MS6051 Statistical Inference Assignment (20% of Module)

Practical Information

- You will be randomly assigned into a group of two students. You must work together as a team on this project. Both of you must be able to answer questions in an interview that will take place in Week 12/13.

- This assignment must be your own work. Evidence of plagiarism will result in a score of zero, and will be reported to Student Academic Administration. In this context, you have to be extremely careful if using recent chat-based AI systems as these systems can only plagiarise (or hallucinate incorrect content). The best advice is to avoid using such systems to avoid any possible doubt. If you do use such systems, then this should only be essentially as a search tool, or to improve grammer and fix typos. It should not be used to actually generate any project work, i.e., coding or writing. As per the first point above, you must be able to explain your work in an interview, and there is no short cut to acquiring knowledge.

- Deadline: **Friday, November 28th (Week 12) before 6.00pm**. Late submissions will have their score discounted according to the penalty $\exp(-t/50)$ where t is in hours, e.g., 24hrs late reduces by about 40%, 48hrs late about 60%, and so on.

- You must submit the following to the "Assignments" section on Brightspace. Do not email your project.

  - Report should be saved as .pdf file. This may be prepared in LaTeX (preferred) or Microsoft Word. Use 11pt text throughout, and use the `courier` style font for any small snippets of `R` code/output presented. However, do not copy all of your `R` code into this document.

  - `R` script file should be saved as .txt file (not .R or .Rmd). This will contain all code used to produce the outputs provided in the report. The code must be organised so that it can exactly replicate all results in the order in which they appear in your report.

  Note: No hard copies are required.

Assignment

Use the `R` programming language to carry out the tasks below. Produce a document that contains the results (i.e., graphs and tables), along with clear explanations/intepretations in your own words.

1) **Exponential data: estimation bias and efficiency**

   i) Using the function `rexp`, create a function that generates a sample of exponential data and computes the maximum likelihood estimator (MLE) of λ. Call this function `mle_exp`. The inputs of this function should be the value of λ and the sample size, n.

   ii) Generate one MLE using `mle_exp` with $\lambda = 1$ and $n = 10$. Note that any time you run this, you will get different results. Therefore, to ensure that I (the examiner) can replicate the exact result which appears in your report, insert `set.seed(SEEDNUMBER)` in your code just prior to `mle_exp` where `SEEDNUMBER` is the sum of your two student ID numbers.

   iii) The result of (ii) only applies to *one* sample so does not tell us much about the general properties of the estimator; for this we need replicate samples. Using a `for` loop, repeat part (ii) 1000 times. Now calculate the mean and variance of these 1000 MLEs. Also calculate the mean squared error. Note: just prior to your loop, insert `set.seed(SEEDNUMBER)`.

   iv) Repeat step (iii) but for $n = 20$, $n = 50$, $n = 200$, $n = 400$, $n = 1000$ (again insert `set.seed(SEEDNUMBER)` just prior to each loop), and produce the following graphs:

   - bias versus sample size,
   - efficiency versus sample size (note: you need to calculate the CRLB by hand first), and
   - mean squared error versus sample size.

   In each of these graphs, indicate the values of the analytic bias, efficiency, and mean squared error, i.e., those computed by hand. Show the hand calculations for these quantities and also for the CRLB.

   v) Now repeat step (iv) but for $\lambda = 0.5$ and $\lambda = 2$, and do not forget `set.seed(SEEDNUMBER)`. Discuss the findings for the various λ values and sample sizes investigated.

Question 1 provides the blueprint for carrying out a "simulation study", i.e., repeat an estimation procedure over a number of simulation replicates (typically 1000 replicates as used above), to investigate the properties of this procedure. Note that, for the exponential example of Question 1, you will calculate bias, efficiency, and mean squared error by hand. However, for more complicated situations where nothing can be calculated by hand, a simulation study like in Question 1 is very useful.

Question 1 also highlights the importance of using `set.seed` so that the results you place in your report can be replicated exactly when your code is rerun, e.g., by an examiner, or even by yourself at another time.

Questions 2 and 3 are also simulation studies. It is assumed that you now know how to carry out such a study, including the use of `set.seed`, and that you will discuss the findings of each question in your report, i.e., do not simply display results without accompanying discussion/interpretation.

2) Comparing maximum likelihood and method of moments

i) Consider the distribution with density function given by

$$f(x) = c\, x^{\theta-1}$$

where $\theta > 0$ and $x \in [0, x_{\max}]$. Randomly select a value for $x_{\max}$ to be used throughout this question by running `sample(2:15, size = 1)` (with `set.seed` prior to it for reproducibility). Note that this is quite similar to Q3 in Tutorial 2 but $x_{\max} \neq 1$ here.

ii) By hand, compute the MLE and the MoM estimator for θ. Show all workings.

iii) Unlike Q1, this is a non-standard distribution, which you must simulate from yourself. This can be done using the method of inversion described here. First, derive the cumulative distribution function (cdf), $F(x) = \int_0^x f(u)du$, and then the associated inverse cdf, $F^{-1}(x)$. Next, the random variable X can be generated as $X = F^{-1}(U)$ where $U \sim \text{Uniform}(0, 1)$, and uniform variables can be generated in `R` via the `runif` function. Using this approach, generate samples of size $n = 5000$ for a few different values of θ values, and, using a histogram, describe the shape of the distribution and how it changes with respect to θ; also indicate the theoretical expected value $E(X)$ on the histogram.

iv) Generate *one* sample of size $n = 100$ with $\theta = 1$ and compute both the MLE and MoM.

v) Now generate two additional samples: one of size $n = 10$ and one of size $n = 1000$. For all three samples, plot the log-likelihood function with respect to θ, i.e., θ values on the x-axis and the $\ell(\theta)$ values on the y-axis. Indicate where both the MLE and MoM lie on these plots. Describe how the MLE and MoM compare to each other, and to the true value of θ, in all three cases.

vi) For each of the three log-likelihood plots, draw a horizontal line that is exactly $d/2$ units below the maximum where $d = \chi^2_{1,0.05}$. Drop two vertical lines down from where the horizontal hits the log-likelihood function to form a Wilks confidence interval for θ (see Section 4.5 of Lecture 5 and Section 5.3 of Lecture 6).

vii) For each of the three sample sizes, find the MLE numerically using the numerical optimiser `nlm`. Note: this is done by minimising *minus* the

log-likelihood (which is equivalent to maximising the log-likelihood); see Section 5.5 of Lecture 5. When using `nlm`, set `hessian = TRUE` so that `nlm` stores the so-called "hessian", i.e., the second derivative of the objective function. Since we minimise *minus* the log-likelihood, this is already the observed information. Use this to produce a 95% Wald confidence interval for θ. How do these compare with the Wilks confidence intervals from part (v)?

viii) Repeat step (vii) 1000 times. In other words, use `nlm` to numerically maximise the log-likelihood function and compute 95% Wald confidence intervals in each of 1000 samples at each of the three sample sizes (but there is no need to compute Wilks intervals). Also compute the MoM. How do the MLEs compare to the true θ value at each of the three sample sizes? And how do the MoM estimators compare? Check the coverage of the Wald confidence intervals for each of the three sample sizes, i.e., what proportion of the intervals contain the true θ value? Are the results as expected?

3) **Bootstrapping**

i) Bootstrapping is a useful technique for calculating confidence intervals. It involves resampling with replacement from the original data. Explain briefly in your own words how it works. (This technique is not contained in the lecture notes.)

ii) Generate a sample of exponential data with $n = 100$ and $\lambda = 1$. Bootstrap this data 1000 times, calculating the the median for the sample in each bootstrapped sample. The 0.025 and 0.975 quantiles of the distribution of bootstrapped medians provides a confidence interval for the median.

iii) Repeat part (ii) here 1000 times to evaluate the performance of the confidence interval. Note: here you need to work out the value of the true median first, i.e., the value of m such that $\Pr(X > m) = 0.5$ (and show your workings). Also, do not get mixed up between the 1000 bootstraps carried out in each of the 1000 simulation replicates, i.e., 1000 bootstraps are required to produce a confidence interval (per part (ii) of this question), but then this procedure is repeated 1000 times to assess the performance of the bootstrapped confidence intervals.

iv) By hand, derive the an approximate confidence interval (Wald-type) for λ (showing workings) and, hence, for the median m. Evaluate the performance of this confidence interval.

v) Repeat parts (iii) and (iv) for $n = 10$ and $n = 50$. Comment on the results.

4) **Statistical neural network regression**

Consider the regression problem of predicting a response variable Y_i using a set of explanatory variables $X_i = (X_{i1}, X_{i2}, \ldots, X_{ip})$, where

i represents the individual, where $i = 1, \ldots, n$.. Assuming a normal distribution, this can be written as

$$Y_i \mid X_i \sim N(\mu_i, \phi),$$

where $\phi = \text{var}(Y_i)$ and $\mu_i = \mu(X_i)$ is some regression function (which we will assume to be a neural network). Note that μ_i provides a prediction of Y_i.

i) Write down the log-likelihood function in terms of μ_i and ϕ. Given an estimator $\hat{\mu}_i$, show that the maximum likelihood estimator of ϕ is given by $\hat{\phi} = \text{MSE}$ where MSE is the mean-squared error in predicting the Y_i values, i.e.,

$$\hat{\phi} = \text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{\mu}_i)^2.$$

Note that this is different to MSE as used in Q1 above as it relates to prediction error rather than error in estimating a parameter.

ii) By replacing ϕ in the log-likelihood with MSE, show that the log-likehood can be written simply as

$$\ell = -n(\log \text{MSE} + \log(2\pi) + 1)/2.$$

The advantage of this is that it shows how a simple transformation of the MSE leads to a normal log-likeliood function, and minimising MSE is how the majority of machine learning models (such as neural networks) are trained on data. Ultimately, this puts a new statistical flavour on machine learning. In particular, it enables the computation of the Baysian Information Criterion,

$$\text{BIC} = -2\ell + \log(n)(k+1)$$

where k is the number of parameters in the regression function μ_i and the "+1" is to account for the additional ϕ parameter.

iii) Select a regression dataset from the UCI Machine Learning Repository where n is less than 5000, and p is between 10 and 30. See here where these filters are already in place. Coduct some basic exploratory analysis of this dataset, especially the relationships between the $X's$ and Y.
Note: You must indicate your selection by replying to the "Dataset" Thread in the Discussion within Communication on Brightspace. This is to ensure you all have different datasets. Make sure you check all previous replies so that your dataset has not already been selected.

iv) Using the `nnet` function (in the `nnet` package) in `R`, fit neural networks with $q \in \{1, 2, 3, 4, 5, 6\}$ hidden neurons to your data. For each of these models, extract the MSE to compute the BIC, where $k = (p+2)q+1$ as described in Section 2 of this paper. Describe the results and compare them to a standard linear regression model (fitted using `lm` in `R`).
Notes.

- You must set `linout = TRUE` as this is a regression problem.
- The objective function of `nnet` is the sum of squared errors, which you need to divide by n to get MSE.
- It is important to run the model a number of times (e.g., 5 to 10), keeping the one with the lowest MSE. Since the `nnet` optimisation procedure starts from different initial values each time, this gives it more chances to converge to a good final solution.
- It is also advisable to set `maxit` to be bigger than its default of 100 (e.g., set it to 1000).

v) Select the neural network with the lowest BIC from part (iv). Now, for this value of q, fit the $p-1$ models that arise from removing one explanatory variable from the full model (keeping all of the others in place). Compute the BIC values. By comparing these to the BIC of the full model, describe which variables are most important, and which are not important. How do these results compare to importance as determined by the p-values of the linear regression model?

vi) For each of the three most important variable in the selected neural network from part (v), provide an explanation of the nature of the effect on the response variable using the $\hat{\tau}$ measure used in Section 5 of this paper.

vii) Make predictions $\hat{\mu}_i$ from the selected neural network from part (v) and hence produce prediction intervals $\hat{\mu}_i \pm 1.96\,\hat{\phi}$. Compare these to prediction intervals from the linear regression model.

5) **Significance Article**

Select an article that you like from Significance Magazine here in any issue from the last 5 years. Indicate your selection in the "Significance Article" Thread in the Discussion on Brightspace and ensure no other group is using it. Write a short summary of the article, including the key topics/contributions within it, and how any of these relate to this Statistical Inference module.
Your discussion of this article should not exceed one page.
