Clickstream Data Processing

Sreejith Sreekumar

December 10, 2021

Contents

1 Notebooks & Code Overview

- 1.reading.ipynb: This notebook shows how the gzipped clickstream data is read and stored in a relational way for further processing and analysis.

  Three tables(in a mysql/mariadb database) are created during the process of parsing data.

  - events: Every row indicates the attributes associated with a click event. A description of the event table will look as follows.

```
MariaDB [apptio]> desc events;
+--------------------+----------+------+-----+---------+-------+
| Field              | Type     | Null | Key | Default | Extra |
+--------------------+----------+------+-----+---------+-------+
| fromCache          | longtext | YES  |     | NULL    |       |
| documentReferer    | longtext | YES  |     | NULL    |       |
| tabId              | longtext | YES  |     | NULL    |       |
| url                | longtext | YES  |     | NULL    |       |
| type               | longtext | YES  |     | NULL    |       |
| windowName         | longtext | YES  |     | NULL    |       |
| frameId            | longtext | YES  |     | NULL    |       |
| requestHeaders     | longtext | YES  |     | NULL    |       |
| requestId          | longtext | YES  |     | NULL    |       |
| timeStamp          | longtext | YES  |     | NULL    |       |
| redirectUrl        | longtext | YES  |     | NULL    |       |
| parentFrameId      | longtext | YES  |     | NULL    |       |
| ip                 | longtext | YES  |     | NULL    |       |
| windowTitle        | longtext | YES  |     | NULL    |       |
| responseHeaders    | longtext | YES  |     | NULL    |       |
| user_guid          | longtext | YES  |     | NULL    |       |
| partner_id         | longtext | YES  |     | NULL    |       |
| software_id        | longtext | YES  |     | NULL    |       |
| request_datetime   | longtext | YES  |     | NULL    |       |
| openerTabId        | longtext | YES  |     | NULL    |       |
| mainFrameRequestId | longtext | YES  |     | NULL    |       |
| navigationId       | longtext | YES  |     | NULL    |       |
| eventId            | longtext | YES  |     | NULL    |       |
| statusCode         | longtext | YES  |     | NULL    |       |
| x_forwarded_for    | longtext | YES  |     | NULL    |       |
+--------------------+----------+------+-----+---------+-------+
25 rows in set (0.001 sec)
```

  - users: Information about the users to whom the ads were served. The same user can have same or a different ad served multiple times.

```
MariaDB [apptio]> desc users;
+-----------------+----------+------+-----+---------+-------+
| Field           | Type     | Null | Key | Default | Extra |
+-----------------+----------+------+-----+---------+-------+
| user_agent      | longtext | YES  |     | NULL    |       |
| x_forwarded_for | longtext | YES  |     | NULL    |       |
+-----------------+----------+------+-----+---------+-------+
2 rows in set (0.001 sec)

MariaDB [apptio]>
```

– status: Slightly less important table. Shows the description of various statuses (their codes and what they mean) during a click event.

```
MariaDB [apptio]> desc status;
+------------+----------+------+-----+---------+-------+
| Field      | Type     | Null | Key | Default | Extra |
+------------+----------+------+-----+---------+-------+
| statusCode | longtext | YES  |     | NULL    |       |
| statusLine | longtext | YES  |     | NULL    |       |
| error      | longtext | YES  |     | NULL    |       |
+------------+----------+------+-----+---------+-------+
3 rows in set (0.000 sec)
```

```
MariaDB [apptio]> select * from status limit 10;
+------------+-----------------------------------------------+-------+
| statusCode | statusLine                                    | error |
+------------+-----------------------------------------------+-------+
| 200        | HTTP/1.1 200 OK                               | NULL  |
| 200        | HTTP/1.1 200                                  | NULL  |
| 307        | HTTP/1.1 307 Internal Redirect                | NULL  |
| 503        | HTTP/1.1 503 Service Temporarily Unavailable  | NULL  |
| 302        | HTTP/1.1 302 Moved Temporarily                | NULL  |
| NULL       | NULL                                          | NULL  |
| 200        | HTTP/1.1 200 OK                               | NULL  |
| 200        | HTTP/1.1 200                                  | NULL  |
| 302        | HTTP/1.1 302                                  | NULL  |
| 204        | HTTP/1.1 204                                  | NULL  |
+------------+-----------------------------------------------+-------+
10 rows in set (0.000 sec)
```

- 2.user-cleanup.ipynb

  This notebook takes the user table as input and builds a profile for every user to whom the ads were served.

```
MariaDB [apptio]> select * from users_clean limit 10;
+-----------------+---------+-----------+-------------------+--------+--------------+--------------+--------------+-----------+-----------+-------+
| x_forwarded_for | Browser | OS        | OS_Version_String | Device | Device_Family | Device_Brand | Device_Model | Is Mobile | Is Tablet | Is PC |
+-----------------+---------+-----------+-------------------+--------+--------------+--------------+--------------+-----------+-----------+-------+
| 69.207.104.248  | Chrome  | Windows   | XP                | Other  | Other        | NULL         | NULL         | 0         | 0         | 1     |
| 99.37.4.61      | Chrome  | Chrome OS | 7647.84.0         | Other  | Other        | NULL         | NULL         | 0         | 0         | 1     |
| 68.205.188.176  | Chrome  | Windows   | 10                | Other  | Other        | NULL         | NULL         | 0         | 0         | 1     |
| 199.94.70.33    | Chrome  | Windows   | 7                 | Other  | Other        | NULL         | NULL         | 0         | 0         | 1     |
| 50.150.90.52    | Chrome  | Chrome OS | 8350.68.0         | Other  | Other        | NULL         | NULL         | 0         | 0         | 1     |
| 209.140.37.70   | Chrome  | Chrome OS | 8350.68.0         | Other  | Other        | NULL         | NULL         | 0         | 0         | 1     |
| 99.140.185.199  | Chrome  | Chrome OS | 8350.68.0         | Other  | Other        | NULL         | NULL         | 0         | 0         | 1     |
| 73.187.115.11   | Chrome  | Windows   | Vista             | Other  | Other        | NULL         | NULL         | 0         | 0         | 1     |
| 50.165.25.248   | Chrome  | Windows   | XP                | Other  | Other        | NULL         | NULL         | 0         | 0         | 1     |
| 172.98.85.219   | Chrome  | Windows   | 7                 | Other  | Other        | NULL         | NULL         | 0         | 0         | 1     |
+-----------------+---------+-----------+-------------------+--------+--------------+--------------+--------------+-----------+-----------+-------+
10 rows in set (0.002 sec)
```

  Additional attributes such as location(region, city, lat-long), ISP etc. could be extracted from the ip. However since I was unable to find a free API, this hasn't been done.

- 3.event-analysis.ipynb

  This notebook takes the event table as input and analyses the ads served during each click event. For instance, where was the ad referred from? Where was the landing page etc:

timeStamp	accept_language	user_guid	type	parentFrameId	windowName	statusCode	documentReferer	x_forwarded_for	user_agent	request_unixtime	country_code
1.473207e+12	en-US,en;q=0.8	2f8b23ca273de94a51281b0697a126d7	sub_frame	0	None	200	None	69.207.104.248	Mozilla/5.0 (Windows NT 5.1) AppleWebKit/537.3...	1473207176	US
1.473207e+12	en-US,en;q=0.8	2f8b23ca273de94a51281b0697a126d7	main_frame	-1		200	http://www.imdb.com/title/tt0364845/episodes?r...	69.207.104.248	Mozilla/5.0 (Windows NT 5.1) AppleWebKit/537.3...	1473207176	US
1.473207e+12	en-US,en;q=0.8	2f8b23ca273de94a51281b0697a126d7	sub_frame	0	None	200	None	69.207.104.248	Mozilla/5.0 (Windows NT 5.1) AppleWebKit/537.3...	1473207159	US
1.473207e+12	en-US,en;q=0.8	2f8b23ca273de94a51281b0697a126d7	sub_frame	0	None	200	None	69.207.104.248	Mozilla/5.0 (Windows NT 5.1) AppleWebKit/537.3...	1473207159	US
1.473207e+12	en-US,en;q=0.8	2f8b23ca273de94a51281b0697a126d7	main_frame	-1		200	http://www.imdb.com/title/tt0364845/?ref_=nv_sr_1	69.207.104.248	Mozilla/5.0 (Windows NT 5.1) AppleWebKit/537.3...	1473207159	US

Snapshot of the data kept in events table

	from	count
0		1236120
26272	www.google.com	380009
25249	www.facebook.com	269685
21490	www.amazon.com	93385
24731	www.ebay.com	83527
31451	www.reddit.com	81046
35440	www.youtube.com	57912
35335	www.yahoo.com	35391
1580	apps.facebook.com	35306
665	accounts.google.com	27403
28673	www.mangahere.co	24309
24602	www.drudgereport.com	22520
10647	login.yahoo.com	19947
28328	www.linkedin.com	18962
10569	login.live.com	16100

Summary of click counts from a few of the top referrers

2 Questions

2.1 How would you validate the data?

---

1. Make sure that the zip files are not corruped, or check why this has happened if it did. The code takes in the variable *log_directory_root* where a report (failed.txt) will be written if any of the gzip files fail to read. It will also create the file (processed.txt) with a list of filenames that will be processed during the run.

2. Validating Headers: Every json record inside a gzipped text has the columns '*request_keys*' and '*server_request_keys*'. However not all jsons have the same keys in the second level of hierarchy. During the first iteration of parsing, the code goes through all json instances and collects all the available keys (headers). From the files given, 36 columns were identified. These columns are written in the file schema.txt inside the configured log directory.

   During the second iteration of parsing the data is decoded from json and is made to a dataframe, the columns that are missing in the dataframe are found and are attached to it with "None" as its values. This standardizes all the columns for all the data (from all the files).

   Additional Validation

   Ideally, the validation of data could be designed like a pipeline (chain) of conditions where each dataframe goes through, for example:

a) Choose a subset of all the columns discovered in data
b) Choose a date-range for the records from each dataframe
c) Remove any instance where there is the error column is not null

These conditions have to be configurable or could be modified with very less code addition.

Note: Please see line 121 in the function *standardize_data* function in the file utils.py

2.2 How would you normalize the data and make it representative?

The raw data can be split into 2 main parts. An **event** (a click) and a **user** (identified by an IP address - $x\_forwarded\_for$).This redundancy can be reduced by splitting the whole data into two tables: events and users for separate analysis.

The attributes associated with the user comes from the $user\_agent$ field in the request. The number of times a user has clicked an ad could also be added to this table through a groupby of $x\_forwarded\_for$ and $user\_agent$. However, this needs to be done in a map reduce way since the data is too large to be held in RAM and processed.

The event table stores all the attributes associated with a clickevent. This table can be used to understand the details associated with an event such as the referrer, and the landing of a click. In additional, the table provides information on when the click happened ($unix\_timestamp$), when the data reached the data collection point (timeStamp), some details of the page such as frameId, tabId etc., and status of the click. The referrers (long urls) and the detination urls could be cleaned up to derive business insights on customer engagements.

The status table is a relatively small table that contains the description of every status that have appeared during a click event.

Note: These tables have been exported to a csv and has been shared as a link.