Programming assignment #2

Course: CHE1147H - Data Mining in Engineering

1 Criminal investigation

1.1 Introduction

You are a member of the Data Science team in the Crime Investigation Unit of Toronto Police. The incoming and outgoing phone calls of a suspect over a few months period are given to you for analysis. Your manager has asked you to answer the following questions.

1.2 Most and least frequent outgoing numbers

1. Identify the 3 numbers with the most **counts** of outgoing calls.
2. Identify the numbers with the least **counts** of outgoing calls. If there are ties (e.g. 1 phone call for many numbers) then identify all numbers.

Final answer format: two tables that include two columns: the numbers the suspect called; the counts of times the respective numbers were called.

1.3 Highest and lowest total duration of outgoing calls

1. Repeat the same exercise as in the previous subsection, but with the total **duration per number** called.

Final answer format: two tables that include two columns: the numbers the suspect called; the total duration (i.e. sum) each number were called in the period given.

1.4 Location of outgoing calls

1. What are the number of **counts** for every location of outgoing calls?

Final answer format: a table that includes two columns: the location of the outgoing call; the counts of outgoing calls per location.

1.5 Incoming and Outgoing calls pattern

1. Calculate the **total duration** of incoming and outgoing calls per month.
2. Plot the results in one graph for both incoming and outgoing calls.
3. Do you see anything unusual in the pattern? E.g. does this look like a phone line that was used for criminal activities or more like a personal phone?

Final answer format #1: a table that includes three columns: the month, the total duration of the incoming calls per month; the total duration of the outgoing calls per month. **Final answer format #2:** a plot of the table in #1 with two lines and square symbols to show the data with the plot legend showing which line is incoming and which line is outgoing. **Hint for #3:**Use **all** the information you derived in the previous questions.

1.6 Distribution of calls

Plot the histogram of the **outbound call counts**; e.g. x-axis: the call counts per client called, y-axis: number of clients called.

2 The ergodicity problem in economics

2.1 Introduction

Here, we will reproduce some simulations from a simple gambling experiment used in the recent paper "The ergodicity problem in economics" by Ole Peters (for a simpler description of the problem and its implications refer to the Bloomberg article here).

Ergodicity is a term used mainly in equilibrium statistical mechanics and in many other areas including economics. A process is called ergodic if the time average and the expectation value are equal. The author, a physicist who specializes in statistical mechanics, claims that its use in economics is abusive and it has led to the narrative of human irrationality.

2.2 The statistical problem

Starting with an initial wealth of 100$, you engage into a simple coin-flipping game in which your wealth increases by 50% every time you flip heads and you lose 40% if you flip tails. Since tossing heads or tails is just as likely, it makes sense to accept such a game if you play enough times because your potential gain is larger than your potential loss each time you play. Yet, as we will see this is a paradox.

2.3 Coding the problem

Below is a description of a simple set of steps you can follow to code this problem. It is not optimal for computation, but it is simple to understand the steps. If you want to follow a different logic

1. Generate a pandas dataframe that simulates **N** coin tosses for **w** gamblers with the np.random.rand function. Set **N**=100 coin tosses and **w**=100 gamblers and add the prefix 'prob_' to the column name.

2. Initialize an additional **w** columns ['balance_' + str(x)] with the value 100 to capture the initial wealth of 100$, where: x in range(0, w).

3. Create a two-level for loop that iterates through rows 1 to N first and columns w to 2×w next implementing the calculation logic for every prob-balance pair:

$$Balance^i = \begin{cases} 1.5 \times Balance^{i-1}, & \text{if } prob^{i-1} \geq 0.5 \\ 0.6 \times Balance^{i-1}, & \text{if } prob^{i-1} < 0.5 \end{cases}$$

4. Plot the time trajectories (in gamble iterations) of the wealth per gambler and calculate how many gamblers have **more** than their initial 100$. What happens when you increase **N** to 1,000 (Figure 2 in the original paper, except the red and blue line)? How many gamblers with more than 100$ do you have at t=1,000?

5. Plot the time trajectories of the **mean** and **median** wealth at every time point from 0 to 1,000. Try both linear and logarithmic scaler for the wealth axis. What are your observations?

6. Repeat steps 3-5 with **N**=10 and **w**=1,000 with the following logic:

$$Balance^i = \begin{cases} Balance^{i-1} + 50, & \text{if } prob^{i-1} \geq 0.5 \\ Balance^{i-1} - 40, & \text{if } prob^{i-1} < 0.5 \end{cases}$$

   How are the results different than the previous logic?