Multilabel классификация

Юрий Мокрий

12 декабря 2016 г.

Постановка задачи

Дано пространство объектов $\mathcal{X}$ и множество классов $\mathcal{Y}$:

$$\mathcal{Y} = \{y_1, y_2, \dots y_q\}$$

Каждому объекту может соответствовать несколько классов из $\mathcal{Y}$, т.е. пространство ответов - $2^{\mathcal{Y}}$.

Задача:

Есть обучающая выборка. Построить «качественный» алгоритм $a : \mathcal{X} \to 2^{\mathcal{Y}}$.

Области применения

- Тексты: определение жанров.
- Аудио: опредление жанра, эмоциальнольсть.
- Медицина: определение заболеваний по симптомам/истории болезней.
- Биоинформатика: определение функций гена.
- ...

Метрики качества

Существует два основных подхода.

- Example-based метрики - применяем к каждому объекту по отдельности и затем усредняем.
- Label-based метрики - применяем к каждому классу по отдельности и затем делаем микро/макро-усреднение.

Example-based метрики

- Subset accuracy

$$subsetacc(a, X) = \frac{1}{\ell} \sum_{i=1}^{\ell} [a(x_i) = Y_i]$$

- Hamming loss

$$hloss(a, X) = \frac{1}{\ell} \sum_{i=1}^{\ell} \frac{|a(x_i) \oplus Y_i|}{|\mathcal{Y}|}$$

- Accuracy

$$accuracy(a, X) = \frac{1}{\ell} \sum_{i=1}^{\ell} \frac{|a(x_i) \cap Y_i|}{|a(x_i) \cup Y_i|}$$

Example-based метрики

- Precision

$$precision(a, X) = \frac{1}{\ell} \sum_{i=1}^{\ell} \frac{|a(x_i) \cap Y_i|}{|a(x_i)|}$$

- Recall

$$recall(a, X) = \frac{1}{\ell} \sum_{i=1}^{\ell} \frac{|a(x_i) \cap Y_i|}{|Y_i|}$$

Example-based метрики

Если в результате классификации у нас есть $b(x, y_i)$ - степень принадлежность объекта x к y_i классу, то можно использовать следующие метрики:

- Ranking Loss

$$rloss(a, X) = \frac{1}{\ell} \sum_{i=1}^{\ell} \frac{|\{(y', y'') | b(x, y') < b(x, y''), y' \in Y_i, y'' \notin Y_i\}|}{|Y_i||\mathcal{Y} \setminus Y_i|}$$

Label-based метрики

Есть два подхода - макро- и микро-усреднение.

Рассмотрим стаститики TP, FP, TN, FN. Пусть метрика $B = B(TP, FP, TN, FN)$.

Пусть q - количество класса, индекс $k(TN_k)$ - статистика считается на классе k.

Тогда при макро-усреднении:

$$B_{macro} = \frac{1}{q} \sum_{k=1}^{q} B(TP_k, FP_k, TN_k, FN_k)$$

Микро-усреднение:

$$B_{micro} = B(\frac{1}{q} \sum_{k=1}^{q} TP_k, \frac{1}{q} \sum_{k=1}^{q} FP_k, \frac{1}{q} \sum_{k=1}^{q} TN_k, \frac{1}{q} \sum_{k=1}^{q} FN_k)$$

Такой вид имеют метрики accuracy, precision, recall, f-measure.

Два подхода

Существуют два основных подхода к решение задачи multi-label классификации:

- Сведение задачи к более простой (бинарная, multiclass классификация)
- Адаптация алгоритмов к multi-label классификации.

Binary Relevance (Независимые классификаторы)

Для каждого класса строим выборку: $D_k = \{(x_i, z_i) | 1 \leq i \leq \ell\}$, где $z_i = 1$, когда $y_k \in Y_i$, и $z_i = -1$, когда $y_k \notin Y_i$.

На каждой выборке обучаем свой бинарный классификатор - b_k.

Тогда итоговый алгоритм:

$$a(x) = \{y_k | b_k(x) = 1, 1 \leq k \leq q\}$$

Проблемы

- Несбалансированные выборки
- Не учитывает никаких взаимодействий классов

Classifier Chains

Каким-то образом упорядочим классы. Для первого класса обучим классификатор на обычных признаках. Для второго класса помимо обычных признаков добавим результаты первого классификатора. Для третьего - первого и второго, и т.д.

Проблемы

- Не распараллелить.
- Зависит от порядка классов.

Стэкинг

Поделим выборку на две части. На первой части обучаем те же классификаторы. Используем эти классификаторы для получения признаков на второй части. И уже на этих новых признаках обучаем классификаторы.

$$a(x) = \{y_k | b'_k(b_k(x)) = 1, 1 \leq k \leq \ell\}$$

Calibrated Label Ranking

Для каждой пары классов k и t строим выборку $D_{kt} = \{(x_i, z_{ik}) | z_{ik} \neq z_{it}\}$, где $z_{ik} = 1$, если $y_k \in Y_i$ и -1, если $y_k \notin Y_i$.

На каждой выборке обучим бинарный классификатор. Введём новое обозначение:

$$\zeta(x, k) = \sum_{\substack{i=1 \\ i \neq k}}^{q} [b_{ki}(x) = 1]$$

Как выбирать порог?

Порог

Добавим виртуальный класс - y_V. Построим выборки $D_{kV} = \{(x_i, z_{ik}) | 1 \leq i \leq \ell\}$, обучим на них бинарные классификаторы (фактически, мы для каждого класса провели one-vs-all классификацию).

Тогда:

$$\zeta^*(x, k) = \zeta(x, k) + [b_{kV} = 1]$$

$$\zeta^*(x, V) = \sum_{i=1}^{q} [b_{iV}(x) = -1]$$

Итоговый алгоритм:

$$a(x) = \{y_k | \zeta^*(x, k) > \zeta^*(x, V), 1 \leq k \leq q\}$$

Random k-labelsets

Попробуем свести multi-label классификацию к multi-class классификации. Для этого закодируем все множества классов из обучающей выборке.

Множество классов для multi-class классификации не больше $\min(\ell, 2^{\mathcal{Y}})$. Понятно, что это не очень хорошо работает.

- При предсказании получить только подмножества классов, встречающиеся в тренировочной выборке.
- Количество новых классов может быть очень большим (например, при $q = 10$ и $\ell = 1000$, все объекты могут иметь разные подмножества классов - в multi-class каждый объект имеет свой класс)

Random k-labelsets

Случайно генерируем n множеств $\mathcal{Y}^m \subseteq \mathcal{Y}$ и $|\mathcal{Y}^m| = m$.

Строим новые выборки: $D_k = \{(x_i, Y_i \cap \mathcal{Y}^m) | 1 \leq i \leq \ell\}$, где $k = 1 \dots n$.

Теперь повторим процедуру, описанную на предыдущем слайде - получим n multi-class классификаторов.

Пусть $\tau(x, y_k) = |\{\mathcal{Y}_i^m | y_k \in \mathcal{Y}_i^m\}|$, $\mu(x, y_k) = |\{i | y_k \in \sigma_i^{-1}(b_i(x))\}|$, где σ_k - функция кодировки k-ой выборки.

$$a(x) = \{y_k | \mu(x, y_k)/\tau(x, y_k) > 0.5\}$$

Рекомендуют делать $k = 3$ и $n = 2q$.

ML-kNN

Пусть $C_i = \sum_{(x^*,Y^*)\in\mathcal{N}(x)}[y_i \in Y^*]$ - количество объектов k-ого класса среди соседей.

Пусть H_i - событие того, что объект принадлежит классу y_i.

Апостериорная вероятность, того что объект принадлежит классу y_i - $p(H_i|C_i)$.

Если мы будем придерживаться MAP rule, решающее правило будет выглядеть так:

$$a(x) = \{y_i|P(H_i|C_i) > P(\neg H_i|C_i)\}$$

По теореме Байеса:

$$a(x) = \{y_i|P(H_i)P(C_i|H_i) > P(\neg H_i)P(C_i|\neg H_i)\}$$

ML-kNN

$$a(x) = \{y_i | P(H_i)P(C_i|H_i) > P(\neg H_i)P(C_i|\neg H_i)\}$$

Сделаем частотные оценки этих вероятностей.

$$P(H_i) = \frac{s + \sum_{j=1}^{\ell}[y_i \in Y_j]}{2s + \ell}$$

$$P(\neg H_i) = 1 - P(H_i)$$

s - параметр, контролирующий равномерность распределения. Обычно $s = 1$.

ML-kNN

$$\kappa_i(r) = \sum_{j=1}^{\ell} [y_i \in Y_j][\beta_i(Y_j) = r]$$

$$\tilde{\kappa}_i(r) = \sum_{j=1}^{\ell} [y_i \notin Y_j][\beta_i(Y_j) = r]$$

$\kappa_i(r)$ $(\tilde{\kappa}_i(r))$ - количество объектов в выборке, (не) принадлежащих к класс y_i и имеющих ровно r соседей, принадлежащих к классу y_i.

$$P(C_i|H_i) = \frac{s + \kappa_i(C_i)}{(k+1)s + \sum_{r=0}^{k} \kappa_i(r)}$$

$$P(C_i|\neg H_i) = \frac{s + \tilde{\kappa}_i(C_i)}{(k+1)s + \sum_{r=0}^{k} \tilde{\kappa}_i(r)}$$

ML-kNN

Недостатки:

- Рассматривает классы по отдельности
- Выбор метрики

ML-DecisionTree

Будем строить стандартными методами. В качестве критерия информативности будем использовать MLEntropy.

$$MLEnt(Y) = -\sum_{Y \subseteq \mathcal{Y}} P(Y) * log(P(Y)), P(Y) = \frac{\sum_{i=1}^{\ell}[Y_i = Y]}{\ell}$$

В таком виде критерий использовать не получится, т.к. подмножеств может быть слишком много.

Сделаем предположение о том, что классы независимы. Тогда критерий можно записать, как:

$$MLEnt(Y) = -\sum_{i=1}^{q}(p_i log(p_i) - (1 - p_i) * log(1 - p_i))$$

$$p_i = \frac{\sum_{j=1}^{\ell}[y_i \in Y_j]}{\ell}$$

Ml-DecisionTree

Преимущества

- Эффективность
- Хорошо работает в композициях

Недостатки

- Рассматривает классы независимо

RankSVM

Для каждого класса свой линейный классификатор:

$$a_i(x) = \langle w_i, x \rangle + b_i$$

Мы хотим хороший результат на RankingLoss.

multi-label отступ:

$$M(x_i, Y_i) = \min_{(y_j, y_k) \in Y_i \times \overline{Y_i}} \frac{\langle w_j - w_k, x \rangle + b_j - b_k}{||w_j - w_k||}$$

RankSVM

В идеальном случае, когда мы можем подобрать веса так, что классификатор для каждого объекта ранжирует принадлежащие ему классы выше, чем другие $\min_{(x_i,Y_i)\in D} M_i > 0$. Отнормируем веса так, чтобы $\langle w_j - w_k, x\rangle + b_j - b_k \geq 1$, и хотя бы одно равенство.

Тогда задача минимизации отступа сводится к:

$$\begin{cases} \min\limits_{(x_i,Y_i)\in\mathcal{D}} \min\limits_{(y_j,y_k)\in Y_i\times\overline{Y_i}} \frac{1}{||w_j - w_k||} \longrightarrow \max\limits_{w} \\ \langle w_j - w_k, x_i\rangle + b_j - b_k \geq 1 \end{cases}$$

Если все пары классов встречаются в выборке, то первое условие можно упростить до $\max\limits_{1\leq j<t\leq q} ||w_j - w_k||^2 \longrightarrow \min\limits_{w}$.

Для упрощения задачи оптимизации заменим максимум на сумму квадратов - $\sum_{k=1}^{q} ||w_k||^2 \longrightarrow \min\limits_{w}$.

RankSVM

$$\begin{cases} \sum_{k=1}^{q} ||w_k||^2 \longrightarrow \min\limits_{w} \\ \langle w_j - w_k, x_i \rangle + b_j - b_k \geq 1 \end{cases}$$

Перейдём к неидеальному случаю.

$$\begin{cases} \sum_{k=1}^{q} ||w_k||^2 + C \sum_{1}^{\ell} \dfrac{1}{|Y_i||\overline{Y_i}|} \sum_{(y_j, y_k) \in Y_i \times \overline{Y_i}} \xi_{i,jk} \longrightarrow \min\limits_{w,\xi} \\ \langle w_j - w_t, x \rangle + b_j - b_k \geq 1 - \xi_{i,jk} \\ \xi_{i,jk} \geq 0 \end{cases}$$

RankSVM

Преимущества:

- Kernel trick
- Работает лучше, чем Binary Relevance из SVM.

Здесь используется полиномиальное ядро со степенью degree.

	Rank-SVM				Binary-SVM			
degree	2	3	4	5	2	3	4	5
Precision	**0.703**	**0.740**	**0.746**	**0.762**	0.692	0.721	0.714	0.753
Ranking Loss	**0.227**	**0.191**	**0.190**	**0.175**	0.241	0.212	0.196	0.184
Hamming Loss	**0.238**	**0.217**	**0.209**	**0.201**	0.247	0.224	0.211	0.207
one-error	**0.334**	**0.262**	**0.255**	**0.232**	0.341	0.306	0.267	0.250

Summary

SUMMARY OF REPRESENTATIVE MULTI-LABEL LEARNING ALGORITHMS BEING REVIEWED.

Algorithm	Basic Idea	Order of Correlations	Complexity [Train/Test]	Tested Domains	Optimized Metric
Binary Relevance [5]	Fit multi-label data to q binary classifiers	first-order	$\mathcal{O}(q \cdot \mathcal{F}_{\mathcal{B}}(m,d))/$ $\mathcal{O}(q \cdot \mathcal{F}'_{\mathcal{B}}(d))$	image	classification (hamming loss)
Classifier Chains [72]	Fit multi-label data to a chain of binary classifiers	high-order	$\mathcal{O}(q \cdot \mathcal{F}_{\mathcal{B}}(m,d+q))/$ $\mathcal{O}(q \cdot \mathcal{F}'_{\mathcal{B}}(d+q))$	image, video text, biology	classification (hamming loss)
Calibrated Label Ranking [30]	Fit multi-label data to $\frac{q(q+1)}{2}$ binary classifiers	second-order	$\mathcal{O}(q^2 \cdot \mathcal{F}_{\mathcal{B}}(m,d))/$ $\mathcal{O}(q^2 \cdot \mathcal{F}'_{\mathcal{B}}(d))$	image, text biology	Ranking (ranking loss)
Random k-Labelsets [94]	Fit multi-label data to n multi-class classifiers	high-order	$\mathcal{O}(n \cdot \mathcal{F}_{\mathcal{M}}(m,d,2^k))/$ $\mathcal{O}(n \cdot \mathcal{F}'_{\mathcal{M}}(d,2^k))$	image, text biology	classification (subset accuracy)
ML-kNN [108]	Fit k-nearest neighbor to multi-label data	first-order	$\mathcal{O}(m^2 d + qmk)/$ $\mathcal{O}(md+qk)$	image, text biology	classification (hamming loss)
ML-DT [16]	Fit decision tree to multi-label data	first-order	$\mathcal{O}(mdq)/\mathcal{O}(mq)$	biology	classification (hamming loss)
Rank-SVM [27]	Fit kernel learning to multi-label data	second-order	$\mathcal{O}(\mathcal{F}_{\mathrm{QP}}(dq+mq^2, mq^2)$ $+q^2(q+m))/\mathcal{O}(dq)$	biology	Ranking (ranking loss)

Литература

- A kernel method for multi-labelled classification. Andre Elisseeff and Jason Weston
- A Review on Multi-Label Learning Algorithms. Min-Ling Zhang and Zhi-Hua Zhou.