Problem Set 1

Jacqueline Bouvier Applied Stats II

Due: 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:

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

```
#Creating ks.test by hand.
# https://stats.stackexchange.com/questions/471732/intuitive-explanation-of-kolmogorov-smirnov-test
my_ksTest <- function(data) {
  l <- length(data) # l will equal the distribution given
  s_data <- sort(data) # l will need to be sorted
  empirical_CDF <- ecdf(s_data)
  theoretical_CDF <- pnorm(s_data)
  D <- max(abs(empirical_CDF(s_data) - theoretical_CDF))
  p_value <- 1 - 2 * sum((-1)^(1:(l-1)) * exp(-(2 * (1:(l-1)) - 1)^2 * pi^2 / (8 * D^2))) #https://stats.stackexchange.com/questions/389034/kolmogorov-smirnov-test-calculating-the-p-value-manually
  return(list(D = D, p_value = p_value))
}

#testing function

cauchy_variables <- rcauchy(1000, location = 0, scale = 1)

test_ks <- my_ksTest(cauchy_variables)
print(test_ks)

#checking function

check <- ks.test(cauchy_variables, "pnorm")
print(check)
```

]

ks.test formula and written funciton resulted in the same D value of 0.1439423, p-value for the written was 1 and p-value for ks.test was 2.2e-16

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)


ols_reg <- function(beta, x, y) {
  y_hat <- beta[1] + beta[2] * x
  sum((y - y_hat)^2) # least squares regression
}

# need to have x and y values from main dataset
x <- data$x
y <- data$y


# Use optim function with BFGS mhttps://stackoverflow.com/questions/71268528/using-pgtol-in-rs-optim-function
result <- optim(par = c(0,0), fn = ols_reg, x = x, y = y, method = "BFGS")

coef_bfgs <- result$par
print(coef_bfgs)

# linear model
lm_model <- lm(y ~ x, data = data)
lm_coef_model <- coef(lm_model)

print(lm_coef_model)
```

The goal of Newton-Raphson method in OLS is to give an estimate the given data using aproximation and least squares calculations. This allows for a more linear model. OLS can be similar as we would get the estimator that close or identical to one we would see in an lm function. Any difference we would see is because of the vairence in OLS to be unbiased as it is different from the least squares estimator..