ML Data Engineer – Home task

Scenario:

Our senior data scientist has been working on a new model, and has come up with a table format that fits the plans for the preparation of a few features.

Your task: *(~3h)*

* Prepare a Spark job (pySpark preferred) to consume, transform, join, and aggregate flattened tables into a feature dataframe as designed by the data scientist.
* Prepare the Airflow DAG that will automatically run this spark job on a daily basis, alerting us on failed steps with a strategy for robustness (time permitting).

Tools:

You are free to use any technology, external package, or framework you are comfortable with. The only mandatory tools are (py)Spark and Airflow.

Data:

In this exercise folder, you will find a “data” directory with a parquet file per flattened table. This is real data that has been reduced and anonymized.

* Sessions: Holds aggregated information about a full session
  + session\_ts: timestamp of the start of the given session, in ms
  + session\_uuid: unique ID for the session in Glassbox
  + struggle\_score\_types: an array of dictionaries representing various types of struggles and their aggregated values throughout the session.
  + uabrowser\_group: Family of browser used for the session (nativeApp for mobile)
  + uaos: operating system used for the session
  + vendor
  + uadevicetype: type of device
  + duration: duration of the session in ms
  + num\_of\_pages
  + unique\_num\_of\_pages
  + struggle\_score: proprietary score assigned to the session based on our internal algorithm
* page\_loads: hold page-oriented information about pages that were loaded during a given session
  + hit\_ts: timestamp of the page load in ms
  + session\_ts: timestamp of the start of the given session, in ms
  + session\_uuid: unique ID for the session in Glassbox
  + page\_uuid: unique ID for the page in Glassbox
  + hit\_uuid: unique ID for the hit in Glassbox
  + navigation\_type: type of navigation that brought to this page-load (e.g. back/forward, regular navigation, external source, etc.)
  + struggle\_score: struggle score for the session **up to this page-load**
  + page\_load\_time: time in ms that the page took to load.
* user\_actions: Holds information about specific “actions” (predefined list) performed during a session
  + session\_ts: timestamp of the start of the given session, in ms
  + session\_uuid: unique ID for the session in Glassbox
  + hit\_uuid: unique ID for the hit in Glassbox
  + page\_uuid: unique ID for the page in Glassbox
  + sequence: order of this specific action/hit within the session (e.g. first comes action number 1, then 2, etc.)
  + action\_external\_id: some external ID for the action when available
  + action\_id: some internal ID for the action when available
  + action\_natural\_name: name of the element on which the action has taken place
  + action\_visual\_name: visual name of the element on which the action has taken place
  + client\_action: name of the action that triggered a row in this table
  + struggle\_score: struggle score for the session up to this action.
* client\_flows: Holds information about “flows” that were defined by the customer during a given session (if any).
  + start\_server\_timestamp: timestamp of when the flow started
  + flow\_id: internal ID for the flow
  + flow\_uuid: unique Glassbox identifier for the flow
  + successful: whether the flow successfully ended (reached the last step)
  + session\_uuid: unique ID for the session in glassbox
  + flow\_name: verbose name for the flow
  + click\_count: number of clicks during the flow
  + click\_repetition\_count: number of repeated clicks during the flow
  + field\_change\_count: number of field changes that happened during the flow (in a webform for example)
  + field\_change\_repetition: number of repeated field changes (back and forth) during the flow
  + Duration: duration to complete the full flow
  + Step\_hit\_count: number of steps in the flow
  + Step\_repetition\_count: number of repeated steps during that flow
  + Close\_reason: Reason the flow was closed on our end (whether it was finished or not)
  + Closing\_hit\_page\_uuid: page uuid of the final page for this flow.

*Note: You won’t need all of the data for this task.*

Results wanted:

For this task, you are required to write a spark job that will apply the following transformation and save the end result as a DataFrame in which each row represents information about a given session:

* Struggle\_score\_types: There is a closed set of struggle types. We would like to “dummify” them, and have an extra column per struggle type, with the relevant value attached in the row.
* Number of client\_flows, of successful client\_flows, and of unsuccessful ones within the session, with aggregate count of clicks, repeated clicks, field change, repeated field change, and most common failure “close\_reason”.
* Number of page loads in the session, and page\_uuid of the page with most significant struggle\_score change (increase/decrease)

In addition, we would like to see a Airflow DAG configuration that will allow running this spark job with AWS EMR, and alert us on key-steps failures.

Deliverable:

Please send us a ZIPPED folder with the spark job(s) and the airflow DAG you’ve written.

Feel free to add any document for explanations, or anything else that you’ve needed, used, or just think can be valuable for us when assessing the task.

Some notes & success criteria:

1. While we’ll be looking at the valid functionality of the transformations required, it is understood that unfamiliarity with the data & some of the tools might lead to less-than-perfect work. That is OK. We will also be putting a lot of attention on how well structured is the task, how segmented are the main computational components, and how is the code delivered. Moreover, one important aspect here will be the orchestration with the DAG.
2. Feel free to explore the data and come up with additional (optional!) transformations – if you feel they will also help us understand your thought process and skills
3. Do not burden yourself with airflow configurations for AWS or EMR. We mainly want to see how you define the steps, configure the DAG, and maybe think about robustness strategies (optional, time permitting).
4. Feel free to reach out to Adir (052-8678202) with any question on this task.