Agranimo Data Challenge

Esperanza Buitrago - Data Engineer

Challenge Overview:

The following challenge consists of 4 parts:

- Querying a NoSQL Database
- Processing data and model coding
- Creating an executable script to test the model correctness and results
- Storing the results in a SQL Database

Challenge Description:

1 - Querying a NoSQL Database

We provide you access to a MongoDB instance containing events created by devices installed in different areas. Each event contains the following fields:

- topic (String): the topic of the event
- packet (Object): information about the packet
- payload (Object): the sensor data
- createdAt (ISODate): date and time of insertion in the Database
- processed (Boolean): a flag that determines if a datapoint has been already processed

The payload of each event contains different fields, based on the device ID (payload.DID).

Here are the descriptions of the most important ones:

- hum1: Humidity of air sensor 1 ([%])
- tem1: Temperature of air sensor 1 ([ºC])
- solr: Solar radiation ([W/m^2])
- smo1 / smo2 / smo3 / smo4: Soil moisture voltage of sensor 1 ([V])
- stm1 / stm2 / stm3 / stm4: Soil temperature of sensor 1 ([ºC])
- TMS: UNIX UTC timestamp (when the event was created) (INT)

To connect to the database, please use the following connection string in your application:

mongodb+srv://admin:AaPqZ4wSeF5lC91t@cluster0.kukot.mongodb.net/agranimo-challenge

You should retrieve events from a device of your choice that contains the information required in the model (see section 2) and at least 1000 events. You are free to retrieve data from whatever device that satisfies the requirement you want, plot the data, explore, and decide which can be reasonable to use to test your model in point 3.

2 - Data processing and model coding

When you are ready with the data that you need from the NoSQL Database, the next step is to create a Python version of the agronomic model provided.
It is a parametrized predictive system used to forecast strawberry powdery mildew disease development. The document is named "*Use of a Real-Time Decision Support System to give accurate timings for fungicide applications*" from the University of Hertfordshire.

Link:

https://uhra.herts.ac.uk/bitstream/handle/2299/22623/Hannah_Wileman_BSPP_Poster.pdf?sequence=1

3 - Test the model correctness and results

Once you are happy with your code, it's time to make it run! Please provide a script that allows to test the correctness of your implementation. The output of the algorithm should be:

- A plot similar to the one presented in the document, that is, a graph showing where hours have accumulated under disease conducive conditions, and when fungicide spray should be applied.
- A list of data points in which the conducive conditions were satisfied, with the following fields:
  - Date and time of the data point (date-time)
  - UTC timestamp of the data point (int)
  - Device DID (str)
  - Temperature Value (float)
  - Humidity Value (float)

- A list of the fungicide applications (if any), with the following data:
  - Date and time of the application (date-time)
  - UTC Timestamp of the application (int)
  - Device DID (str)
  - Time difference between last fungicide application (time delta)

You are allowed to use any library you like.

---

4 - Store results in a SQL Database

The results of the previous step (data points and fungicide application events) should be stored in a PostgreSQL database (provided empty). Create the required tables and insert the results.
The database is accessible using the following connection string:

postgres://vtbzgevk:9p9Qiatj4QOQM2biVZF_3GUXXZBixSpS@tai.db.elephantsql.com:5432/vtbzgevk

Once again, you are free to use whatever client/library you want.

Deadline:

Please email us the code and Github link. The challenge is meant to be finished by **March 15th 2021** at the latest. Feel free to ask questions during the challenge - it will give us an indication of how you think and get a feeling for working together as a team.

Good luck!