Programming Camp: MATLAB

Brian Higgins and Ciaran Rogers

August 2021

Section 1: TripAdvisor Dataset

We are going to work with the TripAdvisor excel file that we sent to you. The code **Lecture 1 2021 Class** contains all the lines that we will go through here, excluding the Exercises at the end of this section.

- **Step 1**: Loading Data

  We start by loading the data. Ensure that TripAdvisor.xlsx is in the same folder as the script (or you have used the command 'addpath' to include the location of the excel file).

  [data textdata] = xlsread('TripAdvisor.xlsx','Raw')

- **Step 2**: Generate Indicators for Price of Restaurant

  Next, we use *unique* to identify the set of possible prices. *unique* produces as output the set of unique entries across all elements within a given matrix. In this case, Prices is a text vector that tells us how high the prices are for the given restaurant.

  Prices = unique(textdata(2:end,5))

  Then, we use *strcmp* to generate logicals from strings. In other words, it returns 1 if there is a match, and 0 otherwise. We make 4 vectors, 3 for Low, Medium and High price, and a fourth for there being no price indicator.

  NoPrice = strcmp(textdata(2:end,5),Prices(1))
  PriceLow = strcmp(textdata(2:end,5),Prices(2))
  PriceMed = strcmp(textdata(2:end,5),Prices(3))
  PriceHig = strcmp(textdata(2:end,5),Prices(4))

- **Step 3**: Clean Data

  Now, we wish to clean the data by removing:

  - Observations where merge was unsuccessful
  - Observations where there is no price indication

  In this data, the merge is unsuccessful if there is no userRating value i.e. no value in Row 3 of data. This means that the identity of the User has not been matched to the observation. The line below checks whether or not there has been a successful merge

  Merged = ∼isnan(data(:,3));

  The command 'isnan' is an indicator equalling 1 if the element is NaN. If it is preceded with a ∼, it equals 1 if the element is not NaN.

  The line below creates an indicators that tells us if the observation exhibits a Price indication:

  PriceInd = 1 - NoPrice

The line below then tells us which observation rows we are going to keep:

```
PriceandMerged = find(Merged.*PriceInd==1);
```

Finally, the two lines below create a new data matrix that contains just the rows that we want.

```
datamatrix = [data PriceLow PriceMed PriceHig];
datamatrix = datamatrix(PriceandMerged,:);
```

This leaves us with the cleaned data matrix.

- **Step 4**: Generating New Variables

  In order to ultimately run regressions on this dataset, we want to assign variable names to the data. This is done below:

```
placeId = datamatrix(:,1);
reviewNumber = datamatrix(:,2);
userRating = datamatrix(:,3);
userContributions = datamatrix(:,4);
ratingrest = datamatrix(:,6);
NumberofReviews = datamatrix(:,7);
isGreat = datamatrix(:,8);
isFriendly = datamatrix(:,9);
isFresh = datamatrix(:,10);
isOk = datamatrix(:,11);
PriceLow = datamatrix(:,12);
PriceMed = datamatrix(:,13);
PriceHig = datamatrix(:,14);
```

  Before running regressions, it may be useful to check what the data looks like. For example, if we want to look at the distribution of ratings by users, we construct a histogram in the following way:

```
figure;
histogram(userRating)
```

- **Step 5**: Running Regressions:

  Next step is to run regressions on MATLAB. To do this, we first set up the X and y variables:

```
y = userRating;
X = [ones(size(userRating)) isGreat];
```

  Then, we are ready to run the regression, obtaining the consequent values of b.

```
b = X over Y
```

EXERCISES

1. Evaluate t-statistics of the simple bi-variate regression
2. Construct a figure containing the fitted regression line against actual values
3. Repeat Exercise 1, but for multiple explanatory variables. What combination is best in predicting user Ratings?

Section 2: Fibonacci Numbers

In this section, we are going to work through the function defining Fibonacci numbers. This function exhibits the following properties:

$$f(1) = 1; f(2) = 1$$
$$f(n) = f(n-1) + f(n-2)$$

So, we are going to write functions that produce the nth number, using two different methods.

- **Recursion**
  In this case, we write the function in the following way:

```
function fib = fiborecursion(n)

if n <= 2
    fib = 1;
else
    fib = fiborecursion(n-1) + ...
        fiborecursion(n-2);
end
```

  Note: If you write "..." at the end of the line, it means that the code will combine that line and the line below as the same line for MATLAB. It is useful for aesthetics and for when you check back on your code.

  This code is written very explicitly, and is easy to read. The only issue here is that, to evaluate fiborecursion(n), you need to first figure out fiborecursion(m) for all m<n first, which could be very slow. As a result, an alternative way of writing it is required.

- **Loop**
  An alternative is to write the following function:

```
function fib = fiboloop(m)

if m<= 2
    fib = 1;
else
    fib = 1;
    fibprev = 1;

    for i = 3:n
        temp = fib;
        fib = fib + fibprev;
        fibprev = temp;
    end
end
```

  If m<=2, then we immediately know that it is 1. Otherwise, what the loop does is that, in loop i, we want to generate and store values of fiboloop(i) and fiboloop(i-1), naming them as fib = fiboloop(i) and fibprev = fiboloop(i-1). Why do this? Because, to evaluate fiboloop(n), we only need to know fiboloop(n-1) and fiboloop(n-2), not fiboloop(m) for all m<n. This avoids the issue of the recursion code.

However, getting this right can be tricky as we must continuously update on each loop. To do so, in loop (i+1), we ultimately want to change fib from fiboloop(i) to fiboloop(i+1) and fibprev from fiboloop(i-1) to fiboloop(i). To do so, we store fib as temp, then update fib to fiboloop(i+1), and then use the value of temp for fibprev, which is now fiboloop(i).

EXERCISES:

1. Write code that returns first n elements (in a loop form)
2. Use tic/toc to time recursion and loop functions; plot the differences up to n=30
3. Question: Why is one faster than the other?

Section 3: Pareto Distributions

Let's consider the following distribution:

$$F_X(x) = \begin{bmatrix} 1 - \frac{x_m}{x}^{\alpha} & if \quad x \geq x_m \\ 0 & if x < x_m \end{bmatrix}$$

$$f_X(x) = \begin{bmatrix} \frac{\alpha x_m^{\alpha}}{x^{\alpha+1}} & if \quad x \geq x_m \\ 0 & if x < x_m \end{bmatrix}$$

EXERCISES

1. Write a function for the Pareto PDF ($\alpha = 2$; $x_m = 1$), call it paretopdf.m
2. Write a function for the Pareto CDF, call it paretocdf.m
3. Plot the Pareto PDF
4. Define a grid x that covers (3,4), with step size d
5. Evaluate Pareto density on x, store as fx
6. Using the grid from 4., and fx, figure out a way to approximate the likelihood of the Pareto distributed random variable to lie within [3 4].