Урок 9

Ранжирование

9.1. Задача ранжирования

Этот урок посвящён задаче ранжирования. Она имеет много общего с уже рассмотренными ранее задачами классификации и регрессии, но есть и ряд особенностей, которые и будут рассмотрены в данном уроке.

9.1.1. Ранжирование

Формальная постановка задачи ранжирования выглядит следующим образом. Имеется выборка, состоящая из l элементов:

$$x_1, \ldots, x_l.$$

Она состоит из объектов, имеющих признаковое описание, как и в задачах регрессии и классификации.

Ключевое отличие ранжирования заключается в целевой переменной. Если ранее в задачах обучения с учителем каждому объекту соответствовал свой ответ, то теперь ответы — это пары вида:

$$\{(i, j) : x_i < x_j\}.$$

Набор таких пар и является целевой переменной.

В задаче требуется построить ранжирующую модель $a(x)$, такую, что:

$$x_i < x_j \Rightarrow a(x_i) < a(x_j).$$

Таким образом, по выходам необходимо восстановить порядок, а величина ответов не имеет значения. В этом заключается главное отличие ранжирования от остальных задач машинного обучения.

9.1.2. Ранжирование поисковой выдачи

Основное применение ранжирования — это ранжирование поисковой выдачи. Например, в Яндексе в качестве объектов выступают пары запрос, документ. Запрос — это ключевые слова, введённые пользователем, документ — один из всех документов, имеющихся в поисковом индексе:

$$(\text{запрос, документ}_1) < (\text{запрос, документ}_2).$$

Задача — научиться восстанавливать порядок на таких парах, причём запрос во всех парах один и тот же. Таким образом, требуется ранжировать документы для одного запроса.

Для обучение такой порядок задаётся асессорами. Это специальные люди, которые получают пары запрос, документ и оценивают релевантность документа, то, насколько хорошо он отвечает данному запросу. Оценки могут быть как числовыми, так и попарными (асессор в паре документов выбирает тот, который более релевантен запросу).

9.1.3. Рекомендации

Другой пример применения ранжирования — это рекомендации. В этой задаче по парам пользователь, товар нужно ранжировать товары для данного пользователя, максимизируя при этом некоторую метрику. Порядок задаётся на основании предпочтений пользователя, его предыдущих покупок и оценок.

9.1.4. Особенности задачи

У задачи ранжирования есть свои особенности. Во-первых, объекты не являются независимыми: целевая переменная зависит от пар объектов. Это необходимо учитывать при решении. Во-вторых, используются более сложные метрики, чем для регрессии или классификации. Обычно они дискретные, потому что важны не получаемые значения, а порядок, который задаёт модель. Ещё одна особенность — для этой задачи очень важно правильно сформировать выборку, потому что это можно сделать множеством способов. Непонятно, например, какие именно пары отдавать на разметку асессорам в задаче ранжирования поисковой выдачи, или как именно получать предпочтения пользователей из данных для задачи построения рекомендаций.

9.2. Метрики качества ранжирования

9.2.1. Точность ранжирования

Для простоты всюду далее в качестве примера будет использоваться задача ранжирования поисковой выдачи, но все рассуждения хорошо обобщаются и для других применений.

В самой простой постановке задачи ранжирования целевая переменная принимает два значения, документ либо релевантен запросу, либо нет:

$$y(q, d) \in \{0, 1\},$$

где y — целевая переменная, q — запрос, d — документ.

Пусть также есть некоторая модель $a(q, d)$, оценивающая релевантность документа запросу. По значениям, полученным с помощью этой модели, можно отранжировать документы. $d_q^{(i)}$ будет обозначать i-й по релевантности документ для запроса q.

После того как введены обозначения, можно задать простейшую метрику ранжирования. Это Precision@k, точность среди первых k документов (k — параметр метрики). Если ранжируется поисковая выдача, и на первой странице показываются 10 документов, то разумно выбирать $k = 10$. Данная метрика определяется как доля релевантных документов среди первых k, полученных с помощью модели:

$$\text{Precision@}k(q) = \frac{1}{k} \sum_{i=1}^{k} y\left(q, d_q^{(i)}\right),$$

Это полезная метрика, потому что обычно важно иметь релевантные документы среди первых k. Однако у неё есть серьёзный недостаток: позиции релевантных документов никак не учитываются. Например, если при $k = 10$ среди первых k документов есть 5 релевантных, то неважно, где они находятся: среди первых или последних 5 документов. Обычно же хочется, чтобы релевантные документы располагались как можно выше.

Описанную проблему можно решить, модифицировав метрику, и определить среднюю точность (average precision, AP). Данная метрика тоже измеряется на уровне k и вычисляется следующим образом:

$$\text{AP@}k(q) = \frac{\sum_{i=1}^{k} y\left(q, d_q^{(i)}\right) \text{Precision@}i(q)}{\sum_{i=1}^{k} y\left(q, d_q^{(i)}\right)}.$$

Данная величина уже зависит от порядка. Она достигает максимума, если все релевантные документы находятся вверху ранжированного списка. Если они смещаются ниже, значение метрики уменьшается.

И точность, и средняя точность вычисляются для конкретного запроса q. Если выборка большая и размечена для многих запросов, то значения метрик усредняются по всем запросам:

$$\text{MAP@}k = \frac{1}{|Q|} \sum_{q \in Q} \text{AP@}k(q).$$

9.2.2. DCG

Второй подход к измерению качества ранжирования — это метрика DCG (discounted cumulative gain). Она используется в более сложной ситуации, когда оценки релевантности y могут быть вещественными:

$$y(q, d) \in \mathbb{R}.$$

То есть для каждого документа теперь существует градация между релевантностью и нерелевантностью. Остальные обозначения остаются теми же, что и для предыдущей метрики.

Формула для вычисления DCG:

$$\mathrm{DCG@}k(q) = \sum_{i=1}^{k} \frac{2^{y\left(q, d_q^{(i)}\right)} - 1}{\log(i+1)}.$$

Метрика — это сумма дробей. Чем более релевантен документ, тем больше числитель в дроби. Знаменатель зависит от позиции документа, он штрафует за то, где находится документ. Если документ очень релевантен, но занимает низкую позицию, то штраф будет большим, если документ релевантен и находится вверху списка, штраф будет маленьким. Таким образом, метрика DCG учитывает и релевантность, и позицию документа. Она достигает максимума, если все релевантные документы находятся в топе списка, причём отсортированные по значению y.

Данную метрику принято нормировать:

$$n\mathrm{DCG@}k(q) = \frac{\mathrm{DCG@}k(q)}{\max \mathrm{DCG@}k(q)},$$

где $\max \mathrm{DCG@}k(q)$ — значение DCG при идеальном ранжировании. После нормировки метрика принимает значения от 0 до 1.

9.3. Методы ранжирования

Всего выделяют три подхода к решению задачи ранжирования: pointwise (поточечный), pairwise (попарный), listwise (списочный). Далее будут приведены по одному методу из каждого подхода, чтобы можно было составить представления об их различиях и особенностях.

9.3.1. Поточечный подход

Самый простой подход — это поточечный. В нём игнорируется тот факт, что целевая переменная $y(q, d) \in \mathbb{R}$ задаётся на парах объектов, и оценка релевантности $a(d, q)$ оценивается непосредственно для каждого объекта.

Если речь идёт о задаче ранжирования, то пусть асессор поставил какую-то оценку y каждой паре запрос, документ. Эта оценка и будет предсказываться. При этом никак не учитывается, что на самом деле нужно предсказать порядок объектов, а не оценки. Этот подход является простым в том смысле, что в нём используются уже известные методы. Например, можно предсказывать оценки с использованием линейной регрессии и квадратичной ошибки:

$$\sum_{i=1}^{\ell} \left(a(q_i, d_i) - y(q_i, d_i)\right)^2 \to \min.$$

Известно, как решать такую задачу, и таким образом будет получена релевантность. Далее по выходам модели можно ранжировать объекты.

9.3.2. Попарный подход

В попарном подходе используются знания об устройстве целевой переменной. Модель строится минимизацией количества дефектных пар, то есть таких, в которых моделью был предсказан неправильный порядок:

$$\sum_{x_i < x_j} [a(x_j) - a(x_i) < 0] \to \min.$$

К сожалению, этот функционал дискретный (в него входят индикаторы), поэтому не получится минимизировать непосредственно его. Однако можно действовать так же, как и с классификаторами: оценить функционал сверху.

Можно считать, что разница между объектами $a(x_j) - a(x_i)$ — это отступ M, и задать некоторую гладкую функцию $L(M)$:

$$\sum_{x_i < x_j} L(a(x_j) - a(x_i)) \to \min.$$

Если использовать функцию как в логистической регрессии $L(M) = \log(1 + e^{-M})$, то полученный метод называется RankNet. Затем можно решать задачу, например, с помощью стохастического градиентного спуска.

9.3.3. Listwise-подход

В методе RankNet шаг стохастического градиентного спуска для линейной модели выглядит следующим образом:

$$w := w + \eta \frac{1}{1 + \exp(\langle w, x_j - x_i \rangle)} (x_j - x_i).$$

Это не очень сложная формула, она зависит от одной пары объектов. Возникает вопрос, можно ли модифицировать данный метод (а именно формулу шага) так, чтобы минимизировался не исходный функционал, оценивающий долю дефектных пар, а DCG.

Ответ на этот вопрос положительный. Можно домножить градиент исходного функционала на то, насколько изменится NDCG, если поменять местами x_i и x_j:

$$w := w + \eta \frac{1}{1 + \exp(\langle w, x_j - x_i \rangle)} (x_j - x_i) \cdot |\Delta \mathrm{NDCG}_{ij}| (x_j - x_i).$$

Оказывается, что при выполнении градиентного спуска с помощью данных шагов оптимизируется NDCG. Это эмпирический факт, и он не доказан. Но на практике NDCG действительно улучшается при решении задачи данным методом.

Существуют и другие подходы к оптимизации NDCG, однако в них предпринимается попытка работы с функционалом, что гораздо сложнее. Выше описан самый простой подход, он называется LambdaRank.