Statistics 2 Pi

Team 8

April 23, 2018

Contents

1 Task 36 **2**

1 Task 36

We will take the 2-parameter Weibull distribution:

$$F_{\lambda,\beta}(x) = \begin{cases} 1 - e^{-(\frac{x}{\lambda})^{\beta}}\,,\, x \geq 0 \\ 0\,,\, x < 0 \end{cases}$$

$$f_{\lambda,\beta}(x) = \begin{cases} \frac{\beta}{\lambda}\left(\frac{x}{\lambda}\right)^{\beta-1} e^{-\left(\frac{x}{\lambda}\right)^{\beta}}\,,\, x \geq 0 \\ 0\,,\, x < 0 \end{cases}$$

with $\beta > 0$ is the shape and $\lambda > 0$ is the scale parameter.

In the task we'll compare our results with the results of a popular package *fitdistrplus*.

Let's download the data into R:

```
> require(fitdistrplus)
> initial <- read.table("http://statmath.wu.ac.at/~hornik/QFS2/Data/Theft.txt")
> Theft <- unname(unlist(initial))
> len <- length(Theft)
```

a) Estimation of parameters

Method of percentiles

In this method one should equal the quantile values for the given data to the theoretical percentiles for the distribution to estimate the necessary parameters. In Weibull distribution we have two parameters: shape (β) and scale (λ).

We use the first and the third quartiles of the given data set, denoted by $x_{0.25}$ and $x_{0.75}$, and we need to solve the following system of equations:

$$\begin{cases} 1 - e^{-(\frac{x_{0.25}}{\lambda})^{\beta}} = 0.25 \\ 1 - e^{-(\frac{x_{0.75}}{\lambda})^{\beta}} = 0.75 \end{cases}$$

The system yields to:

$$\begin{cases} \beta = \frac{log(\frac{log(1-0.25)}{log(1-0.75)})}{log(\frac{x_{0.25}}{x_{0.75}})} \\ \lambda = \frac{x_{0.75}}{(-log(1-0.75))^{\frac{1}{\beta}}} \end{cases}$$

```
> g1_W <- 0.25
> g2_W <- 0.75
> x1_T <- quantile(Theft, probs = g1_W, na.rm = FALSE,  names = FALSE)
> x2_T <- quantile(Theft, probs = g2_W, na.rm = FALSE,  names = FALSE)
> shape_T_QME <- log(log(1-g1_W)/log(1-g2_W))/log(x1_T/x2_T)
> shape_T_QME
```

```
[1] 0.8475029
```

```
> scale_T_QME <- x2_T/(-log(1-g2_W))^(1/shape_T_QME)
> scale_T_QME
```

```
[1] 1178.74
```

Let's compare with *fitdistrplus*:

```
> fit.weibull_qme <- fitdist(Theft, "weibull", method="qme", probs = c(g1_W, g2
> summary(fit.weibull_qme)
```

```
Fitting of the distribution ' weibull ' by matching quantiles
Parameters :
          estimate
shape    0.8474827
scale 1178.7486950
Loglikelihood:  -1028.834   AIC:  2061.669   BIC:  2067.244
```

Our results are almost the same.

MLE

The likelihood function for the Weibull distribution is:

$$\mathcal{L}_{\hat{x}}(\lambda,\beta)=\prod_{i=1}^{N} f_{\lambda,\beta}(x_i)=\prod_{i=1}^{N}\frac{\beta}{\lambda}\left(\frac{x_i}{\lambda}\right)^{\beta-1}e^{-\left(\frac{x_i}{\lambda}\right)^{\beta}}=\frac{\beta^N}{\lambda^{N\beta}}e^{-\sum_{i=1}^{N}\left(\frac{x_i}{\lambda}\right)^{\beta}}\prod_{i=1}^{N}x_i^{\beta-1}$$

loglikelihood function:

$$\ell_{\hat{x}}(\lambda,\beta):=\ln\mathcal{L}_{\hat{x}}(\lambda,\beta)=N\ln\beta-N\beta\ln\lambda-\sum_{i=1}^{N}\left(\frac{x_i}{\lambda}\right)^{\beta}+(\beta-1)\sum_{i=1}^{N}\ln x_i$$

MLE-Problem:

$$\begin{aligned}\max_{(\lambda,\beta)\in\mathbb{R}^2}\quad & \ell_{\hat{x}}(\lambda,\beta)\\ \text{s.t.}\quad & \lambda>0\\ & \beta>0\end{aligned}$$

Maximization by 0-gradients:

$$\frac{\partial l}{\partial \lambda} = -N\beta\frac{1}{\lambda} + \beta\sum_{i=1}^{N} x_i^{\beta}\frac{1}{\lambda^{\beta+1}} \stackrel{!}{=} 0$$

$$\frac{\partial l}{\partial \beta} = \frac{N}{\beta} - N\ln\lambda - \sum_{i=1}^{N}\ln\left(\frac{x_i}{\lambda}\right)e^{\beta\ln\left(\frac{x_i}{\lambda}\right)} + \sum_{i=1}^{N}\ln x_i \stackrel{!}{=} 0$$

It follows:

$$-N\beta\frac{1}{\lambda} + \beta\sum_{i=1}^{N} x_i^{\beta}\frac{1}{\lambda^{\beta+1}} = 0$$

$$-\beta\frac{1}{\lambda}N + \beta\frac{1}{\lambda}\sum_{i=1}^{N} x_i^{\beta}\frac{1}{\lambda^{\beta}} = 0$$

$$-1 + \frac{1}{N}\sum_{i=1}^{N} x_i^{\beta}\frac{1}{\lambda^{\beta}} = 0$$

$$\frac{1}{N}\sum_{i=1}^{N} x_i^{\beta} = \lambda^{\beta}$$

$$\Rightarrow \lambda^* = \left(\frac{1}{N}\sum_{i=1}^{N} x_i^{\beta^*}\right)^{\frac{1}{\beta^*}}$$

Plugging λ^* into the second 0-gradient condition:

$$\Rightarrow \beta^* = \left[\frac{\sum_{i=1}^{N} x_i^{\beta^*}\ln x_i}{\sum_{i=1}^{N} x_i^{\beta^*}} - \overline{\ln x}\right]^{-1}$$

This equation is only numerically solvable.
So let's make MLE estimation in R:

```
> # Weibull MLE
> model <- function(x) {
+   F1 <- (sum(Theft^x[1]*log(Theft))/sum(Theft^x[1]) - mean(log(Theft))) ^ (-1
```

```
+   c(F1 = F1)
+ }
> shape_T_MLE <- rootSolve::multiroot(f = model, start = c(0.85))$root[1]
> shape_T_MLE
```

```
[1] 0.7157348
```

```
> scale_T_MLE <- ((sum(Theft^shape_T_MLE))/length(Theft))^(1/shape_T_MLE)
> scale_T_MLE
```

```
[1] 1557.191
```

Let's compare with *fitdistrplus*:

```
> # to compare with fitdistrplus
> fit.weibull_mle <- fitdist(Theft, "weibull", method="mle")
> summary(fit.weibull_mle)
```

```
Fitting of the distribution ' weibull ' by maximum likelihood
Parameters :
          estimate   Std. Error
shape    0.7158071   0.04733433
scale 1557.0525771 210.29278380
Loglikelihood:  -1017.429   AIC:  2038.858   BIC:  2044.433
Correlation matrix:
         shape    scale
shape 1.000000 0.328723
scale 0.328723 1.000000
```

The results are pretty close again. We can compare values of loglikelihood functions:

```
> loglike <- len*log(shape_T_MLE) - len*shape_T_MLE*log(scale_T_MLE) - sum((The
> formatC(loglike, digits = 7, format = "f") # by explicit formulas
```

```
[1] "-1017.4290040"
```

```
> formatC(summary(fit.weibull_mle)$loglik, digits = 7, format = "f") # by fitdi
```

```
[1] "-1017.4290060"
```

It is clear that the difference appears just on the 6th decimal place.

b) Bootstrapping to estimate the standard errors of the estimates

Method of percentiles

At first, we use a bootstrap to make the samples with estimates. We decided to make a matrix with 1001 samples with each row as one bootstrap sample. Then for every sample we estimate parameters by the method of percentiles as we did for the original data set.

```
> B <- 1001 # number of simulations
> shape_QME <- c(rep(0,B))
> scale_QME <- c(rep(0,B))
> x1 <- c(rep(0,B))
> x2 <- c(rep(0,B))
> s_QME <- matrix(0, B, len) # matrix of samples
> for (i in 1:B) {
+   s_QME[i,] <- sort(rweibull(len, shape_T_QME, scale_T_QME))
+   x1[i] <- quantile(s_QME[i,], probs = g1_W, na.rm = FALSE,  names = FALSE)
+   x2[i] <- quantile(s_QME[i,], probs = g2_W, na.rm = FALSE,  names = FALSE)
+ shape_QME[i] <- log(log(1-g1_W)/log(1-g2_W))/log(x1[i]/x2[i])
+ scale_QME[i] <- x2[i]/(-log(1-g2_W))^(1/shape_QME[i])
+ }
```

So the standard errors are:

```
> sd_shape_QME <- sd(shape_QME)
> sd_shape_QME # standard error of shape

[1] 0.09804499

> sd_scale_QME <- sd(scale_QME)
> sd_scale_QME # standard error of scale

[1] 154.5977
```

MLE

We make the same for MLE

```
> B <- 1001 # number of simulations
> shape_MLE <- c(rep(0,B))
> scale_MLE <- c(rep(0,B))
> s_MLE <- matrix(0, B, len) # matrix of samples
> for (i in 1:B) {
```

```
+   s_MLE[i,] <- rweibull(len, shape_T_MLE, scale_T_MLE)
+   modelle <- function(x) {
+     F1 <- (sum(s_MLE[i,]^x[1]*log(s_MLE[i,]))/sum(s_MLE[i,]^x[1]) - mean(log(
+     c(F1 = F1)
+   }
+
+   shape_MLE[i] <- rootSolve::multiroot(f = modelle, start = c(0.85))$root[1]
+   scale_MLE[i] <- ((sum(s_MLE[i,]^shape_MLE[i]))/length(s_MLE[i,]))^(1/shape_
+ }
```

So the standard errors are:

```
> sd_shape_MLE <- sd(shape_MLE)
> sd_shape_MLE # standard error of shape
```

```
[1] 0.05370651
```

```
> sd_scale_MLE <- sd(scale_MLE)
> sd_scale_MLE # standard error of scale
```

```
[1] 207.505
```

Our result for shape standard error is slightly higher than in *fitdistrplus*, but our result for scale standard error is slightly lower than in *fitdistrplus*.

c) 95% confidence intervals for the parameters

In principle, there exist different approaches for estimation of confidence intervals using bootstrapping.
We use the approach from the lecture notes:
If the distribution of $\Delta = \hat{\theta} - \theta_0$ (where θ is a parameter of distribution) was known, confidence intervals could be obtained via

$$P(Q_\Delta(\alpha/2) \leq \hat{\theta} - \theta_0 \leq Q_\Delta(1 - \alpha/2)) = 1 - \alpha$$

as

$$P(\hat{\theta} - Q_\Delta(1 - \alpha/2) \leq \theta_0 \leq \hat{\theta} - Q_\Delta(\alpha/2)) = 1 - \alpha$$

But since θ_0 is is not known, we use $\hat{\theta}$ in its place: we generate B bootstrap samples from the distribution with value $\hat{\theta}$, and compute the respective estimates of θ_b^* by the method chosen. The distribution of $\hat{\theta} - \theta_0$ is then approximated by that of $\theta^* - \hat{\theta}$ and the quantiles of this are used to form the approximate confidence interval.
Method of percentiles
The boundaries of approximated confidence interval by method of percentiles are as follows:

```
> alpha <- 0.05
> delta_shape_QME <- sort(shape_QME - shape_T_QME)
> lower_shape_QME <- shape_T_QME -  quantile(delta_shape_QME, probs = 1 - alpha
> lower_shape_QME

[1] 0.6141172

> upper_shape_QME <- shape_T_QME -  quantile(delta_shape_QME, probs = alpha/2,
> upper_shape_QME

[1] 1.008488

> delta_scale_QME <- sort(scale_QME - scale_T_QME)
> lower_scale_QME <- scale_T_QME -  quantile(delta_scale_QME, probs = 1 - alpha
> lower_scale_QME

[1] 854.353

> upper_scale_QME <- scale_T_QME -  quantile(delta_scale_QME, probs = alpha/2,
> upper_scale_QME

[1] 1456.808
```

Results of *fitdistrplus*:

```
> bd.weibull_qme <- bootdist(fit.weibull_qme, bootmethod = "param", niter = 100
> summary(bd.weibull_qme)

Parametric bootstrap medians and 95% percentile CI
           Median       2.5%       97.5%
shape    0.855844   0.6923288    1.064457
scale 1162.733884 905.9685049 1497.270753
```

MLE

The boundaries of approximated confidence interval by MLE are as follows:

```
> delta_shape_MLE <- sort(shape_MLE - shape_T_MLE)
> lower_shape_MLE <- shape_T_MLE -  quantile(delta_shape_MLE, probs = 1 - alpha
> lower_shape_MLE

[1] 0.5898405

> upper_shape_MLE <- shape_T_MLE -  quantile(delta_shape_MLE, probs = alpha/2,
> upper_shape_MLE
```

```
[1] 0.8031262

> delta_scale_MLE <- sort(scale_MLE - scale_T_MLE)
> lower_scale_MLE <- scale_T_MLE -  quantile(delta_scale_MLE, probs = 1 - alpha
> lower_scale_MLE

[1] 1113.75

> upper_scale_MLE <- scale_T_MLE -  quantile(delta_scale_MLE, probs = alpha/2,
> upper_scale_MLE

[1] 1929.816
```

Results of *fitdistrplus*:

```
> bd.weibull_mle <- bootdist(fit.weibull_mle, bootmethod = "param", niter = 10C
> summary(bd.weibull_mle)

Parametric bootstrap medians and 95% percentile CI
            Median         2.5%        97.5%
shape    0.7184324    0.6305841    0.8304139
scale 1545.1041899 1164.0259813 1992.9283754
```

As a result, standard error of shape parameter is less with MLE. However, standard error of scale parameter is less with method of percentiles. Therefore, 95% confidence interval for shape parameter is narrower with MLE and for scale parameter it is narrower with the method of percentiles. Confidence intervals calculated by approach given in lecture notes are somewhat skewed to the left in comparison with what is calculated by *fitdistrplus*.