

Quantile Regression with quantreg

N. Gennaro - F. Pontarin

28 giugno 2018

A bit of Theory

The dataset

The package

Some Examples

A bit of Theory

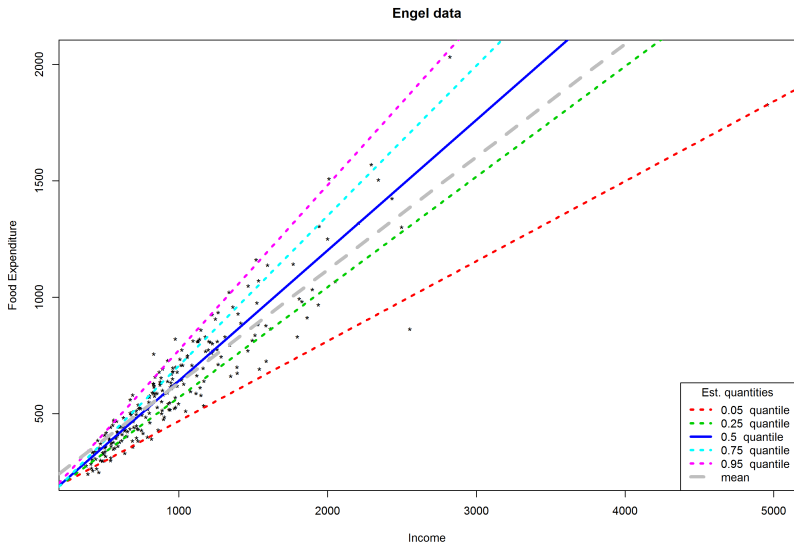
Expanding the usual approach

The objective of linear regression analysis is the estimation of the conditional *mean*

Quantile Regression aims at estimating specific conditional *quantiles* (e.g. median)

It was introduced by Roger Koenker and Gilbert Bassett:
“Regression Quantiles”, *Econometrica*, 46, 33-50 (1978)

Expanding the usual approach



Expanding the usual approach

- ▶ Linear regression is supported by a number of assumptions about the predictor variables, the response variables and their relationship (error independence, homoscedasticity,...)

The structure underlying this type problems allow them to be transformed into linear algebra equivalents

- ▶ On the other hand, Quantile regression makes no particular assumption on the distribution of the response nor about its variance

This leads instead to linear programming problems

Expanding the usual approach

Table 1: A quick comparison between the two regressions

Linear Regression	Quantile Regression
Predicts the mean	Predicts conditional quantiles
Applies when n is small	Needs sufficient data
Often assumes normality	Is distribution agnostic
Is sensitive to outliers	Is robust to response outliers
Is computationally inexpensive	Is computationally intensive

Under the hood

Called τ the objective quantile of a random variable Y , Quantile regression minimizes a sum that gives asymmetric penalties:

- ▶ $(\tau - 1)e_i$ for underestimates
- ▶ τe_i for overestimates

This is condensed in what's defined as the *loss function*

$$\rho_\tau(y) = (\tau - 1_{\{y < 0\}})y$$

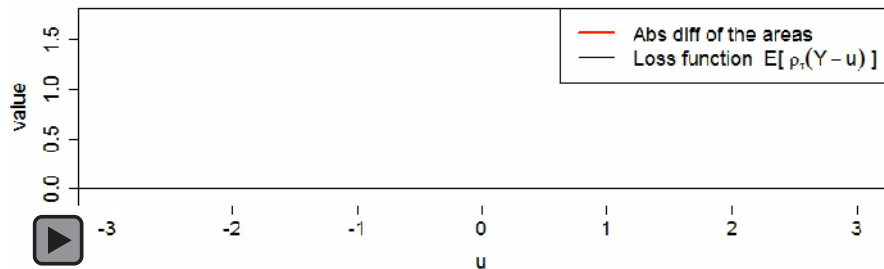
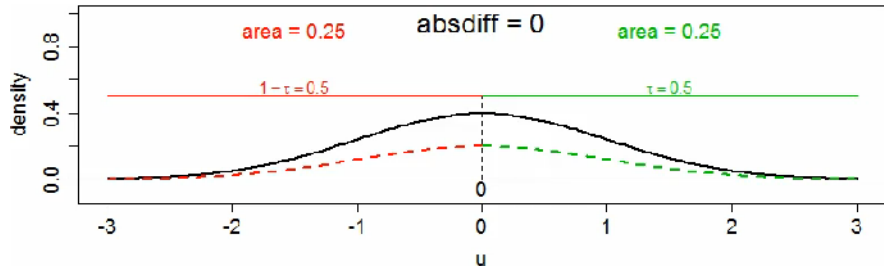
Under the hood

The wanted quantile u can be calculated by minimizing the expected loss for $Y - u$ that is

$$E[\rho_\tau(Y - u)] = (\tau - 1) \int_{-\infty}^u (y - u) dF_Y(y) + \tau \int_u^{+\infty} (y - u) dF_Y(y)$$

The concept is generalized to allow its use with distributions coming from observed samples

A visual hint



The dataset

California Housing Data

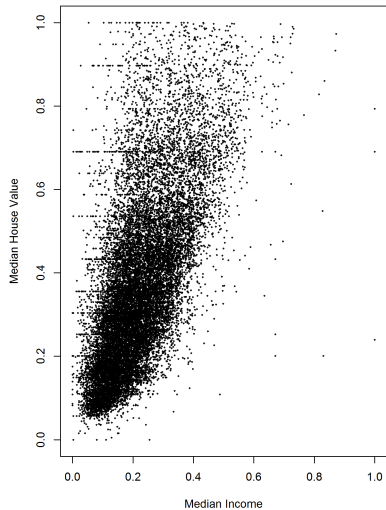
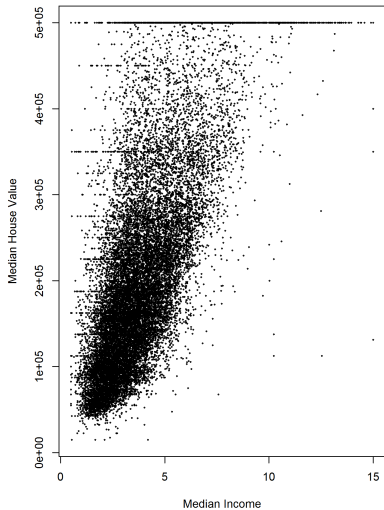
This dataset found on Kaggle gathers information from the census of the Californian districts.

It contains about 20600 observations of many different variables:

```
house <- read.csv("house_from_gitsklearn.csv")
summary(house)
```

```
##      longitude      latitude  housing_median_age  total_rooms
##  Min.   :-124.3    Min.     :32.54    Min.    : 1.00    Min.     : 2
##  1st Qu.: -121.8    1st Qu.:33.93    1st Qu.:18.00    1st Qu.: 1448
##  Median : -118.5    Median :34.26    Median :29.00    Median : 2127
##  Mean   : -119.6    Mean    :35.63    Mean    :28.64    Mean    : 2636
##  3rd Qu.: -118.0    3rd Qu.:37.71    3rd Qu.:37.00    3rd Qu.: 3148
##  Max.    : -114.3    Max.     :41.95    Max.     :52.00    Max.    :39320
##
##  total_bedrooms  population    households    median_income
##  Min.    : 1.0    Min.     : 3    Min.     : 1.0    Min.     : 0.4999
##  1st Qu.: 296.0    1st Qu.: 787    1st Qu.: 280.0    1st Qu.: 2.5634
##  Median : 435.0    Median : 1166    Median : 409.0    Median : 3.5348
##  Mean    : 537.9    Mean    : 1425    Mean    : 499.5    Mean    : 3.8707
##  3rd Qu.: 647.0    3rd Qu.: 1725    3rd Qu.: 605.0    3rd Qu.: 4.7432
##  Max.    :6445.0    Max.     :35682    Max.     :6082.0    Max.    :15.0001
##  NA's     :207
##  median_house_value  ocean_proximity
##  Min.    : 14999    <1H OCEAN :9136
##  1st Qu.:119600    INLAND    :6551
##  Median :179700    ISLAND    : 5
##  Mean    :206856    NEAR BAY  :2290
##  3rd Qu.:264725    NEAR OCEAN:2658
##  Max.    :500001
##
```

Cleaning the dataset



The package

quantreg

The R-package quantreg provides the estimation and inference methods for models of conditional quantiles

It was written by Roger Koenker

```
library(quantreg)
```

Some Examples

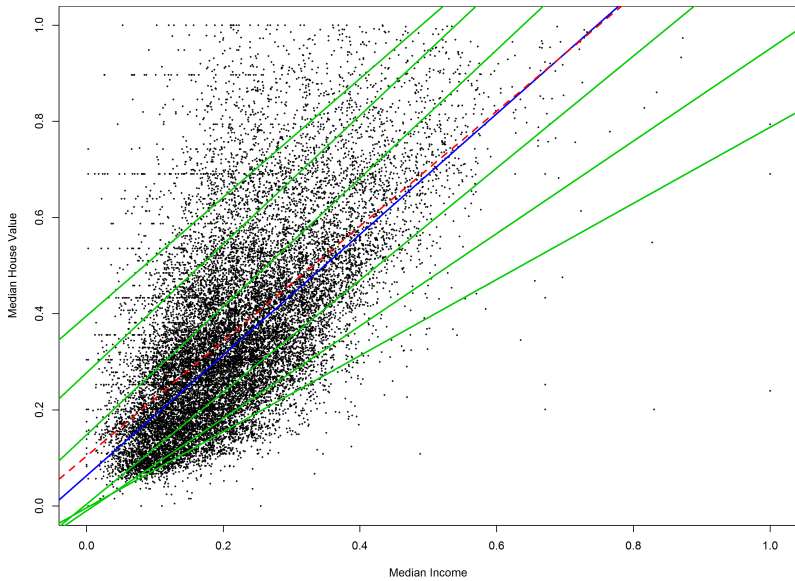
The function rq

to perform a Quantile regression one can call:

```
rq(formula, tau=.5, data, subset, weights, na.action,  
   method="br", model = TRUE, contrasts, ...)
```

So for example:

```
taus <- c(.05,.1,.25,.50,.75,.90,.95)  
for( i in 1:length(taus)){  
  current.rq<- rq( median_house_value ~ median_income,  
                  data=house,tau=taus[i])  
  abline(current.rq)  
}
```



The function `summary.rqs`

With the result given by `rq` one can print the summary with the function `summary.rqs`

```
summary.rqs(object, se = NULL, covariance=FALSE,  
            hs = TRUE, U = NULL, gamma = 0.7, ...)
```

the coefficients for each requested quantiles are shown

```
population.rq<- rq(population ~ households,  
                  data=house, tau=taus)  
population.summary<-summary.rqs(population.rq, ci='boot')  
population.summary
```

```
##  
## Call: rq(formula = population ~ households, tau = taus, data = house)  
##  
## tau: [1] 0.05  
##  
## Coefficients:  
##           Value      Std. Error t value  Pr(>|t|)  
## (Intercept)  0.00065    0.00016   4.18270  0.00003  
## households   0.30683    0.00287  107.02170  0.00000  
##  
...
```

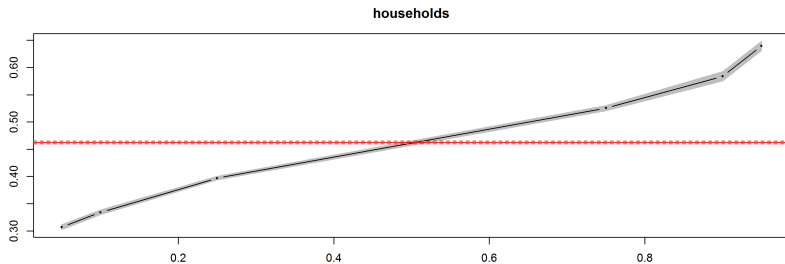
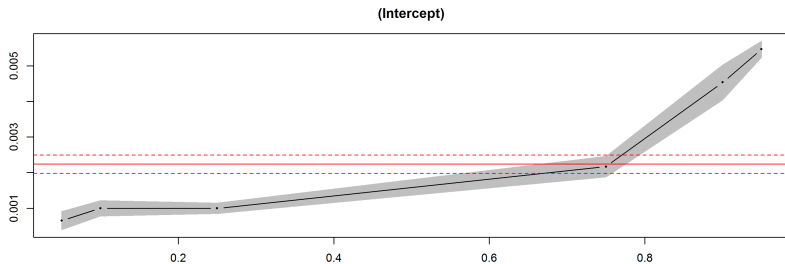
The function `plot.summary.rqs`

One can plot the summary with the following command:

```
plot.summary.rqs(x, parm = NULL, level = 0.9, ols = TRUE,  
  mfrow = NULL, mar = NULL, ylim = NULL, main = NULL,  
  col = gray(c(0, 0.75)), border = NULL, lcol = 2,  
  lty = 1:2, cex = 0.5, pch = 20, type = "b", xlab = "",  
  ylab = "", ...)
```

The coefficients for each requested quantiles are plotted together with their confidence intervals.

```
plot.summary.rqs(population.summary)
```



The function `anova.rq`

It is possible to perform the Anova of `rq` objects.

`anova.rq` comes in two forms:

1. when the fit is performed for a vector of quantiles the function perform a test on the hypothesis that all the coefficients are the same
2. compare nested models as in linear regression

anova.rq compare slopes

```
anova.rqs(fit)
```

```
## Quantile Regression Analysis of Deviance Table
##
## Model: median_house_value ~ median_income
## Joint Test of Equality of Slopes: tau in { 0.05 0.1 0.25 0.5 0.75 0.9 0.95 }
##
##   Df Resid Df F value    Pr(>F)
## 1   6   136319 210.36 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The results show indeed that it is unlikely that the lines have the same slope

anova.rq nested models

```
fit.1 <- rq(median_house_value ~ median_income, tau=0.5, data=house)
fit.2 <- rq(median_house_value ~ median_income + total_rooms,
            tau=0.5, data=house)
fit.3 <- rq(median_house_value ~ median_income + total_bedrooms,
            tau=0.5, data=house)

anova.rq(fit.1, fit.2)
```

```
## Quantile Regression Analysis of Deviance Table
##
## Model 1: median_house_value ~ median_income + total_rooms
## Model 2: median_house_value ~ median_income
##   Df Resid Df F value Pr(>F)
## 1 1      19472  1.5544 0.2125
```

```
anova.rq(fit.1, fit.3)
```

```
## Quantile Regression Analysis of Deviance Table
##
## Model 1: median_house_value ~ median_income + total_bedrooms
## Model 2: median_house_value ~ median_income
##   Df Resid Df F value      Pr(>F)
## 1 1      19472  49.759 1.797e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The results shows that the additional covariate `total_rooms` is not significant while the `total_bedrooms` covariate does improve significantly the prediction

The function `predict.rq`

quantreg gives also the possibility to predict the quantiles given the fit

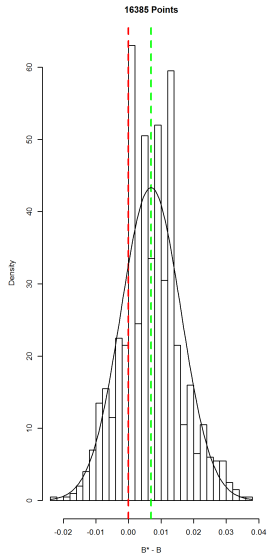
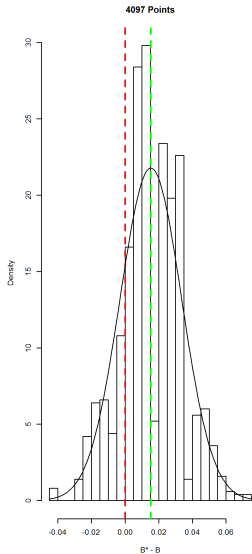
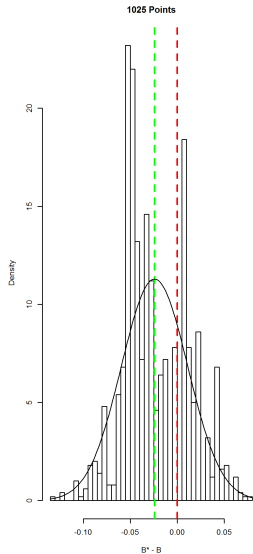
```
newdata <- house[3,]  
yy<-predict.rq(object=fit.1,newdata=newdata,type="none",  
               stepfun = FALSE)  
yy
```

```
##           3  
## 0.6479275
```

The function boot.rq

```
set.seed(4)
ns <- c(10, 12, 14)

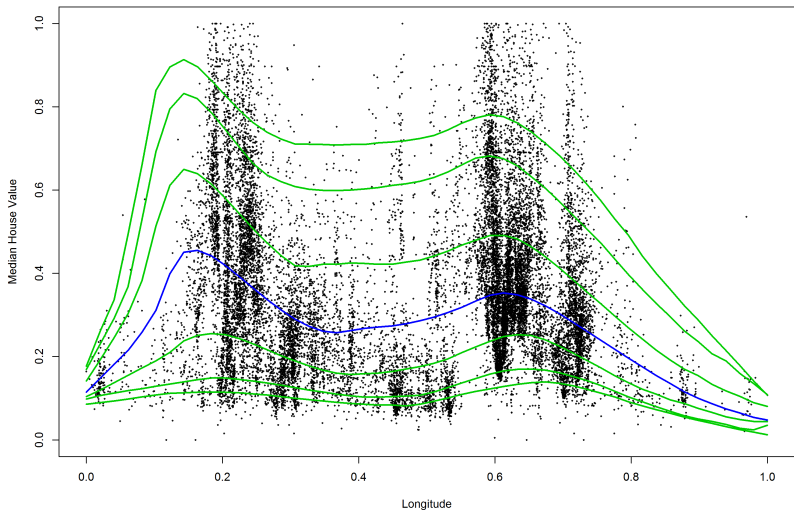
par(mfrow = c(1, length(ns)))
for( i in ns ){
  n <- (2^i)+1
  x <- seq(0, 2, length=n); y <- -x + rnorm(n)
  df <- data.frame(x, y)
  b <- boot.rq(x=x, y=y, tau=.5, R = 1000)
  main <- paste(n, "Points")
  m <- mean(b$B) + 1; s <- sd(b$B)
  hist(b$B + 1, breaks = 30, prob=TRUE,
       xlab = "B* - B", main = main)
  abline(v=0, add=TRUE, col='red', lty=2, lwd=2)
  abline(v=m, add=TRUE, col='green', lty=2, lwd=2)
  curve(dnorm(x, m, s), add=TRUE)
}
```



The function `lprq`

This function does locally polynomial quantile regression univariate smoothing

```
plot(house$longitude, house$median_house_value,
     xlab="Longitude", ylab="Median House Value",
     cex=.25, cex.lab=1, cex.axis=1, lheight=0.5)
for(tau in taus){
  fit <- lprq(house$longitude, house$median_house_value,
              tau=tau, h=0.1, m=50)
  if( tau == 0.5){
    lines(fit$xx, fit$f_v, col="4", lwd=2)
  }
  else{
    lines(fit$xx, fit$f_v, col="green3", lwd=2)
  }
}
```



The function rqss

rqss fits additive quantile regression models with possible univariate and/or bivariate nonparametric terms.

```
rqss(formula, tau = 0.5, data = parent.frame(), weights,  
      na.action, method = "sfn", lambda = NULL,  
      contrasts = NULL, ztol = 1e-5, control, ...)
```

```
fhouse<-house  
fhouse$longitude <- round(fhouse$longitude * 100)  
fhouse$latitude <- round(fhouse$latitude * 100)  
fit <- rqss(median_house_value ~ qss(cbind(longitude,latitude),  
                                       lambda = 5),  
            data = fhouse)
```

The resulting fit can be plotted with the function `plot.rqss`

```
plot.new()

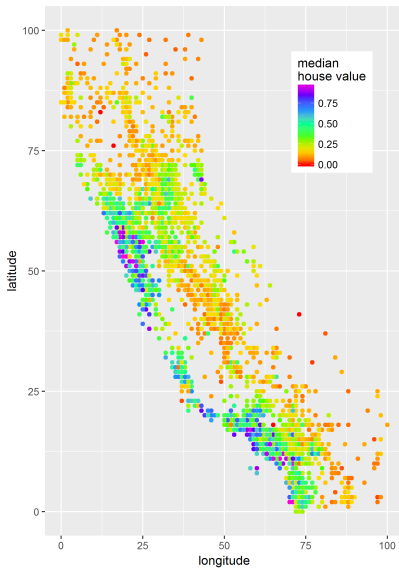
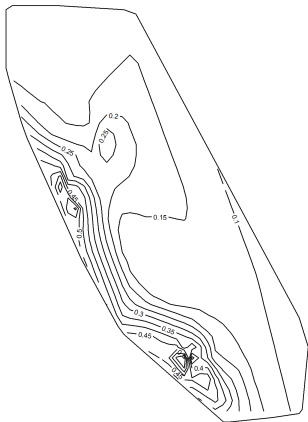
gl <- grid.layout(nrow=1, ncol=2)
vp.1 <- viewport(layout.pos.col=1, layout.pos.row=1)
vp.2 <- viewport(layout.pos.col=2, layout.pos.row=1)

pushViewport(viewport(layout=gl))
pushViewport(vp.1)

par(new=TRUE, fig=gridFIG())
plot.rqss(fit, axes = FALSE, xlab = "", ylab = "",
         render="contour", bands="uniform")
popViewport()

pushViewport(vp.2)
ggplotted <- ggplot(fhouse) + geom_point(
  mapping=aes(x=longitude, y=latitude,
              col=median_house_value)) +
  scale_color_gradientn("median\house value",
                        colours = rainbow(7)) +
  theme(legend.position = c(.8, .8))
print(ggplotted, newpage = FALSE)

popViewport(1)
```



the command `predict.rqss` allows to predict the values

```
predict.rqss(object, newdata, interval = "none",  
              level = 0.95, ...)
```

```
newdata1<-fhouse[c("longitude","latitude")]  
newdata1$median_house_value<-  
  predict.rqss(fit, newdata=newdata1, interval="none")
```

