, а Y – подлежащим сравнению (кандидатом). Количество значений списка X=n, Y=m. В общем случае $m \neq n$.

Метод динамического программирования может найти практическое применение в ситуациях, когда требуется распознать несколько реализаций какого-либо объекта или явления, при этом содержания у всех реализаций одинаковые, а внешние представления – разные. Например, при записи одного и того же слова, произнесенного с различной скоростью каждый раз; при анализе фотоснимков объекта, выполненных при разных увеличениях или условиях. Информационное содержание в каждом из случаев остается неизменным.

Решение задачи методом динамического программирования начинается с установления оптимального соответствия между двумя списками. Схема этого процесса приведена на рисунке 1.

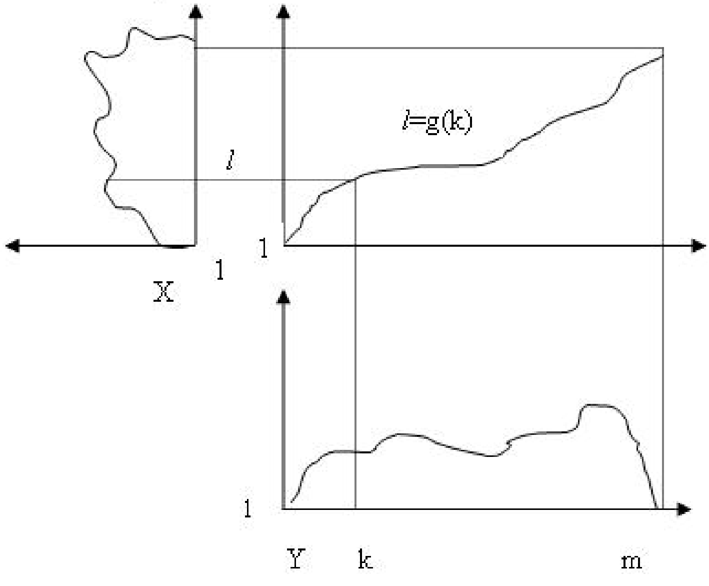

Рисунок 1 – Схема метода динамического программирования

Согласование опорного и предъявляемого списков выполняется с помощью функции g, связывающей l и k, $l=g(k)$. Примем $g(1)=1$; $g(m)=n$. В качестве первого приближения можно принять линейную зависимость $l = g(k) = (k-1)\frac{n-1}{m-1} + 1$. Более качественный способ согласования можно

представить так $D^* = \min_{g(k)}[\sum_{k=1}^{m} d(b(k), a[g(k)])]$, где $b(k)$ – $к$-ая составляющая вектора Y, которой соответствует $g(k)$-я составляющая вектора X.

Расстояние между этими составляющими вычисляется по одной из формул для расстояния. Функция $g(k)$ должна обеспечивать возможность минимизации в целом расстояния D_c между двумя списками. После того как оба списка исчерпаны, получают величину D^*, позволяющую оценить нормализованное расстояние между списками: $\delta(X,Y) = \frac{D^*}{n+m} = \frac{D_C(n,m)}{n+m}$. Ранее делается предположение, что D^* дает оптимальный результат. Фактически целью данного метода является определение пар точек из каждого списка, находящихся на минимальном расстоянии друг от друга.

Выражение для D_c означает, что $D_C(l,n) = d(a_l, b_k) + \min_{q \le k}[D_C(l-1), q)]$, т.е. $D_c(l, k)$ – текущее расстояние, накапливаемое от точки $l=k=1$ до точки с координатами (k, l) (рисунок 1).

Сначала $l=1$, $k=1$. $D(1,1) = 2d(1,1) = d(1,1) + \min(D_c(1,1))$. Затем $l=l+1$ и находится min расстояние от l до к, т.е. просматриваются все возможные значения $к$ на отрезке от 1 до $к$. Можно одновременно исследовать не всю плоскость, а некоторую ее часть (рисунок 2), положив условие $k - r \le l \le k + r$, где r выбирается заранее. Тогда сужается пространство значений $к$, которые нужно проверять для каждого конкретного l.

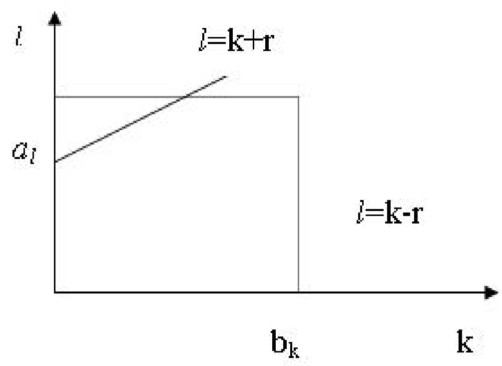

Рисунок 2 – Полоса области, выделенная для исследования