Problem Set 1

Yana Konshyna

February 11, 2024

Instructions

- Please show your work! You may lose points by simply writing in the answer. If the problem requires you to execute commands in `R`, please include the code you used to get your answers. Please also include the `.R` file that contains your code. If you are not sure if work needs to be shown for a particular problem, please ask.
- Your homework should be submitted electronically on GitHub in `.pdf` form.
- This problem set is due before 23:59 on Sunday February 11, 2024. No late assignments will be accepted.

Question 1

The Kolmogorov-Smirnov test uses cumulative distribution statistics test the similarity of the empirical distribution of some observed data and a specified PDF, and serves as a goodness of fit test. The test statistic is created by:

$$D = \max_{i=1:n} \left\{ \frac{i}{n} - F_{(i)}, F_{(i)} - \frac{i-1}{n} \right\}$$

where F is the theoretical cumulative distribution of the distribution being tested and $F_{(i)}$ is the ith ordered value. Intuitively, the statistic takes the largest absolute difference between the two distribution functions across all x values. Large values indicate dissimilarity and the rejection of the hypothesis that the empirical distribution matches the queried theoretical distribution. The p-value is calculated from the Kolmogorov- Smirnoff CDF:

$$p(D \leq d) = \frac{\sqrt{2\pi}}{d} \sum_{k=1}^{\infty} e^{-(2k-1)^2\pi^2/(8d^2)}$$

which generally requires approximation methods (see Marsaglia, Tsang, and Wang 2003). This so-called non-parametric test (this label comes from the fact that the distribution of the test statistic does not depend on the distribution of the data being tested) performs poorly in small samples, but works well in a simulation environment. Write an `R` function that implements this test where the reference distribution is normal. Using `R` generate 1,000 Cauchy random variables (`rcauchy(1000, location = 0, scale = 1)`) and perform the test (remember, use the same seed, something like `set.seed(123)`, whenever you're generating your own data).

As a hint, you can create the empirical distribution and theoretical CDF using this code:

```r
  # create empirical distribution of observed data
  ECDF <- ecdf(data)
  empiricalCDF <- ECDF(data)
  # generate test statistic
  D <- max(abs(empiricalCDF - pnorm(data)))

```

1.Creating data:

```r
set.seed(123)
# Generating 1,000 Cauchy random variables
cauchyData <- rcauchy(1000, location = 0, scale = 1)
```

2.Creating function `ksTestFunction`, which accept our data and then calculate test-statistics and p-value:

```r
ksTestFunction <- function(data){
  # creating empirical distribution of observed data
  ECDF <- ecdf(data)
  empiricalCDF <- ECDF(data)
  # generating test statistic
  D <- max(abs(empiricalCDF - pnorm(data)))
  # calculating p-value
  pValue <- sqrt(2*pi)/D * sum(sapply(1:1000, function(k) {
    exp(-(2*k - 1)^2 * pi^2 / (8 * D^2))
  }))
  # printing result of the function
  list(D = D, P_value = pValue)
  }

# Performing the K-S test on the Cauchy data comparing it
# to a normal distribution using created function
ksTestFunction(cauchyData)
```

Output:

```
$D
[1] 0.1347281

$P_value
[1] 5.652523e-29
```

3. Checking the results by using ks.test() built-in function:

```
ks.test(cauchyData, "pnorm")
```

Output:

```
Asymptotic one-sample Kolmogorov-Smirnov

testdata:  cauchyData
D = 0.13573, p-value = 2.22e-16
alternative hypothesis: two-sided
```

Question 2

Estimate an OLS regression in `R` that uses the Newton-Raphson algorithm (specifically `BFGS`, which is a quasi-Newton method), and show that you get the equivalent results to using `lm`. Use the code below to create your data.

```
set.seed (123)
data <- data.frame(x = runif(200, 1, 10))
data$y <- 0 + 2.75*data$x + rnorm(200, 0, 1.5)
```

1. Creating data:

```
set.seed (123)
data <- data.frame(x = runif(200, 1, 10))
data$y <- 0 + 2.75*data$x + rnorm(200, 0, 1.5)
```

The probability density function (PDF) for normal distribution is

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{y_i - x_i\beta}{\sigma})^2} \tag{1}$$

To find the parameters I use log of likelihood function:

$$\log(\mathcal{L}) = \log(f(x)) = -\frac{1}{2}n\log(2\pi) - \frac{1}{2}n\log(\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - x_i\beta)^2 \tag{2}$$

2. Creating log-likelihood function using formula (2):

```
logLikelihoodFunction <- function(parameter, y, X) {
  n <- nrow(X)
  k <- ncol(X)
  # estimating betas
  beta <- parameter[1:k]
  # estimating sigma squared
  sigma2 <- parameter[k+1]^2
  # estimating residuals
  e <- y - X %*% beta
  # calculating Log-likelihood
  logLikelihood <- -0.5*n*log(2*pi) - 0.5*n*log(sigma2) - (sum(e^2) / (2*
    sigma2))
  # returning result of the function
  return(-logLikelihood)
}
```

3. Doing optimization using `optim` function by using BFGS method. This function will adjust the `par` vector to minimize the negative log-likelihood, finding the best-fitting linear model to the data given the assumption of normally distributed errors:

```
optim_results <- optim(fn = logLikelihoodFunction, par = c(1, 1, 1),
                       hessian = TRUE, y=data$y, X=cbind(1, data$x), method =
    "BFGS")
```

4. Display optimization results:

```
round(optim_results$par, 2)[1:2]
```

Output:

```
0.14 2.73

```

5. Checking the results by performing OLS regression using lm function

```
lmModel <- lm(y~x, data = data)
stargazer::stargazer(lmModel)
```

Table 1: OLS regression

	Dependent variable:
	y
x	2.727***
	(0.042)
Constant	0.139
	(0.253)
Observations	200
R^2	0.956
Adjusted R^2	0.956
Residual Std. Error	1.447 (df = 198)
F Statistic	4,298.687*** (df = 1; 198)
Note:	*p<0.1; **p<0.05; ***p<0.01