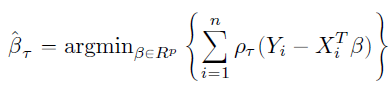
Если есть выборка размера n, и классическая матрица признаки-ответы (X-Y), то задача квантильной регрессии для квантиля τ записывается как



Оценки коэффициентов β вычисляются по формуле



где  - функция потерь квантильной регрессии

Развитие методов регуляризации в обычной регрессии логичным образом перешло и в квантильную регрессию. Постановка задачи регуляризации в квантильной регрессии такая же, как и в обычной:



Где L(f) – функция потерь, P(f) – штрафная функция, λ – коэффициент

Более полно эту задачу можно расписать как

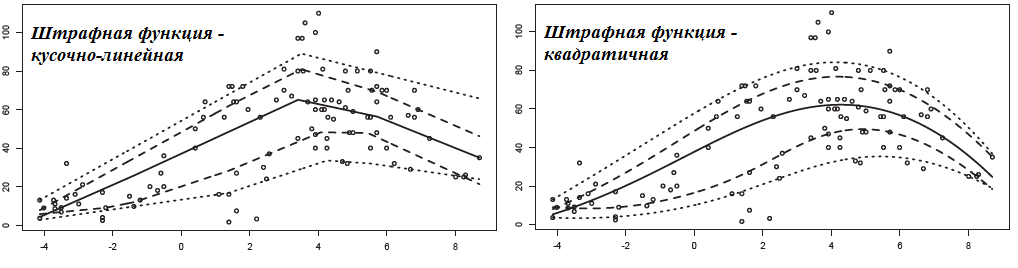


где Ω = [a;b] – интервал, охватывающий все точки, f(x) – функция регрессии

Решение этого функционала сводится к задаче квадратичного программирования. В 1994 году было показано, что можно убрать квадрат из штрафной функции, заменив ее задачей



Разница между получающимися функциями представлена на рисунке ниже



Цитируется по: Koenker. R., Chernozhukov, V., He, X., Peng, L. Handbook of Quantile Regression. CRC Press. 2018. ISBN 978-1-4987-2528-6, с.47

Фактически, последняя формула повторяет постановку задачи лассо-регрессии, в которой берется модуль от некоторого оценщика. Еще одно ее преимущество – вычислительная простота, поскольку конкретные значения коэффициентов можно получить в результате решения задачи линейного программирования.

Последующее развитие постановки задачи привело ее к виду



где степень производной k предполагалось выбирать в зависимости от свойств данных:

- если функция f(x) предполагает наличие мгновенных скачков, то выбирается k=1

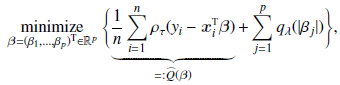
- если функция f(x) непрерывна, но имеет точки перегиба, то k=2

- если функция f(x) непрерывна и дифференцируема, то k=3

Далее усилия теоретиков были направлены на определение оптимального значения параметра λ. Как правило, для этого используется алгоритм перекрестной проверки (кросс-валидация с разбиением на несколько групп)

В сентябре 2021 года коллектив китайских ученых выпустил припринт статьи (https://arxiv.org/abs/2109.05640), в котором предложил способ борьбы с систематическими ошибками, которые дает лассо-регуляризация в квантильной регрессия.

Постановка задачи нахождения коэффициентов квантильной регрессии записывается теперь так:



где qλ – некоторая функция от λ

Кроме того, первое слагаемое функции потерь заменяется ими на ядро K(∙), и предлагается общий итеративный алгоритм нахождения коэффициентов квантильной регрессии. Именно он и используется в новом пакете conquer

Для расчетов возьмем базу данных Schooling из пакета Ecdat. В ней представлены данные о зарплате американских рабочих. Опрос был проведен в 1976 году.

В качестве зависимой переменной выберем заработную плату, независимых – количество лет образования, возраст, результаты теста iq и опыт работы.

Простую модель квантильной регрессии можно построить с помощью команды conquer(), которая содержит следующие параметры:

X, Y – матрица независимых и вектор зависимого признаков

tau – квантиль (при tau = 0.5 получается медианная регрессия)

kernel - определяет, какое ядро сглаживания K(∙) используется (выбор из следующий значений: "Gaussian", "logistic", "uniform", "parabolic", "triangular")

ci – параметр, определяющий необходимость расчета доверительных интервалов коэффициентов регрессии

h, checkSing, tol, iteMax, alpha – параметры алгоритма расчета

library(Ecdat)

library(tidyverse)

library(conquer)

Base1<-na.omit(Ecdat::Schooling)

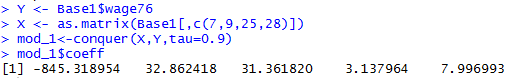
glimpse(Base1)

Y <- Base1$wage76

X <- as.matrix(Base1[,c(7,9,25,28)])

mod\_1<-conquer(X,Y,tau=0.9)

mod\_1$coeff



В итоге выводятся коэффициенты регрессии в порядке следования параметров, первое число – значение свободного члена

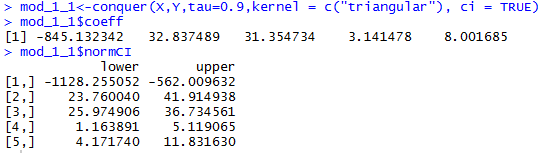
Также по результатам расчетов можно вывести:

ite - Число итераций до сходимости.

residual - Вектор полученных моделью значений

perCI, pivCI, normCI – Доверительные интервалы коэффициентов регрессии

Посчитаем модель с другим ядром и выведем доверительные интервалы



Как видно, коэффициенты, во-первых, отличаются не сильно, и, во-вторых, относительно устойчивы (доверительные интервалы не включают 0)

Функция conquer.cv.reg – позволяет реализовать квантильную регрессию с применением методов регуляризации (за это отвечает параметр penalty, которое может принимать одно из трех значений: "lasso", "scad", "mcp") и кросс-валидации (параметр kfolds отвечает за число блоков). Посчитаем квантильную регрессию с лассо-регуляризацией (придется подождать, считает дольше)



Как видно, регуляризация посчитала, что последнюю переменную можно не учитывать.

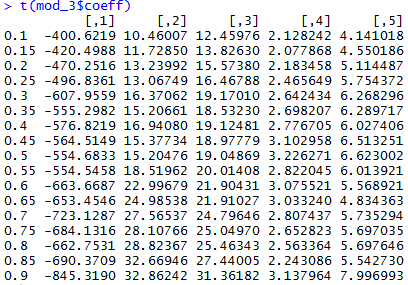
Также крайне удобна функция conquer.process. Она позволяет посчитать сразу целый блок квантильных регрессий для разных значений квантиля:

*tau\_t <- seq(0.1, 0.9, by = 0.05)*

*mod\_3 <-conquer.process(X,Y,tauSeq = tau\_t)*

*colnames(mod\_3$coeff) <- tau\_t*

*t(mod\_3$coeff)*



Собственно, это показывает, что для разных квантилей коэффициенты регрессии могут сильно отличаться

Кроме того, с небольшой модификацией можно использовать эти функции для отбора признаков. Посмотрим, как будут меняться коэффициенты функции при разных значениях λ

*lambda\_v <- 10^(seq(-4, -1, by = 0.1))*

*Res <- matrix(0, nrow = 31, ncol = 5)*

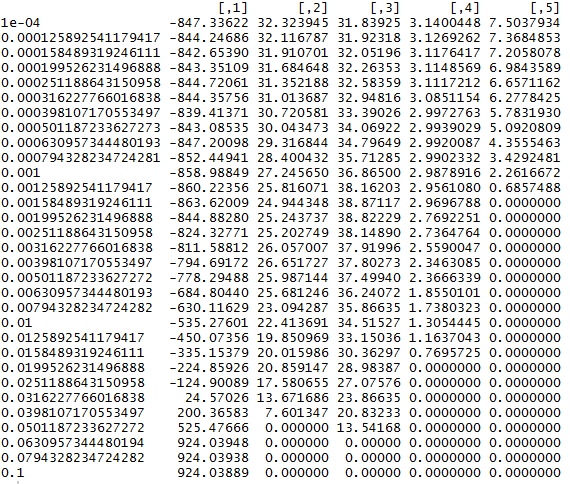
*for(i in 1:31) {*

*mod\_4 <- conquer.reg(X,Y,lambda = lambda\_v[i],tau = 0.9, penalty = c("lasso"))*

*Res[i,]<-mod\_4$coeff}*

*rownames(Res)<-lambda\_v*

*Res*



Эту динамику значений вполне можно использовать как характеристику важности переменных. Таким образом, ранжирование переменных по значимости будет выглядеть так: возраст –> количество лет учебы -> результаты теста iq -> опыт работы.