Введение в анализ данных, майнор
Семинар №8

За основу взят аналогичный семинар с курса МО-1 ФКН НИУ ВШЭ

1 Логистическая регрессия

Везде далее мы будем требовать, чтобы алгоритм $b(x)$ возвращал числа из отрезка $[0, 1]$ (вероятность принадлоежности объекта к положительному классу). Этого легко достичь, если положить $b(x) = \sigma(\langle w, x\rangle)$, где в качестве σ может выступать любая монотонно неубывающая функция с областью значений $[0, 1]$. Мы будем использовать сигмоидную функцию: $\sigma(z) = \frac{1}{1+\exp(-z)}$. Таким образом, чем больше скалярное произведение $\langle w, x\rangle$, тем больше будет предсказанная вероятность. Как при этом можно интерпретировать данное скалярное произведение? Чтобы ответить на этот вопрос, преобразуем уравнение

$$p(y = 1 \,|\, x) = \frac{1}{1 + \exp(-\langle w, x\rangle)}.$$

Выражая из него скалярное произведение, получим

$$\langle w, x\rangle = \log \frac{p(y = +1 \,|\, x)}{p(y = -1 \,|\, x)}.$$

Получим, что скалярное произведение будет равно логарифму отношения вероятностей классов (log-odds).

Как уже упоминалось выше, при использовании квадратичной функции потерь алгоритм будет пытаться предсказывать вероятности, но данная функция потерь является далеко не самой лучшей, поскольку слабо штрафует за грубые ошибки. Логарифмическая функция потерь подходит гораздо лучше, поскольку не позволяет алгоритму сильно ошибаться в вероятностях.

Подставим трансформированный ответ линейной модели в логарифмическую функцию потерь:

$$-\sum_{i=1}^{\ell}\left([y_i=+1]\log\frac{1}{1+\exp(-\langle w,x_i\rangle)}+[y_i=-1]\log\frac{\exp(-\langle w,x_i\rangle)}{1+\exp(-\langle w,x_i\rangle)}\right)=$$
$$=-\sum_{i=1}^{\ell}\left([y_i=+1]\log\frac{1}{1+\exp(-\langle w,x_i\rangle)}+[y_i=-1]\log\frac{1}{1+\exp(\langle w,x_i\rangle)}\right)=$$
$$=\sum_{i=1}^{\ell}\log\left(1+\exp(-y_i\langle w,x_i\rangle)\right).$$

Полученная функция в точности представляет собой логистические потери, упомянутые в начале. Линейная модель классификации, настроенная путём минимизации данного функционала, называется логистической регрессией. Как видно из приведенных рассуждений, она оптимизирует правдоподобие выборки и дает корректные оценки вероятности принадлежности к положительному классу.

2 Многоклассовая классификация

Ранее мы разобрались с общим подходом к решению задачи бинарной классификации, а также изучили свойства двух конкретных методов: логистической регрессии и метода опорных векторов. Теперь мы перейдём к более общей, многоклассовой постановке задачи классификации, и попытаемся понять, как можно свести её к уже известным нам методам.

Также мы обсудим методы работы с категориальными признаками и поймём, почему на таких данных чаще всего используются линейные модели.

В данном разделе будем считать, что каждый объект относится к одному из K классов: $\mathbb{Y}=\{1,\dots,K\}$.

§2.1 Сведение к серии бинарных задач

Мы уже подробно изучили задачу бинарной классификации, и поэтому вполне естественно попытаться свести многоклассовую задачу к набору бинарных. Существует достаточно много способов сделать это — мы перечислим лишь два самых популярных, а об остальных можно почитать, например, [**?**, раздел 4.2.7]

Один против всех (one-versus-all). Обучим K линейных классификаторов $b_1(x),\dots,b_K(x)$, выдающих оценки принадлежности классам $1,\dots,K$ со-

ответственно. Например, в случае с линейными моделями эти модели будут иметь вид

$$b_k(x) = \langle w_k, x\rangle + w_{0k}.$$

Классификатор с номером k будем обучать по выборке $(x_i, 2[y_i = k] - 1)_{i=1}^{\ell}$; иными словами, мы учим классификатор отличать k-й класс от всех остальных.

Итоговый классификатор будет выдавать класс, соответствующий самому уверенному из бинарных алгоритмов:

$$a(x) = \arg\max_{k\in\{1,\ldots,K\}} b_k(x).$$

Проблема данного подхода заключается в том, что каждый из классификаторов $b_1(x), \ldots, b_K(x)$ обучается на своей выборке, и выходы этих классификаторов могут иметь разные масштабы, из-за чего сравнивать их будет неправильно [**?**]. Нормировать вектора весов, чтобы они выдавали ответы в одной и той же шкале, не всегда может быть разумным решением — так, в случае с SVM веса перестанут являться решением задачи, поскольку нормировка изменит норму весов.

Все против всех (all-versus-all). Обучим C_K^2 классификаторов $a_{ij}(x)$, $i, j = 1, \ldots, K$, $i \neq j$. Например, в случае с линейными моделями эти модели будут иметь вид

$$b_k(x) = \text{sign}(\langle w_k, x\rangle + w_{0k}).$$

Классификатор $a_{ij}(x)$ будем настраивать по подвыборке $X_{ij} \subset X$, содержащей только объекты классов i и j:

$$X_{ij} = \{(x_n, y_n) \in X \mid [y_n = i] = 1 \text{ или } [y_n = j] = 1\}.$$

Соответственно, классификатор $a_{ij}(x)$ будет выдавать для любого объекта либо класс i, либо класс j.

Чтобы классифицировать новый объект, подадим его на вход каждого из построенных бинарных классификаторов. Каждый из них проголосует за своей класс; в качестве ответа выберем тот класс, за который наберется больше всего голосов:

$$a(x) = \arg\max_{k\in\{1,\ldots,K\}} \sum_{i=1}^{K} \sum_{j\neq i} [a_{ij}(x) = k].$$

§2.2 Многоклассовая логистическая регрессия

Некоторые методы бинарной классификации можно напрямую обобщить на случай многих классов. Выясним, как это можно проделать с логистической регрессией.

В логистической регрессии для двух классов мы строили линейную модель $b(x) = \langle w, x\rangle + w_0$, а затем переводили её прогноз в вероятность с помощью сигмоидной функции $\sigma(z) = \frac{1}{1+\exp(-z)}$. Допустим, что мы теперь решаем многоклассовую задачу и построили K линейных моделей $b_k(x) = \langle w_k, x\rangle + + w_{0k}$, каждая из которых даёт оценку принадлежности объекта одному из классов. Как преобразовать вектор оценок $(b_1(x), \ldots, b_K(x))$ в вероятности? Для этого можно воспользоваться оператором $\mathrm{SoftMax}(z_1, \ldots, z_K)$, который производит «нормировку» вектора:

$$\mathrm{SoftMax}(z_1, \ldots, z_K) = \left(\frac{\exp(z_1)}{\sum_{k=1}^K \exp(z_k)}, \ldots, \frac{\exp(z_K)}{\sum_{k=1}^K \exp(z_k)} \right).$$

В этом случае вероятность k-го класса будет выражаться как

$$P(y = k \,|\, x, w) = \frac{\exp\left(\langle w_k, x\rangle + w_{0k}\right)}{\sum_{j=1}^K \exp\left(\langle w_j, x\rangle + w_{0j}\right)}.$$

Обучать эти веса предлагается с помощью метода максимального правдоподобия — так же, как и в случае с двухклассовой логистической регрессией:

$$\sum_{i=1}^{\ell} \log P(y = y_i \,|\, x_i, w) \to \max_{w_1,\ldots,w_K}.$$

3 Категориальные признаки

Допустим, в выборке имеется категориальный признак, значение которого на объекте x будем обозначать через $f(x)$. Будем считать, что он принимает значения из множества $U = \{u_1, \ldots, u_n\}$. Чтобы использовать такой признак в линейных моделях, необходимо сначала его закодировать. Существует много подходов к использованию категориальных признаков — о многих из них можно узнать в работе [?], мы же рассмотрим несколько наиболее популярных.

§3.1 Бинарное кодирование (one-hot encoding)

Простейший способ — создать n индикаторов, каждый из которых будет отвечать за одно из возможных значений признака. Иными словами, мы

формируем n бинарных признаков $g_1(x), \ldots, g_n(x)$, которые определяются как

$$g_j(x) = [f(x) = u_j].$$

Главная проблема этого подхода заключается в том, что на выборках, где категориальные признаки имеют миллионы возможных значений, мы получим огромное количество признаков. Линейные модели хорошо справляются с такими ситуациями за счёт небольшого количества параметров и достаточно простых методов обучения, и поэтому их часто используют на выборках с категориальными признаками.

§3.2 Бинарное кодирование с хэшированием

Рассмотрим модификацию бинарного кодирования, которая позволяет ускорить процесс вычисления признаков. Выберем хэш-функцию $h : U \to \{1, 2, \ldots, B\}$, которая переводит значения категориального признака в числа от 1 до B. После этого бинарные признаки можно индексировать значениями хэш-функции:

$$g_j(x) = [h(f(x)) = j], \quad j = 1, \ldots, B.$$

Основное преимущество этого подхода состоит в том, что отпадает необходимость в хранении соответствий между значениями категориального признака и индексами бинарных признаков. Теперь достаточно лишь уметь вычислять саму хэш-функцию, которая уже автоматически даёт правильную индексацию.

Также хэширование позволяет понизить количество признаков (если $B < |U|$), причём, как правило, это не приводит к существенной потере качества. Это можно объяснить и с помощью интуиции — если у категориального признака много значений, то, скорее всего, большая часть этих значений крайне редко встречается в выборке, и поэтому не несёт в себе много информации; основную ценность представляют значения $u \in U$, которые много раз встречаются в выборке, поскольку для них можно установить связи с целевой переменной. Хэширование, по сути, случайно группирует значения признака — в одну группу попадают значения, получающие одинаковые индексы $h(u)$. Поскольку «частых» значений не так много, вероятность их попадания в одну группу будет ниже, чем вероятность группировки редких значений.

§3.3 Счётчики

Попытаемся закодировать признаки более экономно. Заметим, что значения категориального признака нужны нам не сами по себе, а лишь для

предсказания класса. Соответственно, если два возможных значения u_i и u_j характерны для одного и того же класса, то можно их и не различать.

Определим наш способ кодирования. Вычислим для каждого значения u категориального признака $(K+1)$ величин:

$$\text{counts}(u, X) = \sum_{(x,y)\in X} [f(x) = u],$$
$$\text{successes}_k(u, X) = \sum_{(x,y)\in X} [f(x) = u][y = k], \quad k = 1, \ldots, K.$$

По сути, мы посчитали количество объектов с данным значением признака, а также количество объектов различных классов среди них.

После того, как данные величины подсчитаны, заменим наш категориальный признак $f(x)$ на K вещественных $g_1(x), \ldots, g_K(x)$:

$$g_k(x, X) = \frac{\text{successes}_k(f(x), X) + c_k}{\text{counts}(f(x), X) + \sum_{m=1}^{K} c_m}, \quad k = 1, \ldots, K.$$

Здесь признак $g_k(x)$ фактически оценивает вероятность $p(y = k \mid f(x))$. Величины c_k являются своего рода регуляризаторами и предотвращают деление на ноль в случае, если в выборке X нет ни одного объекта с таким значением признака. Для простоты можно полагать их все равными единице: $c_1 = \cdots = c_K = 1$. У признаков $g_k(x)$ есть много названий: счётчики [1], правдоподобия и т.д.

Отметим, что $g_k(x)$ можно воспринимать как простейший классификатор — значит, при обучении полноценного классификатора на признаках-счётчиках мы рискуем столкнуться с переобучением из-за «утечки» целевой переменной в значения признаков (мы уже обсуждали эту проблему, разбираясь со стекингом). Чтобы избежать переобучения, как правило, пользуются подходом, аналогичным кросс-валидации. Выборка разбивается на m частей $X_1, \ldots, X_m$, и для подвыборки X_i значения признаков вычисляются на основе статистик, подсчитанных по всем остальным частям:

$$x \in X_i \Rightarrow g_k(x) = g_k(x, X \setminus X_i).$$

Можно взять число блоков разбиения равным числу объектов $m = \ell$ — в этом случае значения признаков для каждого объекта будут вычисляться по статистикам, подсчитанным по всем остальным объектам.

[1] http://blogs.technet.com/b/machinelearning/archive/2015/02/17/big-learning-made-easy-with-counts.aspx

Для тестовой выборки значения целевой переменной неизвестны, поэтому на таких объектах признаки-счётчики вычисляются на основе статистик $\text{successes}(u, X)$ и $\text{counts}(u, X)$, подсчитанных по всей обучающей выборке.

При использовании счётчиков нередко используют следующие трюки:

1. К признакам можно добавлять не только дроби $g_k(x)$, но и значения $\text{counts}(f(x), X)$ и $\text{successes}_k(f(x), X)$.

2. Можно сгенерировать парные категориальные признаки, т.е. для каждой пары категориальных признаков $f_i(x)$ и $f_j(x)$ создать новый признак $f_{ij}(x) = (f_i(x), f_j(x))$. После этого счётчики можно вычислить и для парных признаков; при этом общее количество признаков существенно увеличится, но при этом, как правило, прирост качества тоже оказывается существенным.

3. Если у категориальных признаков много возможных значений, то хранение статистик $\text{counts}(u, X)$ и $\text{successes}_k(u, X)$ может потребовать существенного количества памяти. Для экономии памяти можно хранить статистику не по самим значениям категориального признака $u \in U$, а по хэшам от этих значений $h(u)$. Регулируя количество возможных значений хэш-функции, можно ограничивать количество используемой памяти.

4. Можно вычислять несколько счётчиков для разных значений параметров $c_1, \ldots, c_K$.

5. Можно все редкие значения категориального признака объединить в одно, поскольку скорее всего, для редких значений не получится качественно оценить статистики successes_k и counts. Благодаря этому можно будет сократить расходы на память при хранении статистики. Более того, можно предположить, что все редкие значения похожи, и относить к данной «объединённой» группе и новые значения признака, которые впервые встретятся на тестовой выборке.

Отметим, что данный подход работает только для задач классификации. В то же время можно пытаться адаптировать его и для задач регрессии, вычисляя несколько бинаризаций целевой переменной по разным порогам, и для каждой такой бинаризации вычисляя счётчики.