

Capgemini test: forecasting water levels

Stefano Petrucci
petrucci.ste@gmail.com

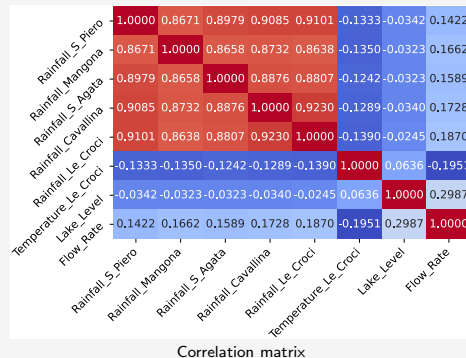
- 1 Software
- 2 Data set
- 3 Strategy
- 4 Results
- 5 Conclusions

Software

- Written in Python and available on [GitHub](#)
- Used standard libraries:
 - Pandas
 - Seaborn
 - Statsmodels
- One class for the data set (with methods for plotting and split data into train/test)
- Some functions for the statistical models
- Everything is documented within the code (it should be Doxygen friendly as well)

Data set

- Chosen *Lake_Bilancino*
- Two target variables:
 - Lake level
 - Flow rate
- Most variables missing before 01/01/2004
⇒ removed ($\sim 8\%$ of the total)
- Missing data not replaced^a

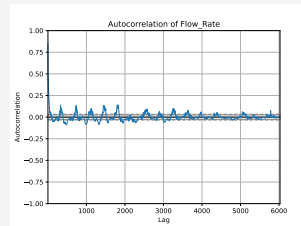


^aApplied an interpolation only when computing the autocorrelation.

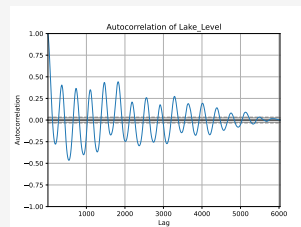
Forecasting strategy

- 1 Simple AutoRegressive (AR) model
- 2 More complex AutoRegressive Integrated Moving Average (ARIMA) model
- 3 Multivariate analysis (not implemented)

Both models used in this project require to setup the lag.
⇒ chosen from autocorrelation plots.



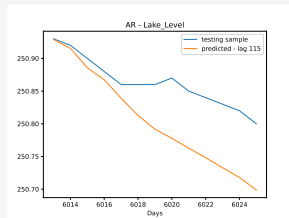
Flow rate - autocorrelation



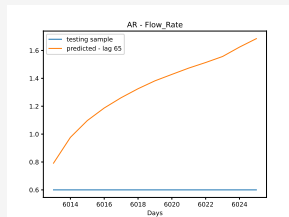
Lake level - autocorrelation

Predictions - 13 samples

AutoRegressive model

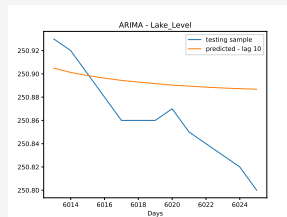


Lake level

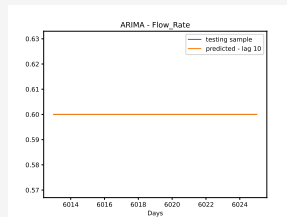


Flow rate

ARIMA model



Lake level



Flow rate

Conclusions

- Implemented a toy script to compute AR and ARIMA algorithms
- Both algorithms showed better performance on `Lake_level`
- The behavior of `Flow_rate` requires additional investigation
Optimizing ARIMA's parameters might improve performance

Thank you for your attention