Data Science at Clarion – Practical Exercise (Target less than 2.5h)

Note: We are not looking for perfectly executable code and rather would like to get a feel how you tackle the model building task and structuring your code for readability, maintainability and to a lesser extent computational overhead, as well as any comments you feel is helpful.

The Data Science team has just built a scraper that gobbled a lot of posts from an Instagram user's feed and associated activity. This user create posts once a month and comments on other posts on some specific days.

After a quick chat with your teammates, everybody thought it would be a good idea to model their social media activity.

1/ Can you write self-contained python file(s), with object-oriented classes to promote reusability, to pass to your teammates to:

(a) parse the json script (below) from the scraper

(b) store the number of (i) posts and (ii)comments on a daily basis across the time period in a python class

(c) calculate the sum of posts and comments on a daily basis

(d) calculate the aggregate number of posts and comments on a monthly basis

(e) store the monthly totals for the whole period for (i) posts and (ii) comments in a csv file

{ “periodStart”: “15/02/11”,

“periodEnd”: “34/08/21”,

“monthlyPostingDay”: 11,

“comments” : [[“2/3/21”, “Justin Bieber”, 5], [“5/4/21”, “Lady Gaga”, 6],], [“5/4/21”, “Snoop Dog”, 2] , [“13/5/21”,”Justin Bieber", 3]]

}

2/ AI Agent Chatbot Exercise

Can you write self-contained python file(s) to perform the following:

a/ devise an internal data structure and write code to store the types of actions that the chatbot can do, as shown below:

- “Extend the lease for Drake & Morgan Limited by 5 years”
- “Sell the Ground & Lower Ground unit for £2.5 million in December 2026.”
- “Change the area of the Stott & May Professional Search Limited to 4200 sqft starting from July 2021”

- “Change the ERV of Interquest Group PLC to 100K”
- "What is the rent for the tenant on the third floor?

b/ write code to create a loop for user input-> language processing -> reply generation -> output to user, in particular, the natural language processing to recognise one of the above actions from any user command in the input

c/ upgrade the processing of natural language to cope with many potential matches (ambiguous user input that potentially matches to more than one command) and request clarification from the user, and also in the case of no match, ask the user to give a command in the available list

d/ describe in words what changes or future features you would create to handle different “surface form variations”, that is different valid english sentences that the user can write to ask for the above commands.