Cycle Script:

The script can be found in the following files:

1. Bonikro:

***‘…\Dropbox\Jupyter\Scripts OVH\Cycle\Count\_cycles\_ETL\_v11\_Bonikro.py’***

1. Sadiola:

***‘…\Dropbox\Jupyter\Scripts OVH\Cycle\Count\_cycles\_ETL\_v11\_Sadiola.py’***

1. Definition

The number of cycles is defined as the total number of trips made by a single truck on a single day. A trip is a complete journey from the ores’ deposit to the top of the different pits.

In more practical terms, in order to compute this number, we use GPS sensors to monitor the trucks’ position at different points in time. And for example, if a truck was spotted on pit 1 then on the ores deposit area and then on pit 1 again, that is a cycle, so we add 1 to the cycle variable.

Let’s delve a little more into details and give a general overview of the script:

1. Parameters:

Two main groups of parameters are set up here:

1. Min date and Max date: they are the same so that we can focus on a specific date and check the number of cycles then.
2. API connection information: the client id, the API key, etc;
3. Geofences names: The geofences in the different pits are hard coded and the ores deposit area geofence name is ‘Rom pad’
4. Data fetching and preliminary processing:
5. We gather the raw data from two main sources: SQL databases, and the API. In particular, we fetch all the vehicle Ids and all the Ids of the tracks these vehicles will go through.
6. We clean the data so that no bug may prevent the cycle variable computation.
7. We convert all dates to the local (the mines’) timezone
8. We then focus on subsets of this clean data: i.e. we only consider the trucks, tracks and geofences we want to perform our analysis on.

The *geofence\_to\_consider* variable is to be manually parametrized as we consider different mines:

For Bonikro:

*geofence\_to\_consider = ['Akissi-so ROM pad','La Chapelle ROM pad', 'ROM pad']*

For Sadiola:

*geofence\_to\_consider = ['Pit\_1','Pit\_2','Rom\_Pad','Pit\_3']*

1. And then we resort to the following method:

If a vehicle is spotted on pit1(resp. pit2) and wasn’t on pit2 (resp. pit1) before, we start a cycle. If a cycle is started and the vehicle is spotted on the Rom pad, we add 1 to the cycle variable, as this means that a travel was made (of course the whole trip isn’t finished yet but after the vehicle goes to the rom pad he then returns to a pit which starts a new cycle)

The reason why we use this methodology is that the KPI we are interested in is the number of **‘loaded cycles’** each vehicle goes through.

Example:

* + Deposit > pit > deposit / devrait générer 1 cycle ? Oui
  + Deposit > pit > deposit > pit / devrait générer 2 cycles ? Oui
  + Pit > deposit > pit > deposit / idem 2 cycles ? Oui
  + Prevoir aussi le cas du pit > deposit ou deposit > pit / 1 cycle ? Pit>deposit mais pas l’inverse

Should you have any further question, please refer to the comments on the script. If those comments are enought, please contact:

* Henri Bouxin: [henri.bouxin@eleven-strategy.com](mailto:henri.bouxin@eleven-strategy.com)
* Ismaïl Alaoui Hassani Atlas: ismail.alaoui@eleven-strategy.com