A software service supporting software quality forecasting

Martí Manzano
Universitat Politècnica de Catalunya
Barcelona, Spain
mmanzano@essi.upc.edu

Claudia Ayala
Universitat Politècnica de Catalunya
Barcelona, Spain
cayala@essi.upc.edu

Cristina Gómez
Universitat Politècnica de Catalunya
Barcelona, Spain
cristina@essi.upc.edu

Lidia López Cuesta
Universitat Politècnica de Catalunya
Barcelona, Spain
llopez@essi.upc.edu

***Abstract*—Software repositories such as source control, defect tracking systems and project management tools, are used to support the progress of software projects. The exploitation of such data with techniques like forecasting is becoming an increasing need in several domains to support decision-making processes. However, although there exist several statistical tools and languages supporting forecasting, there is a lack of friendly approaches that enable practitioners to exploit the advantages of creating and using such models in their dashboard tools. Therefore, we have developed a modular and flexible forecasting service allowing the interconnection with different kinds of databases/data repositories for creating and exploiting forecasting models based on methods like ARIMA or ETS. The service is open source software, has been developed in Java and R and exposes its functionalities through a REST API. Architecture details are provided, along with functionalities' description and an example of its use for software quality forecasting.**

Keywords— forecasting, software service, software quality, software metrics, REST API

I. Introduction

Nowadays, the high availability of data about the software development process from tools like SonarQube or Jenkins is being exploited to improve the software process and product quality through well-known statistical techniques such as machine learning (e.g., anomaly detection, clustering or classification) leading to a business value increment. Besides, to predict and anticipate the future status of the software process or product is becoming an important asset for supporting strategic planning. Several forecasting methods exist in the literature to support this endeavor.

Currently, there exist two main approaches to create and use software quality forecasting models in real world applications: 1) the use of programming languages for statistics such as R [1], that allow to create or reuse some implementations of forecasting methods and 2) the use of specialized software tools such as ForecastPro [2], AUTOBOX [3], or even MS Excel plugins like Forecast X [4], that automate the creation of forecasting models. The first approach is flexible as allows the creation of particularized forecasting models but requires extensive programming and statistical knowledge. The second approach is automatic but lacks of flexibility to particularize the models to the users' contexts or technologies and to integrate the models into the actual users' reporting tools. As a result, this gap makes that the majority of practitioners are not able to exploit the benefits of forecasting in their decision-making processes. In order to deal with this gap, we designed a forecasting service called *qr-forecast*. Its main characteristics are: 1) To enable the automatic creation and ease the use of forecasting models without requiring statistics and programming knowledge; 2) To provide implementations of diverse forecasting methods in order to build the forecasting models; 3) To ease the interconnection to the database/repositories containing the historical data to serve as input to fit the forecasting models; 4) To be easily integrable in tools like dashboards or reporting software.

II. Qr-Forecast Service

In this section, the functionalities and the architecture and implementation details of the *qr-forecast* service are presented.

A. Functionalities

The main functionalities of the service are:

- **ForecastTechniques:** Provides the list of all forecasting methods provided by the service, so that the user can select the ones to be used. (See Table I for the current available methods).
- **ForecastTraining:** Fits forecasting model(s) for a specific set of variables (modeled as univariate time-series). It uses: the selected forecasting method to be executed, the specific set of variables to build the model and the historical data that will be used as a basis to build the model(s). For each input variable, the functionality returns the resulting status of the model's fitting process. The fitted models are stored to be reused later on, hence speeding up the forecast requests' processing time.
- **Forecast:** Given a set of variables, a forecasting method and a forecasting horizon, it returns the lower and upper 80% and 95% prediction intervals along with the forecasted means, for every requested variable. If there exists a saved forecasting model, then it is reused to compute the result, if not, it is created and saved to disk for later reuse.

B. Architecture and Implementation

We decided to expose the expected functionalities as a service in order to reach a high level or integrability with other services/software reporting tools. The architecture is modular and composed of three components, each one with specific goals that allow us to clearly separate concerns. Fig. 1 shows an overview of the architecture and the relations among components.

The *R Scripts*[1] component manages the data gathering process (i.e., the connection with the database/repository) and the execution of the model fitting and forecasting processes. Such processes are based on specific R packages containing the forecasting methods' implementations. The fact that this component directly connects to the database/repository helps to minimize the data transfer loads when executing the model fitting/forecasting processes. This *R Scripts* component was implemented in R and gathers the historical data from an Elasticsearch engine.

The *Forecast library*[2] contains the implementation of the functionalities presented in Section 2 and is in charge of handling exceptions. This component communicates through RServe [5] with the *R Scripts* component and was developed in Java as an importable JAR library.

The *Forecast wrapper*[3] imports the *Forecast library* JAR component and exposes its public functions as a REST API, hence making it reachable to reporting tools or other services. This component was implemented as a Spring deployable WAR. Its use is optional, as the *Forecast library* can be used as a regular JAR library if the REST API is not needed.

The separation of concerns gained with these three components led us to easily handle the evolution and potential changes of the diverse capabilities of the service. For instance, changing the Elasticsearch engine from the *R Scripts* component to any other database or adding new forecasting methods can be handled with reduced effort.

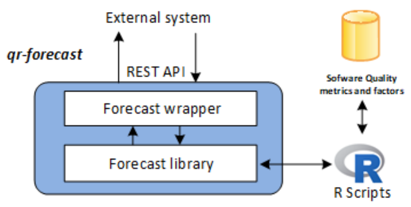

Fig. 1 Architecture of the qr-forecast service

The currently available forecasting methods include very diverse methods such as: statistically sophisticated methods like ARIMA [6], decomposition methods like *Theta* [7] or STL [8]; machine-learning based like Neural Networks [9], hybrid approaches like "forecastHybrid" [10] and additive-model based like "Prophet" [11]. The complete set of available methods along with the R package used in *qr-forecast* are shown in Table I. We decided to include these methods as they are well-known methods in the community, and cover a wide spectrum of method's categories (e.g., (not) seasonal, and with or without trend input data). In terms of computational cost, ensemble methods like "forecastHybrid" are expensive but can produce accurate forecast, as they combine values from several methods. The use of the **ForecastTraining** functionality is adequate for these methods to avoid long waiting times when requesting forecasts through the **Forecast** functionality. On the other hand, methods like "ETS" [12] are very fast, at the cost of being limited for forecasting data with no clear seasonal pattern or trend. Furthermore, some of these forecasting methods performed well in the past M3 forecasting competition [13]. Including this heterogeneous set of forecasting methods fosters the suitability of the *qr-forecast* service for processing different kinds of data and for being computationally afforded by most of their potential users.

Table I. Forecasting methods included in the qr-forecast service

Method Name	**R Package**
Arima [6]	forecast [14]
Arima (forcing seasonal models) [6]	
Theta [7]	
ETS (Exponential Smoothing State Space Model) [7]	
ETS (forcing damped models) [7]	
Bagged ETS [7]	
STL (Seasonal Decomposition of Time Series by Loess) [8]	
Neural Network [9]	
Theta [7]	forecastHybrid [10]
Hybrid [10]	
Prophet [11]	prophet [11]

III. Examples

A specific example of the use of the *qr-forecast* service is the Q-Rapids Dashboard, developed in the context of the Q-Rapids European project [15]. This dashboard provides an easy and attractive informative interface to show textual and graphical information about the quality aspects of the software processes and products developed in software companies, based on software metrics, factors and strategic indicators. Specifically, the dashboard presents the current and the historical assessment of the software quality aspects and uses the *qr-forecast* service for forecasting such aspects.

Fig. 2 (left) shows the forecasted means for three factors of a specific software product: "Activities Completion", "Known Remaining Defects" and "Product Stability". The purpose of these factors is having a higher-level view of its comprising software quality metrics. For instance, the factor "Activities Completion" is computed using the metrics "Development Task Completion" and "Specification Task Completion". Using the *qr-forecast*'s functionality, the forecasted means of these factors are presented at Fig. 2 (right). Further information of the qr-forecast service functionality integrated into the Q-Rapids dashboard can be consulted at this video[4]. Deployment instructions are also available[5].

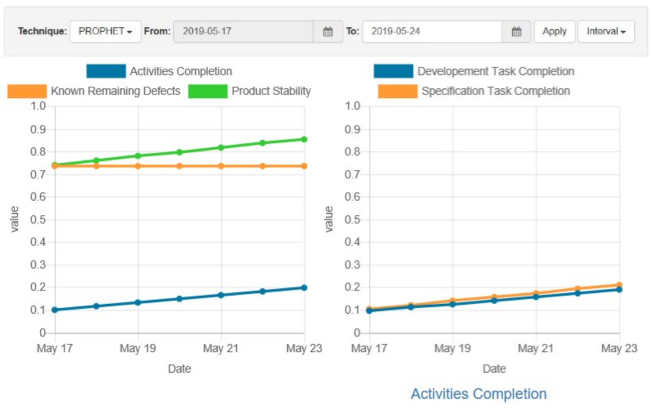
Technique: PROPHET From: 2019-05-17 To: 2019-05-24 Apply Interval

Fig. 2 Forecasted means of software quality aspects using qr-forecast

[1]R Scripts - https://git.io/fj8Nw
[2] Forecast Library - https://git.io/fj8Ni
[3] Forecast Wrapper - https://git.io/fj8NM
[4] Video demonstration - https://youtu.be/uVumzekT4-s
[5] Deployment instructions - https://git.io/fj8ND

Another example showing forecasted means of some particular project metrics is shown in Fig. 3 (row 1 and row 2). In this case, the user has selected a forecasting horizon of almost two months, and Arima (forcing seasonal models) as a forecasting method. It's noticeable how the returned values follow a seasonal pattern, as the method used is forcing the search and selection of seasonal ARIMA models. The forecasts shown in Fig. 3 refer to the following software related metrics:

- The developer's "ability" or performance fixing software bugs ("Bug Correction Throughput", Fig. 3 row 1-left).
- The communication throughput between developers and testers ("Developer Tester Communication", Fig. 3 row 1-right).
- The amount of feedback provided by end users of the monitored software ("End User Feedback", Fig. 3 row 2-left).
- The percentage of commits related to core components ("Core Component Commits", Fig. 3 row 2-right).

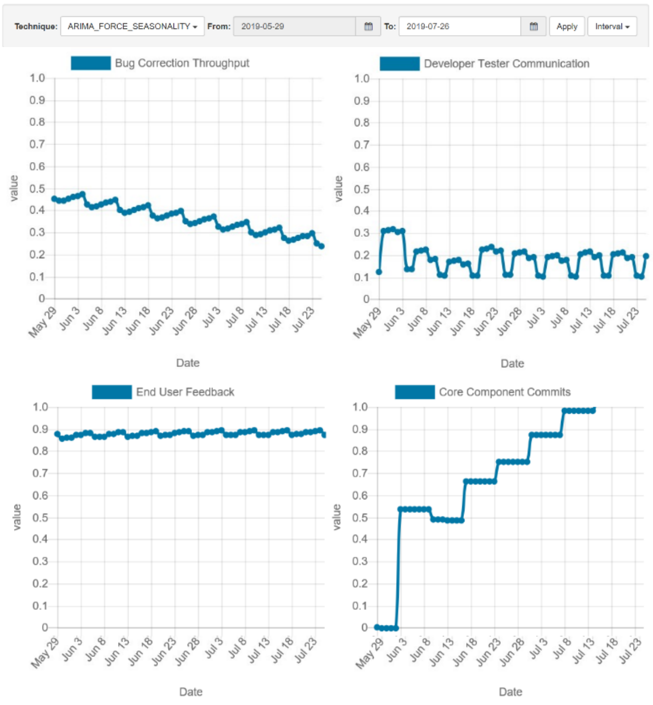
Technique: ARIMA_FORCE_SEASONALITY From: 2019-05-29 To: 2019-07-26

Fig. 3 Forecasted means of project metrics using qr-forecast and seasonal models

IV. Conclusion and Future Work

We presented the *qr-forecast* service that aims to support practitioners without specific statistical nor programming knowledge to use and integrate the decision-support benefits of forecasting models into their existing software reporting tools. In addition, we have shown a specific application of the *qr-forecast* service into an existing dashboard, showing forecasts for several software quality aspects like *Product Stability* or *Bug Correction Throughput*. The service has been already deployed in four software related companies and positive feedback has been gathered. As future work, we plan to extend the available set of forecasting methods and to include functionalities for automatic recommendation of the most suitable forecasting method according to the nature of the input data from the database/repository and in terms of well-known forecasting accuracy measures like sMAPE [16] or MASE [17].

Acknowledgments

This work is a result of the Q-Rapids project, which has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement N° 732253.

References

[1] R. Development Core Team, "R: A Language and Environment for Statistical Computing," *Vienna Austria R Foundation for Statistical Computing*. 2008.

[2] R. L. Goodrich, "The Forecast Pro methodology," *Int. J. Forecast.*, vol. 16, no. 4, pp. 533–535, Oct. 2000.

[3] D. Reilly, "The AUTOBOX system," *Int. J. Forecast.*, vol. 16, no. 4, pp. 531–533, Oct. 2000.

[4] J. H. Wilson and B. Keating, *Forecasting and Predictive Analytics with Forecast X*. 2018.

[5] S. Urbanek. Rserve: Binary R server. R package version 1.7-3.1. https://CRAN.R-project.org/package=Rserve. 2019.

[6] P. Newbold, "ARIMA model building and the time series analysis approach to forecasting," *J. Forecast.*, vol. 2, no. 1, pp. 23–35, Jan. 1983.

[7] V. Assimakopoulos and K. Nikolopoulos, "The theta model: a decomposition approach to forecasting," *Int. J. Forecast.*, vol. 16, no. 4, pp. 521–530, Oct. 2000.

[8] R. B. Cleveland, W. S. Cleveland, J. E. McRae, and I. Terpenning, "STL: A Seasonal-Trend Decomposition Procedure Based on Loess (with Discussion)," *J. Off. Stat.*, vol. 6, pp. 3–73, 1990.

[9] S. Haykin, *Neural Networks: A Comprehensive Foundation (3rd Edition)*. 1999.

[10] D. Shaub and P. Ellis. forecastHybrid: Convenient Functions for Ensemble Time Series Forecasts. R package version 4.2.17. https://CRAN.R-project.org/package=forecastHybrid. 2019.

[11] S. J. Taylor, M. Park, U. States, B. Letham, M. Park, and U. States, "Forecasting at scale at Facebook," pp. 1–25, Sep. 2017.

[12] R. J. Hyndman, A. B. Koehler, R. D. Snyder, and S. Grose, "A state space framework for automatic forecasting using exponential smoothing methods," *Int. J. Forecast.*, vol. 18, no. 3, pp. 439–454, Jul. 2002.

[13] S. Makridakis and M. Hibon, "The M3-Competition: results, conclusions and implications," *Int. J. Forecast.*, vol. 16, no. 4, pp. 451–476, Oct. 2000.

[14] R. J. Hyndman and Y. Khandakar, "Automatic Time Series Forecasting: The forecast Package for R," *J. Stat. Softw.*, vol. 27, no. 3, 2008.

[15] L. López *et al.*, "Q-Rapids Tool Prototype: Supporting Decision-Makers in Managing Quality in Rapid Software Development," CAiSE Forum. 2018.

[16] S. Makridakis, "Accuracy measures: theoretical and practical concerns," *Int. J. Forecast.*, vol. 9, no. 4, pp. 527–529, Dec. 1993.

[17] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *Int. J. Forecast.*, vol. 22, no. 4, pp. 679–688, Oct. 2006.