

Глава 6

Альтернативы МНК и обобщенная линейная модель

6.1 Робастная регрессия

К сожалению, линейная регрессионная модель, основанная на квадратичных отклонениях, крайне неустойчива к выбросам.

М-оценки

Можем решать более общую задачу, рассматривая задачу минимизации выражения

$$\sum_{i=1}^n \rho(r_i), \quad r_i = y_i - \langle x_i, b \rangle.$$

где ρ — некоторая функция или же, в случае дифференцируемых ρ , решение уравнения

$$\sum_{i=1}^n \psi(r_i) = 0, \quad \psi(x) = \rho'(x).$$

Разумеется, ρ и ψ выбираются с точностью до положительного множителя.

Зачастую рассматривают форму от стандартизованных остатков

$$\sum_{i=1}^n \rho(\hat{r}_i) \rightarrow \min, \quad \hat{r}_i = \frac{y_i - \langle x_i, b \rangle}{\hat{\sigma}},$$

где $\hat{\sigma}$ — некоторая оценка масштаба.

Рассматривая другие функции потерь вместо квадратичных, мы можем получать другие оценки:

1. Метод наименьшего абсолютного отклонения (Least Absolute Deviation или LAD) предлагает использовать

$$\rho(x) = |x|$$

2. Метод Хубера предлагает использовать

$$\rho(x) = \begin{cases} x^2, & |x| \leq c, \\ c(2|x| - c), & |x| > c. \end{cases}, \quad \psi(x) = \begin{cases} 2x, & |x| \leq c, \\ 2 \operatorname{sgn} x, & |x| > c. \end{cases},$$

В качестве c в случае стандартизованных остатков предлагается использовать $c = 1.345$.

3. Метод Тьюки предлагает использовать

$$\rho(x) = \begin{cases} x^2((x/c)^4 - 3(x/c)^2 + 3)/6, & |x| \leq c, \\ c^2/6, & |x| > c. \end{cases}, \quad \psi(x) = \begin{cases} x(1 - (x/c)^2)^2, & |x| \leq c, \\ 0, & |x| > c. \end{cases}.$$

В качестве c в случае стандартизованных остатков предлагается использовать $c = 4.685$

В Python регрессия Хубера представлены в методе HuberRegressor в sklearn.linear_model. При этом минимизируется величина

$$\sum_{i=1}^n \left(\sigma + H\left(\frac{\langle X_i, \vec{a} \rangle - y_i}{\sigma}\right) \sigma \right) + \alpha \|\vec{a}\|^2.$$

Это ридж-версия регрессии, включающая к тому же оценку параметра σ для стандартизации.

LAD является частным случаем более общей квантильной регрессии, использующей

$$\rho_\alpha(x) = x(\alpha - I_{x<0})$$

При $\alpha = 1/2$ мы получим LAD, при $\alpha \neq 0.5$ ошибки вверх и вниз будут трактоваться по-разному. Скажем, при $\alpha = 1/3$ при $x < 0$ мы получим $2|x|/3$ при $x < 0$ и $|x|/3$ при $x > 0$, то есть отрицательные ошибки стоят вдвое дороже, чем положительные. Квантильная регрессия реализована в Python в sklearn.linear_model.QuantileRegressor. Здесь используется Lasso-регуляризация.

Квантильная регрессия, а также регрессия Тьюки и Хубера не позволяют получить аналитического решения соответствующих задач и решаются численно.

Задание 1. Положим $x_i = i/10$, $i \leq 100$, $x_{101} = 50$, $y_i \sim \mathcal{N}(x_i, 1)$, $i \leq 100$, $y_{101} \sim \mathcal{N}(150, 1)$. Сгенерировать выборки, построить линейную регрессионную модель и модель Хубера, а также квантильные регрессии с $\alpha = 0.3$, $\alpha = 0.5$.

Некоторые другие подходы

- RANSAC (Random Sample Consensus) стоит группу моделей по различным поднаборам данных. Строя модель по какому-то набору точек, мы называем часть остальных точек выбросами если модуль остатка регрессии оказывается больше некоторого порога. Модель, дающая минимальное число выбросов, объявляется требуемой. В Python этот метод есть в sklearn.
- Метод Тэйла-Сена в двумерном случае предлагает рассматривать всевозможные прямые, проходящие через пары точек (x_i, y_i) , (x_j, y_j) и выбирать в качестве наклона регрессионной прямой медиану наклона всех указанных:

$$\hat{a} = \frac{y_i - y_j}{x_i - x_j}, \quad \hat{b} = \text{med}(y_i - \hat{a}x_i).$$

В многомерном случае в соответствующей работе предложено проводить плоскость через каждые k точек, а затем брать пространственную медиану от полученного набора коэффициентов.

Под пространственной медианой здесь понимается

$$\vec{a} : \mathbf{E}(\|\vec{X} - \vec{a}\| - \|\vec{X}\|) \rightarrow \min.$$

Вычитание $\|\vec{X}\|$ здесь нужно, чтобы сделать математическое ожидание конечным. Такая медиана единственна коль скоро распределение \vec{X} не сосредоточено на прямой.

В более общей версии алгоритма проведение плоскости заменяется на построение МНК оценки на основе m -элементных множеств для всех возможных таких подмножеств. Метод реализован в функции Theil Sen все в том же scikit-learn в linear_model.

- Метод усеченных квадратов (LTS = least trimmed squares) предлагает рассматривать

$$\sum_{i=1}^m r_{(i)}^2,$$

где $r_{(i)}$ – остатки регрессии, упорядоченные по возрастанию. Тем самым, мы берем сумму квадратов минимальных остатков, но не учитываем часть самых больших. В качестве m принято брать $[n/2]$

или $[n/2] + [(k+1)/2]$. Точность у метода крайней низкая, поэтому его используют как начальное приближение или в случаях, когда данные очень сильно зашумлены.

В Python этот метод реализован здесь.

Мы привели лишь некоторый из огромного числа М-оценок и других рабочих способов оценивания. Подробный анализ соответствующих оценок, а также их свойств вы можете найти в Rousseeuw, Leroy, Robust Regression and Outlier Detection.

6.2 Взвешенная регрессия

6.2.1 Итеративно взвешиваемая линейная регрессия

Пример 1. Рассмотрим задачу LAD регрессии:

$$\sum_{i=1}^n |y_i - ax_i - b| \rightarrow \min.$$

Его можно рассматривать как взвешенную МНК-регрессию с коэффициентами

$$w_i = \frac{1}{|y_i - ax_i - b|}.$$

Можем ли мы решить такую задачу, учитывая, что w_i зависят от параметров минимизации. Можно делать это итеративно:

- Возьмем $w_{0,i}$, например, равными 1.
- На n -шаге решим задачу

$$\sum_{i=1}^n w_{n-1,i} (y_i - ax_i - b)^2 \rightarrow \min,$$

найдя из нее a , b , а затем пересчитаем веса

$$w_{n,i} = \frac{1}{|y_i - ax_i - b|}.$$

Спустя достаточное число итераций получим оценку, близкую к требуемой. Такой подход называют итеративно взвешиваемая линейная регрессия (IRLS).

В общем случае аналогичная задача решается в виде

$$\sum_{i=1}^n w(\vec{b})(y_i - g(x_i, \vec{b}))^2 \rightarrow \min$$

для некоторых известных функций w , g . Метод остается в точности тем же

- Возьмем $w_{0,i}$, например, равными 1.
- На n -шаге решим задачу

$$\sum_{i=1}^n w_{n-1,i} (y_i - g(x_i, \vec{b}))^2 \rightarrow \min,$$

найдя из нее \vec{b} , а затем пересчитаем веса

$$w_{n,i} = w(\vec{b}).$$

6.3 Обобщенная линейная модель (GLM)

6.3.1 Экспоненциальное семейство

Распределение, имеющее плотность или дискретное распределение вида

$$f(x; \theta, \varphi) = \exp\left(\frac{x\theta - d(\theta)}{a(\varphi)} + c(x, \varphi)\right)$$

на каком-то множестве, не зависящем от θ , где a, d, c – некоторые функции, называют распределением, принадлежащим экспоненциальному семейству. К таким распределениям относятся, например, бернуlliевское, геометрическое, пуассоновское, гамма (с известным параметром формы), нормальное (с известной дисперсией).

Параметр θ при этом называется натуральным или каноническим параметром, $\sum_{i=1}^n x_i$ является для него достаточной статистикой (при известном φ).

Вопрос 1. Показать что для $\mathcal{N}(\mu, \sigma^2)$ натуральный параметр есть μ , для бернуlliевского $\ln p - \ln(1-p)$, а для пуассоновского $\ln \lambda$.

Вопрос 2. Показать, что математическое ожидание величины с плотностью из экспоненциального семейства есть $d'(\theta)$, дисперсия $d''(\theta)/a(\varphi)$.

Нетрудно видеть, что

$$\mathbf{E}_\theta X_1 = \mu = d'(\theta), \quad \mathbf{D}_\theta X_1 = d''(\theta)a(\varphi).$$

6.3.2 Задача регрессии для GLM модели

Общая задача регрессии заключается в поиске функции h , для которой

$$\mathbf{E}(Y|X = \vec{x}) = h(\vec{x}).$$

Для задания модели мы должны знать три компоненты:

1. Случайная компонента $f(y; \theta)$: условное распределение y при заданном θ является заданным распределением, принадлежащем экспоненциальному семейству;
2. Систематическая компонента X : параметр $\eta = X\vec{b}$, где X – матрица предикторов, \vec{b} – неизвестный вектор параметров;
3. Функция связи $\eta = g(\mu)$, где $\mu = \mu(\theta, \varphi)$ – математическое ожидание Y при заданных θ, φ .

Иначе говоря, мы предполагаем, что μ известным нам образом связан с линейной функцией предикторов с неизвестными коэффициентами.

Пример 2. Для $g(x) = x$ и $X_i \sim \mathcal{N}(\theta, \sigma^2)$ получаем

$$h(x_{i,1}, \dots, x_{i,m}) = x_{i,1}b_1 + x_{i,2}b_2 + \dots + x_{i,m}b_m, \quad y_i = h(x_{i,1}, \dots, x_{i,m}) + \varepsilon_i,$$

где $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$. Это привычная нам линейная модель. Однако, для другого выбора g мы получим

$$y_i = g^{-1}(x_{i,1}b_1 + x_{i,2}b_2 + \dots + x_{i,m}b_m) + \varepsilon_i.$$

Это даже не преобразование зависимой переменной, а именно преобразования вида зависимости из линейного в функции от линейного.

Далее мы можем решать задачу максимизации правдоподобия

$$\prod_{i=1}^n f(y_i; g^{-1}((X\vec{b})_i)).$$

по \vec{b} .

Канонической функцией связи будем называть $g(\mu) = \theta$. Таким образом, каноническая регрессия предлагает считать, что натуральный параметр θ выражается как $X\vec{b}$. Для описанного выше примера с нормальным распределением мы получим обычную линейную регрессию. Также нам понадобится $V(\mu)$ – функция дисперсии, которая задается соотношением $V(\mu) = -d''(g(\mu))$. Это часть $\mathbf{D}_{\theta,\varphi}X_i$, зависящая от θ .

Пример 3. Для пуассоновского распределения каноническая функция связи даст нам $\lambda_i = \exp(x_{i,1}b_1 + x_{i,2}b_2 + \dots + x_{i,m}b_m)$, $Y_i \sim Poiss(\lambda_i)$.

Для бернульевского

$$p_i = h(x_{i,1}b_1 + x_{i,2}b_2 + \dots + x_{i,m}b_m), \quad h(x) = g^{-1}(x) = \frac{e^x}{1 + e^x}$$

Функцию $h(x)$ называют логистической, а полученную модель — логистической регрессией.

	$g(x)$	$V(x)$
Нормальное	$g(x) = x$	1,
Биномиальное m	$\ln(x/(1-x))$	$mx(1-x)$
Пуассоновское	$\ln x$	x
Гамма	$1/x$	x^2 .

6.3.3 ОМП-оценивание в канонической GLM модели

При построении ОМП мы получаем задачу

$$\sum_{i=1}^n \left(\frac{y_i \langle X_i, \vec{b} \rangle - d(\langle X_i, \vec{b} \rangle)}{a(\varphi)} + c(y_i, \varphi) \right) \rightarrow \max.$$

Эта задача не решается аналитически, однако, может быть решена численно. Продифференцируем имеющееся выражение по b_j и получим

$$u_j(\vec{b}) = \sum_{i=1}^n \left(y_i - d'(\langle X_i, \vec{b} \rangle) \right) X_{i,j} = 0$$

при всех j . Иначе говоря,

$$0 = u(\vec{b}) = X^T(\vec{Y} - \vec{g}(\vec{b})), \quad g_i(\vec{b}) = d'(\langle X_i, \vec{b} \rangle).$$

Для решения уравнения $u(\vec{b}) = 0$ мы скажем, что

$$0 = u(\vec{b}) \approx u(\vec{b}_0) + H(\vec{b}_0) \cdot (\vec{b} - \vec{b}_0),$$

где $H(\vec{b}_0) = (h_{j,l})$ – матрица Гессе правдоподобия $\ln L$, то есть

$$h_{j,l} = \partial u_j(\vec{b}) / \partial b_l = \sum_{i=1}^n d''(\langle X_i, \vec{b}_0 \rangle) X_{i,j} X_{i,l}.$$

Это так называемый алгоритм Ньютона-Рафсона. Таким образом, мы можем итеративно пересчитывать

$$\vec{b}_{n+1} = \vec{b}_n - H^{-1}(\vec{b}_n) \cdot u(\vec{b}_n),$$

начиная с некоторого \vec{b}_0 . При этом $H = X^T W X$, где W – диагональная матрица с

$$w_{i,i} = -d''(\langle X_i, \vec{b}_n \rangle).$$

Следовательно,

$$\vec{b}_{n+1} = H^{-1}(\vec{b}_n) \cdot (H(\vec{b}_n) \cdot \vec{b}_n + X^T(\vec{Y} - \vec{g}(\vec{b}))) = H^{-1}(\vec{b}_n) \cdot X^T W (X \vec{b}_n + W^{-1}(\vec{Y} - \vec{g}(\vec{b}))).$$

Полагая

$$\vec{Z} = X \vec{b}_n + W^{-1}(\vec{Y} - \vec{g}(\vec{b})),$$

получаем

$$\vec{b}_{n+1} = (X^T W X)^{-1} X^T W \vec{Z},$$

что соответствует решению задачи взвешенной МНК. Таким образом, на шаге мы решаем взвешенную МНК задачу, а в общем осуществляем IRLS для задачи

$$(\vec{Z} - X \vec{b})^T W (\vec{Z} - X \vec{b}).$$

В связи с этим задача оценивания в GLM-модели сводится к IRLS.

6.3.4 Бинарная зависимая переменная

Предположим, что величины Y_i принимают только два значения 0 и 1, причем $\mathbf{P}(Y_i = 1|X_1, \dots, X_n)$ имеет вид

$$p_i = \mathbf{P}(Y_i = 1).$$

Тогда каноническая модель предлагает рассматривать уравнение

$$\ln \frac{p_i}{1 - p_i} = \sum_{j=1}^k b_j X_{i,j}$$

или логарифм отношения среднего числа 1 к среднему числу 0 линейно зависит от значений признаков. Такая регрессия называется логистической.

Для логистической регрессии подсчитывая правдоподобие

$$L(y_1, \dots, y_n; b_1, \dots, b_k) = \prod_{i=1}^n p_i(b_1, \dots, b_k)^{Y_i} (1 - p_i(b_1, \dots, b_k))^{1-Y_i}$$

и находя ОМП $\hat{b}_1, \dots, \hat{b}_k$, мы получим коэффициенты логистической регрессии. Задача опять же сводится к IRLS.

В Python это осуществляется в `sklearn.linear_model` с `LogisticRegression`. Опять же в этом случае есть `predict` для прогнозирования самих значений и `predict_proba` для прогнозирования вероятностей.

6.3.5 Пуассоновская регрессия

В случае, когда Y имеют пуассоновское распределение, каноническая функция связи даст нам

$$\lambda = \exp(\langle x, \vec{b} \rangle).$$

Таким образом, рассчитывая правдоподобие

$$L(y_1, \dots, y_n, \beta_1, \dots, \beta_k) = \exp(-n \exp(\langle x, \vec{b} \rangle)) \prod_{i=1}^n \frac{1}{y_i!} \exp \left(y_i \sum_{j=1}^k \beta_j x_{i,j} \right)$$

и максимизируя его по β_1, \dots, β_k , получаем пуассоновскую регрессию.

В Python эта регрессия сделана в `linear_model.PoissonRegressor`.

Также в Python есть, в частности, гамма-регрессия.

6.3.6 Остатки в GLM-модели

Естественным образом, остатки в общей модели представляют собой $r_i = y_i - \hat{\mu} = y_i - g^{-1}(X\hat{b}_i)$. Обычно рассматривают стандартизированные остатки.

Одним из вариантов являются пирсоновские остатки

$$r_{P,i} = \frac{y_i - \hat{\mu}}{\sqrt{V(\hat{\mu})}},$$

где $V(\mu) = d''(\theta)$. Эти величины зависимы, но имеют нулевое среднее и единичную дисперсию. Эти остатки называют пирсоновскими, потому что сумма их квадратов представляет собой статистику хи-квадрат для проверки гипотезы о том, что y действительно принадлежат нужной параметрической модели.

Другим популярным вариантом являются deviance остатки, заданные формулой

$$r_{D,i} = \text{sign}(y_i - \hat{\mu}) \sqrt{dev_i}.$$

Чтобы объяснить, что такое dev_i , найдем с того, что введем deviance

$$D = \sum_{i=1}^n dev_i.$$

Представим себе, что мы рассмотрели так называемую насыщенную модель, в которой каждое наблюдение имеет свой параметр θ , тогда максимум правдоподобия будет равен

$$\ln L(y_1, \dots, y_n, \tilde{\theta}_i) = \sum_{i=1}^n \frac{y_i \tilde{\theta}_i - d(\tilde{\theta}_i)}{a(\varphi)} + \sum_{i=1}^n c(y_i, \varphi),$$

где $\tilde{\theta}_i$ – ОМП для данной модели. Дифференцируя, мы находим $\tilde{\theta}_i$ из условий

$$x_i = d'(\tilde{\theta}_i),$$

если система имеет решение. При этом в нашей ненасыщенной модели максимум правдоподобия равен

$$\ln L(y_1, \dots, y_n, \theta_i) = \sum_{i=1}^n \frac{y_i \hat{\theta}_i - d(\hat{\theta}_i)}{a(\varphi)} + \sum_{i=1}^n c(y_i, \varphi),$$

где $\hat{\theta} = g(\hat{\mu})$ – ОМП в нашей модели. Следовательно, статистика к.о.о.п. имеет вид

$$2 \ln \frac{L(y_1, \dots, y_n, \tilde{\theta}_i)}{L(y_1, \dots, y_n, \hat{\theta}_i)} = 2 \sum_{i=1}^n \frac{y_i (\tilde{\theta}_i - \hat{\theta}_i) + d(\hat{\theta}_i) - d(\tilde{\theta}_i)}{a(\varphi)}.$$

Эта величина пропорциональна так называемой deviance

$$D = \sum dev_i, \quad dev_i = V(\hat{\mu}_i)(y_i(\tilde{\theta}_i - \hat{\theta}_i) + d(\hat{\theta}_i) - d(\tilde{\theta}_i))^2.$$

Приведем таблицу формул D для основных моделей

	D
Нормальное	$\sum (y_i - \hat{\mu}_i)^2$
Биномиальное	$2 \sum (y_i \ln(y_i/\hat{\mu}_i) + (m - y_i) \ln((m - y_i)/(m - \hat{\mu}_i)))$
Пуассоновское	$2 \sum (y_i \ln(y_i/\hat{\mu}_i) - y_i + \hat{\mu})$
Гамма	$2 \sum (-\ln(y_i/\hat{\mu}_i) + (y_i - \hat{\mu}_i)/\hat{\mu}_i)$.

Можно построить графики пирсоновских или deviance остатков в зависимости от $X\hat{\beta}$ или $\hat{\mu}$. Анализ графиков остатков несколько сложнее чем в линейной модели, но при верной модели пирсоновские остатки должны иметь нулевое среднее и единичную дисперсию. Если мы видим значительное отклонение от этого, то модель может быть неадекватной.

нение от этой модели – стоит задуматься об ее адекватности.

Можно ввести и преобразование Бокса-Кокса для предикторов и/или зависимой переменной. При этом к предикторам их применяют его в обычном формате, а к зависимой переменной добавляют ее в виде:

$$\theta = g_\lambda(\vec{X}\vec{b}),$$

где g_λ преобразование Бокса-Кокса (или Йео-Джонсона). После этого нам нужно заново осуществить поиск ОМП.