**Quantile regression: what is it?**

Let y \in \mathbb{R}be some response variable of interest, and let x \in \mathbb{R}^pbe a vector of features or predictors that we want to use to model the response. In linear regression, we are trying to estimate the *conditional mean function*, \mathbb{E}[y \mid x], by a linear combination of the features.

While the conditional mean function is often what we want to model, sometimes we may want to model something else. On a [recent episode of the Linear Digressions podcast](http://lineardigressions.com/episodes/2019/1/12/quantile-regression), Katie and Ben talked about a [situation in Uber](https://eng.uber.com/analyzing-experiment-outcomes/) where it might make sense to model a ***conditional quantile function***.

Let’s make this more concrete. Say Uber came up with a new algorithm for dispatching drivers and we are interested in how this algorithm fares in terms of wait times for consumers. A simple (linear regression) model for this is

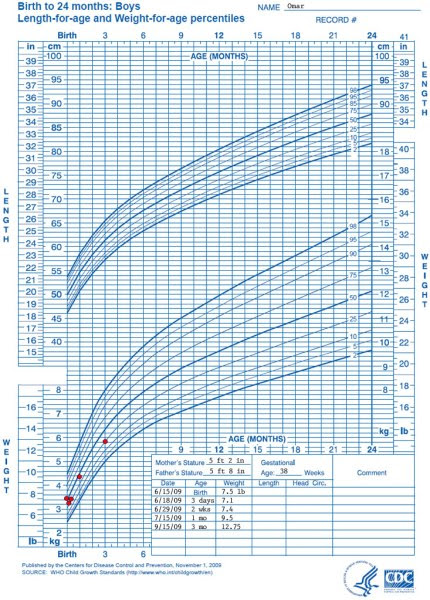
\mathbb{E}[wait\_time \mid algorithm] = a + b \cdot algorithm,

where algorithm = 1if the new algorithm was used to dispatch a driver, and algorithm = 0if the previous algorithm was used. From this model, we can say that under the old algorithm, mean wait time was a, but under the new algorithm, mean wait time is a + b. So if b < 0, I would infer that my new algorithm is "doing better".

But is it really? What if the new algorithm improves wait times for 90% of customers by 1 min, but makes the wait times for the remaining 10% longer by 5 min? Overall, I would see a decrease in mean wait time, but things got significantly worse for a segment of my population. What if that 10% whose wait times became 5 minutes longer were already having the longest wait times to begin with? That seems like a bad situation to have, but our earlier model would not pick it up.

One way to pick up such situations is to model ***conditional quantile functions*** instead. That is, trying to estimate the mean of ygiven the features x, let’s trying to estimate a quantile of ygiven the features x. In our example above, instead of trying to estimate the mean wait time, we could estimate the 95th quantile wait time to catch anything going wrong out in the tails of the distribution.

Another example where estimating conditional quantiles is useful is in growth charts. All of you have probably seen one of these charts below in a doctor’s office before. Each line in the growth chart represents some quantile for length/weight given the person’s age. We track our children’s length and weight on this chart to see if they are growing normally or not.



WHO growth chart for boys. The lines represent conditional quantile functions for different quantiles: of length given age and weight given age.

**Quantile regression** is a regression method for estimating these conditional quantile functions. Just as linear regression estimates the conditional mean function as a linear combination of the predictors, quantile regression estimates the conditional quantile function as a linear combination of the predictors.

**Quantile regression in R**

We can perform quantile regression in R easily with the [quantreg](https://cran.r-project.org/package=quantreg) package. I will demonstrate how to use it on the mtcars dataset. (For more details on the quantreg package, you can read the package’s vignette [here](https://cran.r-project.org/web/packages/quantreg/vignettes/rq.pdf).)

Let’s load our packages and data:

library(quantreg)

data(mtcars)

We can perform quantile regression using the rq function. We can specify a tau option which tells rq which conditional quantile we want. The default value for tau is 0.5 which corresponds to ***median regression***. Below, we fit a quantile regression of miles per gallon vs. car weight:

rqfit <- rq(mpg ~ wt, data = mtcars)

rqfit

# Call:

# rq(formula = mpg ~ wt, data = mtcars)

#

# Coefficients:

# (Intercept) wt

# 34.232237 -4.539474

#

# Degrees of freedom: 32 total; 30 residual

Printing the fitted object to the console gives some rudimentary information on the regression fit. We can use the summary function to get more information (just like we do for the lm function):

summary(rqfit)

# Call: rq(formula = mpg ~ wt, data = mtcars)

#

# tau: [1] 0.5

#

# Coefficients:

# coefficients lower bd upper bd

# (Intercept) 34.23224 32.25029 39.74085

# wt -4.53947 -6.47553 -4.16390

In the table above, the lower bd and upper bd columns represent the endpoints of confidence intervals for the model coefficients. There are a number of ways for these confidence intervals to be computed; this can be specified using the seoption when invoking the summary function. The default value is se="rank", with the other options being “iid”, “nid”, “ker”, “boot” and “BLB” (type ?summary.rq for details).

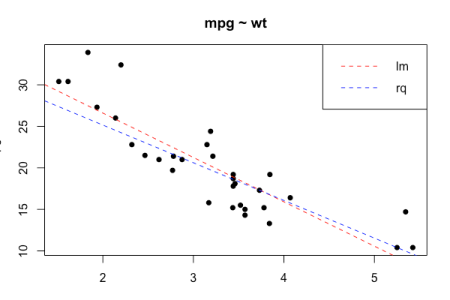
This next block of code plots the quantile regression line in blue and the linear regression line in red:

plot(mpg ~ wt, data = mtcars, pch = 16, main = "mpg ~ wt")

abline(lm(mpg ~ wt, data = mtcars), col = "red", lty = 2)

abline(rq(mpg ~ wt, data = mtcars), col = "blue", lty = 2)

legend("topright", legend = c("lm", "rq"), col = c("red", "blue"), lty = 2)



Median regression (i.e. 50th quantile regression) is sometimes preferred to linear regression because it is “robust to outliers”. The next plot illustrates this. We add two outliers to the data (colored in orange) and see how it affects our regressions. The dotted lines are the fits for the original data, while the solid lines are for the data with outliers. As before, red is for linear regression while blue is for quantile regression. See how the linear regression fit shifts a fair amount compared to the median regression fit (which barely moves!)?

y <- c(mtcars$mpg, 40, 36)

x <- c(mtcars$wt, 5, 4)

plot(y ~ x, pch = 16, main = "mpg ~ wt")

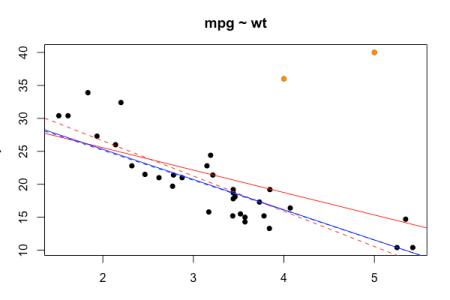
points(c(5, 4), c(40, 36), pch = 16, col = "dark orange")

abline(lm(mpg ~ wt, data = mtcars), col = "red", lty = 2)

abline(lm(y ~ x), col = "red")

abline(rq(mpg ~ wt, data = mtcars), col = "blue", lty = 2)

abline(rq(y ~ x), col = "blue")



As I mentioned before, the tau option tells rq which conditional quantile we want. What is interesting is that we can set tau to be a vector and rq will give us the fits for all those quantiles:

multi\_rqfit <- rq(mpg ~ wt, data = mtcars, tau = seq(0, 1, by = 0.1))

multi\_rqfit

# Call:

# rq(formula = mpg ~ wt, tau = seq(0, 1, by = 0.1), data = mtcars)

#

# Coefficients:

# tau= 0.0 tau= 0.1 tau= 0.2 tau= 0.3 tau= 0.4 tau= 0.5 tau= 0.6 tau= 0.7 tau= 0.8 tau= 0.9

# (Intercept) 27.6875 37.509794 37.2768 34.876712 34.891176 34.232237 36.734405 38.497404 38.879916 44.391047

# wt -3.7500 -6.494845 -6.2400 -5.205479 -4.852941 -4.539474 -5.016077 -5.351887 -5.250722 -6.266786

# tau= 1.0

# (Intercept) 44.781558

# wt -5.627981

#

# Degrees of freedom: 32 total; 30 residual

I don’t think there’s a clean one-liner to plot all the quantile fits at once. The following loop is one possible solution:

# plotting different quantiles

colors <- c("#ffe6e6", "#ffcccc", "#ff9999", "#ff6666", "#ff3333",

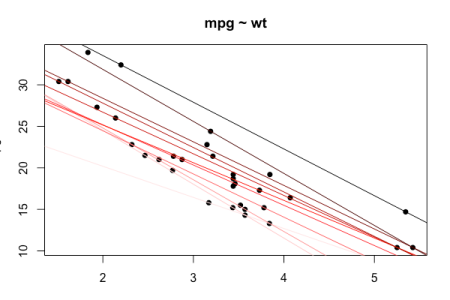
"#ff0000", "#cc0000", "#b30000", "#800000", "#4d0000", "#000000")

plot(mpg ~ wt, data = mtcars, pch = 16, main = "mpg ~ wt")

for (j in 1:ncol(multi\_rqfit$coefficients)) {

abline(coef(multi\_rqfit)[, j], col = colors[j])

}



rq exhibits interesting behavior if tau is set to some value outside of [0,1]: it gives solutions for ALL values of tau in [0,1]! (It turns out that there are a finite number of them.)

One final note: rq also supports more complicated formula.

# no intercept

rq(mpg ~ wt - 1, data = mtcars)

# Call:

# rq(formula = mpg ~ wt - 1, data = mtcars)

#

# Coefficients:

# wt

# 4.993498

#

# Degrees of freedom: 32 total; 31 residual

# additive model

rq(mpg ~ wt + cyl, data = mtcars)

# Call:

# rq(formula = mpg ~ wt + cyl, data = mtcars)

#

# Coefficients:

# (Intercept) wt cyl

# 38.871429 -2.678571 -1.742857

#

# Degrees of freedom: 32 total; 29 residual

# model with interactions

rq(mpg ~ wt \* cyl, data = mtcars)

# Call:

# rq(formula = mpg ~ wt \* cyl, data = mtcars)

#

# Coefficients:

# (Intercept) wt cyl wt:cyl

# 51.6650791 -7.2496674 -3.4759342 0.5960682

#

# Degrees of freedom: 32 total; 28 residual

# fit on all other predictors

rq(mpg ~ ., data = mtcars)

# Call:

# rq(formula = mpg ~ ., data = mtcars)

#

# Coefficients:

# (Intercept) cyl disp hp drat wt qsec vs am

# 7.61900909 0.64658010 0.02330302 -0.03149324 0.78058219 -4.56231735 0.57610198 1.58875367 1.33175736

# gear carb

# 2.37453015 -0.33136465

#

# Degrees of freedom: 32 total; 21 residual

References:

1. University of Virginia Library. [Getting started with quantile regression](https://data.library.virginia.edu/getting-started-with-quantile-regression/).
2. Wikipedia. [Quantile regression](https://en.wikipedia.org/wiki/Quantile_regression).