Practice tasks: Bivariate and multivariate regression models

Note: You need to install R to run the code below. Its installer (for Unix/Linux, MacOS, and Windows) can be found at https://www.r-project.org/.
It is strongly recommended to also install RStudio, an integrated development environment with graphical user interface for R, which can be downloaded at https://posit.co/download/rstudio-desktop/.
If you need information about the R functions used below, type ?functionname in the R console (example: ?lm).

Task 1
The code below generates a data set with 50 observations for variables X and Y. Y is a function of X, with some error added to it (controlled by parameter 'sigma').

```
## Setting RNG for replicability
set.seed(8419357) # It can be any other number.

## Data generation
# Number of data points
n <- 50
# Value of x
x <- 1:n
# Regression parameters: Intercept
a <- 10
# Regression parameters: Coefficient/slope
b <- 0.3
# Error variance
sigma <- 3
# Value of y
y <- a + b*x + sigma*rnorm(n, mean = 0, sd = 1)
# Data set
data1 <- data.frame(x, y)
```

You can visualize the data using:

```
## Plot graph
par(mfrow = c(1,1))
plot(data1$x, data1$y)
```

Now, fit a bivariate (simple) linear regression model, regressing Y on X using the code below.

```
fit_1 <- lm(y ~ x, data = data1)
```

(a) Inspect the output from the regression model, stored in the object fit_1 using:

```
summary(fit_1)
```

Calculate and report the 95% confidence interval for the model parameters. Do the CIs include the true parameters used to generate the data?

(b) Interpret the coefficient parameters. What does the regression coefficient (slope) mean? What does the intercept mean? Any critical comment on how they should be interpreted?

(c) Extrapolation: What is the expected value for Y if X = 67? Write the equation used to calculate the value and explain what you did.

(d) Recall the meaning of residuals in a regression model. What does it mean?

(e) Run the data generating code again, with different values for 'sigma'. You should use smaller and larger values for sigma, and generate new datasets with the values of sigma of your choice. For each new dataset, run the same regression model (it may be a good idea to use different names for the objects storing the outputs). Inspect (and report) the regression parameters for each of the models. Are the parameters similar to those in fit_1?

(f) Using the same regression results in item (e): Pay close attention to the Residual Standard Error (RSE) for each regression. What is the relation between sigma and RSE? Explain.

Task 2

Recall a quote from one of the lectures: “The standard error of a parameter is the standard deviation of its sampling distribution.”

Let’s work it out using simulated data: Pretend we are analyzing data from a Russian city with a population of approximately 1.5 million. (Say, Novosibirsk.)

```
## Setting RNG for replicability
set.seed(8419357) # It can be any other number.

## Generate the "population"
# Pop size
N_pop <- 1500000
# Population parameters
a_pop <- 25
b_pop <- 0.3
sigma_pop <- 7.5
# Generate x
x_pop <- runif(N_pop, min = 1, max = 1000)
# Generate the noise in y
error_pop <- rnorm(N_pop, mean = 0, sd = sigma_pop)
# Generate y
y_pop <- a_pop + (b_pop * x_pop) + error_pop
# Putting x_pop and y_pop in a dataset, for further manipulation
pop <- data.frame(
  id = 1:N_pop,
  x = x_pop,
  y = y_pop,
  error = error_pop
```

Now, we will draw samples from the population — say, you work for a government agency and need to collect information for city planning purposes. Assume that you have infinite resources and will collect data not from one sample, but from one thousand samples, and in each sample you will collect data from 300 individuals.

```
n_samples <- 1000
n_obs <- 300
samples <- list()
set.seed(8419357)
for (i in 1:n_samples) {
  index <- sample(1:nrow(pop), size = n_obs, replace = FALSE)
  samples[[i]] <- pop[index, ]
}
```

Regress y_pop on x_pop for each one of the samples, and store the regression coefficient (slope) parameters. Use the code below:

```
regr_outputs <- list()
coef_estimates <- numeric(n_samples)
standard_errors <- numeric(n_samples)
for (i in 1:n_samples) {
  regr_outputs[[i]] <- lm(y ~ x, data = samples[[i]])
  coef_estimates[i] <- coef(regr_outputs[[i]])[2]
  standard_errors[i] <- summary(regr_outputs[[i]])$coefficients[2, 2]
}
```

(a) Create a histogram of the coefficients. You can use `hist()` or any other plot of your choice. (Use the argument `break` in `hist()` to control the number of bins displayed in the histogram, for visualization purposes.)

(b) Calculate and report the mean and the standard deviation for parameters in the object `coef_estimates`, as well as the 95% confidence interval.

(c) Calculate and report the mean and the standard deviation for parameters in the object `standard_errors`, as well as the 95% confidence interval.

(d) Compare the results for (b) and (c). Explain.

(e) What would happen to the standard error of the regression coefficient if you rather draw 100 samples of 300 observations? What if you draw 1000 samples of 100 observations? Provide an explanation.

Task 3

Download the dataset `kidiq` (the same used in class) from: https://github.com/avehtari/ROS-Examples/raw/refs/heads/master/KidIQ/data/kidiq.rda and load it. (If you have RStudio, just double-click on it and it will be loaded.) We will use three variables in this task:

- `kid_score`: score of a child in an exam;
- `mom_hs`: whether the child's mothers has completed (=1) or not (=0) high school/secondary education;
- `mom_iq`: the child's mother IQ.

(a) Run regression model: Regress `kid_score` on `mom_hs` and `mom_iq` and save the regression output as an object — you choose the name.

(b) Interpret the all the model parameters.