INTRODUCTION

Our project consisted in collecting, processing, and analysing quantitative observations about the three variables, Water flow, water temperature and water level, of three rivers in the Bolzano area. In terms of raw data, we found two main sources of information: (the website of historical data) and the (website of new data). Both these sites allowed us to extract data in JSON format. The website of historical data allowed us to download a dataset comprehensive of observations dated until March/April 2019. This source of an historic dataset was extremely useful for us to train a predictive model over a large amount of data. The website of new data instead provides new observations every 20 minutes. Our choice was to take data only every hour because the goal was to make predictions of the variables after an hour or further. This choice was also supported by the evidence of very little, if not null, variance in variables values in a timespan shorter than a single hour. The first step to of our project was to create a database to store observations and predictions. For our purposes we decide to implement a MySQL database since we only needed to store many observations in a small number of columns and the structure suits perfectly into a tabular one. The columns of our database consist in the Timestamp, the values, the season, which was derived from the Timestamp, and the ID of the river. As a matter of fact, an ID was assigned to each river in order not to store the name of the river in each time a new observation comes in. Thus, we created another table containing the three names of the rivers and their corresponding IDs. To create Our database, we decided to exploit the services offered by Amazon. Indeed, the database was created on the cloud through an ec2 instance of AWS. We downloaded docker on this virtual machine in order to use a MySQL image and using it without the need to install it on the machine. Once the database was created and we were able to access it remotely, it was time to ingest data. Firstly, we had to download and update the tables of the database with the historic data, and then we could move to ingesting periodically the new data. Since we could download manually data from the website, and we would do this once, we proceeded to make computations and to process our historic data to make it as fast as possible to update our database tables. To make computations on such a big amount of data faster we used PySpark, which drastically improves performances with respect to standard python. In addition, since executing an insert query for many data is extremely slow, our first step was to convert the historic data into a csv file to be able to perform a ‘load data local infile’ query. This is much faster. Once our database was provided with enough data, we could move to training predictive models. Once our models were trained on the historic data, we moved to setting our new data ingestion and storage system. Firstly, a script running in a forever loop, extracting, and processing the new data, was created. Then the new data, coming in json format, after being processed, would still be transformed in csv format to be uploaded to the database through a ‘load data local infile query’. In this case the volume of the data was such to make the usage of insert still feasible but also the used query is a valid choice. Once our data was ingested, we could use our trained models, one for each variable of each river, to compute predictions to be stored in apposite tables of the database. Lastly, to present our analysis and the results of the predictions, a webapp has been implemented. Thanks to it is possible to observe trends and predictions for all variables and all rivers. In the end, in order to control each step of our pipeline and make it automatic, we decide to implement a pub-sub system that would allow to trigger different processes based on the messages provided at the end of each step. This would also allow us to identify issues quickly.

RESULTS

The final output of the project is a WebApp created with Streamlit an open-source app framework. The WebApp can be deployed by running the WebApp\_FINAL.py, this will create both a local URL and a network URL than can be used to view the App in the browser.

The application is structured as follows: the first thing that pops up is the title: Analysis of Rivers in the Bolzano's Area followed by an interactive map created with folium where it is possible to visualize the rivers. On the right of the page there is a sidebar followed by two checkboxes where it is possible to choose the river and the measure the user is interested about. After the choice has been made the page will reload unless the default choices have been selected. Then, it is possible to click on a checkbox that if activated shows the data frame associated to the river and the chosen variable. Moreover, it will also shown a line plot where it is represented the trend of the chosen variable.