After my [series of post](https://freakonometrics.hypotheses.org/category/r/from-scatch) on classification algorithms, it’s time to get back to R codes, this time for quantile regression. Yes, I still want to get a better understanding of optimization routines, in R. Before looking at the quantile regression, let us compute the median, or the quantile, from a sample.

**Median**

Consider a sample . To compute the median, solvewhich can be solved using linear programming techniques. More precisely, this problem is equivalent towith and , .  
To illustrate, consider a sample from a lognormal distribution,

|  |  |
| --- | --- |
| 1  2  3  4  5 | n = 101  **set.seed**(1)  y = **rlnorm**(n)  **median**(y)  [1] 1.077415 |

For the optimization problem, use the matrix form, with constraints, and parameters,

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | **library**(lpSolve)  A1 = **cbind**(**diag**(2\*n),0)  A2 = **cbind**(**diag**(n), -**diag**(n), 1)  r = lp("min", **c**(**rep**(1,2\*n),0),  **rbind**(A1, A2),**c**(**rep**("&gt;=", 2\*n), **rep**("=", n)), **c**(**rep**(0,2\*n), y))  **tail**(r$solution,1)  [1] 1.077415 |

It looks like it’s working well…

**Quantile**

Of course, we can adapt our previous code for quantiles

|  |  |
| --- | --- |
| 1  2  3  4 | tau = .3  **quantile**(x,tau)  30%  0.6741586 |

The linear program is nowwith and , . The R code is now

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | A1 = **cbind**(**diag**(2\*n),0)  A2 = **cbind**(**diag**(n), -**diag**(n), 1)  r = lp("min", **c**(**rep**(tau,n),**rep**(1-tau,n),0),  **rbind**(A1, A2),**c**(**rep**("&gt;=", 2\*n), **rep**("=", n)), **c**(**rep**(0,2\*n), y))  **tail**(r$solution,1)  [1] 0.6741586 |

So far so good…

**Quantile Regression (simple)**

Consider the following dataset, with rents of flat, in a major German city, as function of the surface, the year of construction, etc.

|  |  |
| --- | --- |
| 1 | base=**read.table**("http://freakonometrics.free.fr/rent98\_00.txt",header=TRUE) |

The linear program for the quantile regression is nowwith and . So use here

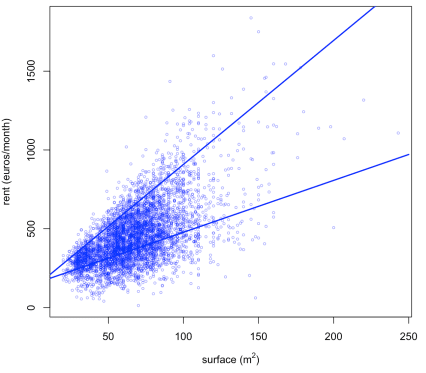
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | **require**(lpSolve)  tau = .3  n=**nrow**(base)  X = **cbind**( 1, base$area)  y = base$rent\_euro  A1 = **cbind**(**diag**(2\*n), 0,0)  A2 = **cbind**(**diag**(n), -**diag**(n), X)  r = lp("min",  **c**(**rep**(tau,n), **rep**(1-tau,n),0,0), **rbind**(A1, A2),  **c**(**rep**("&gt;=", 2\*n), **rep**("=", n)), **c**(**rep**(0,2\*n), y))  **tail**(r$solution,2)  [1] 148.946864 3.289674 |

Of course, we can use R function to fit that model

|  |  |
| --- | --- |
| 1  2  3  4  5 | **library**(quantreg)  rq(rent\_euro~area, tau=tau, **data**=base)  Coefficients:  (Intercept) area  148.946864 3.289674 |

Here again, it seems to work quite well. We can use a different probability level, of course, and get a plot

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | **plot**(base$area,base$rent\_euro,xlab=**expression**(**paste**("surface (",m^2,")")),  ylab="rent (euros/month)",**col**=**rgb**(0,0,1,.4),cex=.5)  sf=0:250  yr=r$solution[2\*n+1]+r$solution[2\*n+2]\*sf  **lines**(sf,yr,lwd=2,**col**="blue")  tau = .9  r = lp("min",  **c**(**rep**(tau,n), **rep**(1-tau,n),0,0), **rbind**(A1, A2),  **c**(**rep**("&gt;=", 2\*n), **rep**("=", n)), **c**(**rep**(0,2\*n), y))  **tail**(r$solution,2)  [1] 121.815505 7.865536  yr=r$solution[2\*n+1]+r$solution[2\*n+2]\*sf  **lines**(sf,yr,lwd=2,**col**="blue") |



**Quantile Regression (multiple)**

Now that we understand how to run the optimization program with one covariate, why not try with two ? For instance, let us see if we can explain the rent of a flat as a (linear) function of the surface and the age of the building.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | **require**(lpSolve)  tau = .3  n=**nrow**(base)  X = **cbind**( 1, base$area, base$yearc )  y = base$rent\_euro  A1 = **cbind**(**diag**(2\*n), 0,0,0)  A2 = **cbind**(**diag**(n), -**diag**(n), X)  r = lp("min",  **c**(**rep**(tau,n), **rep**(1-tau,n),0,0,0), **rbind**(A1, A2),  **c**(**rep**("&gt;=", 2\*n), **rep**("=", n)), **c**(**rep**(0,2\*n), y))  **tail**(r$solution,3)  [1] 0.000000 3.257562 0.077501 |

Unfortunately, this time, it is not working well…

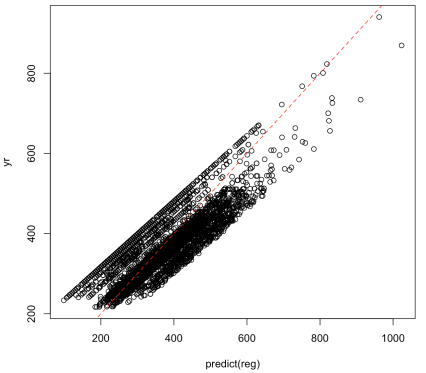
|  |  |
| --- | --- |
| 1  2  3  4  5 | **library**(quantreg)  rq(rent\_euro~area+yearc, tau=tau, **data**=base)  Coefficients:  (Intercept) area yearc  -5542.503252 3.978135 2.887234 |

Results are quite different. And actually, another technique can confirm the later ([IRLS](https://en.wikipedia.org/wiki/Iteratively_reweighted_least_squares) – Iteratively Reweighted Least Squares)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | eps = **residuals**(**lm**(rent\_euro~area+yearc, **data**=base))  **for**(s **in** 1:500){  reg = **lm**(rent\_euro~area+yearc, **data**=base, **weights**=(tau\*(eps&gt;0)+(1-tau)\*(eps&lt;0))/**abs**(eps))  eps = **residuals**(reg)  }  reg$coefficients  (Intercept) area yearc  -5484.443043 3.955134 2.857943 |

I could not figure out what went wrong with the linear program. Not only coefficients are very different, but also predictions…

|  |  |
| --- | --- |
| 1  2  3 | yr = r$solution[2\*n+1]+r$solution[2\*n+2]\*base$area+r$solution[2\*n+3]\*base$yearc  **plot**(**predict**(reg),yr)  **abline**(a=0,b=1,lty=2,**col**="red") |

  
It’s now time to investigate….