# A Data Mining Service for Non-Programmers

Artur Pedroso, Bruno Leonel Lopes, Jaime Correia, Filipe Araujo, Jorge Cardoso and Rui Pedro Paiva
*CISUC, Dept. of Informatics Engineering, University of Coimbra, Portugal*

Keywords: Data Science, Data Mining, Machine Learning, Microservices.

Abstract: With the emergence of Big Data, the scarcity of data scientists to analyse all the data being produced in different domains became evident. To train new data scientists faster, web applications providing data science practices without requiring programming skills can be a great help. However, some available web applications lack in providing good data mining practices, specially for assessment and selection of models. Thus, in this paper we describe a system, currently under development, that will provide the construction of data mining processes enforcing good data mining practices. The system will be available through a web UI and will follow a microservices architecture that is still being designed and tested. Preliminary usability tests, were conducted with two groups of users to evaluate the envisioned concept for the creation of data mining processes. In these tests we observed a general high level of user satisfaction. To assess the performance of the current system design, we have done tests in a public cloud where we observed interesting results that will guide us in new directions.

## 1 INTRODUCTION

In a broad view, data mining is the process of discovering interesting patterns and knowledge from large amounts of data (Han et al., 2011). However, for the correct application of data mining processes and also for the evolution of the field, competent data scientists are required, a resource in high demand these days (Henke et al., 2016; Miller and Hughes, 2017). To fill such demand, more data scientists need to be trained, which requires time due to the diversity of disciplines to learn(Cao, 2017). Thus, by abstracting somehow programming languages from the data scientist's path, we might reduce the necessary time to train them.

Having the data mining process in mind, we decided to create a system that allows users to build workflows representing the data mining process. It will be available through a web UI providing good usability heuristics (Nielsen, 1994), and guiding the user in the creation of data mining processes without requiring programming skills.

The user will be able to create experiments based on workflows composed by sequential data mining tasks. These tasks will allow data insertion, preprocessing, feature selection, model creation and model evaluation. Some tasks will include parameters that can be used in grid search along with nested cross validation enforcing good model assessment and selection practices (Cawley and Talbot, 2010).

To evaluate the envisioned system, we created a first prototype and conducted usability tests using a group of users familiar with data mining frameworks, and another group of users without experience with related tools, though having a background in statistics, whom can also benefit with our software. We observed an overall positive user satisfaction with both groups.

To evaluate the impact of the current microservices architecture in the performance of the system, we deployed it in a public cloud and realised tests using datasets with different sizes. The results are interesting and an incentive to guide us in new directions.

The remaining document is organised as follows. In Section 2, we analyse related research and applications. In Section 3, we present an overview of the envisioned user interface and the system architecture. In Section 4, we present preliminary experiments done and the respective results. Finally, in section 5 we draw the main conclusions of this work and point out future research directions.

## 2 RELATED WORK

Some applications in production already provide the creation of data mining processes without requiring



Pedroso, A., Lopes, B., Correia, J., Araujo, F., Cardoso, J. and Paiva, R.
A Data Mining Service for Non-Programmers.
DOI: 10.5220/0007226003400346
In *Proceedings of the 10th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2018) - Volume 1: KDIR*, pages 340-346
ISBN: 978-989-758-330-8

users to hold programming skills.

Azure Machine Learning Studio[1] is a publicly available software-as-a-service solution that allows its users to create data mining workflows by dragging blocks that represent data mining tasks into a working area.

RapidMiner Studio[2] and Orange[3] provide the same concept as Azure Machine Learning Studio for the creation of data mining processes. However, these are local solutions.

The three previous tools require users to create complex workflows to assess the performance of models including different tasks and parameters. Cross validation in Azure and Orange is just applied to the model creation phase and does not include prior operations like feature selection which is a bad practice for estimating the model's performance (Cawley and Talbot, 2010).

H2O Flow[4] offers a fully distributed in-memory ML open source platform that can be deployed in clusters. The platform can be used from a web UI that gives the possibility to apply machine learning (ML) in a sequence of steps without requiring users to have programming skills. However the user is limited to uploading datasets and building models using the provided ML algorithms. Other data mining tasks (e.g., feature selection) are not available.

Weka[5] is a local solution that enables the application of data mining tasks to datasets. It can become complex to build data mining processes composed of multiple tasks and parameters.

(Kranjc et al., 2017) and (Medvedev et al., 2017) are both research projects to provide cloud solutions for the creation of data mining processes through a web UI employing similar concepts (drag-and-drop) as Azure, RapidMiner and Orange. Both systems do not solve the problems exposed by the previous systems.

Besides RapidMiner, none of the above applications provide the insertion of a data mining experiment in a (nested) cross validation loop. It is also common to see in some of the previous systems that cross validation is applied only to the final model without including prior tasks, such as feature selection in the loop, which is a bad practice (Hastie et al., 2001; Cawley and Talbot, 2010).

Adding to the problems abovementioned, none of these systems guide the user in the data mining process.

[1]https://studio.azureml.net/

[2]https://rapidminer.com/products/studio/

[3]https://orange.biolab.si/

[4]https://www.h2o.ai/

[5]https://www.cs.waikato.ac.nz/ml/weka/

Having in mind these limitations, the following requirements will be addressed in our system:

- Provide a web UI with good usability for non-programmers to execute data mining tasks.
- Guide the user in the creation of a data mining process.
- Provide different data preprocessing methods, feature selection and machine learning algorithms.
- Allow the creation of data mining experiments including different tasks, features and parameters for evaluation and selection of the best model (the one with "best" features and parameters). Here, good data mining practices will be guaranteed, e.g., nested cross validation.
- Provide an application accessible from the cloud where data mining workflows can be left running and accessed later.
- Provide a scalable system to support a large numbers of simultaneous users.

# 3 DESIGN AND IMPLEMENTATION

In this section we proceed to present the user interface that was used in the usability tests and the architecture as it is at the moment.

## 3.1 User Interface

The UI is divided in two key areas, as we can see in Figure 1. The darker area on the left includes operations for creation and retrieval of workflows and datasets. It also enables the execution and interruption of workflows that are built on the right area.


Figure 1: User interface - showing a dataset insertion task and the option to insert a validation procedure after clicking the *plus* button.

The area on the right is where the user builds the workflow inserting tasks that compose a data mining process.

To guide the user in the data mining process, the tasks are available for insertion according to a predefined grammar that is presented next in EBNF notation:

```
start = dataset_input val_procedure
val_procedure = ((assessment_method_1
    {(preprocessing_1 | feature_selection_1)}
    create_model) | ( assessment_method_2
    (preprocessing_2 | feature_selection_2 |
    create_model)))
preprocessing_1 = "preprocessing_method"
    {val_procedure_1}
feature_selection_1 = "feature_selection_algorithm"
    {val_procedure_1}
preprocessing_2 = "preprocessing_method"
    {val_procedure_2}
feature_selection_2 = "feature_selection_algorithm"
    {val_procedure_2}
create_model = "machine_learning_algorithm"
    "eval_metrics"
assessment_method_1 = "cross_validation" |
    "hold_out" | "t_v_t"
assessment_method_2 = "use_entire_data"
dataset_input = "dataset_input"
val_procedure_1 = (preprocessing_1 |
    feature_selection_1)
val_procedure_2 = (preprocessing_2 |
    feature_selection_2 | create_model)
```

In this grammar, the terminals are between double quotes. These are specific tasks to be executed and might have different representations. For example, "preprocessing_method" might be a z-score normalisation or a min-max normalisation task.

In Figures 1 and 2 we show that when the user clicks the *plus* button to add a new task, depending on the current state of the workflow, s/he only sees the tasks according to the previous grammar.


Figure 2: UI - Showing cross validation task (a validation procedure task) and the tasks that can be used after.

In summary, the six types of task that can be used in the workflow are the following:

- **Dataset Input:** a unique task where the user specifies the dataset to use. S/he can also choose to remove features during this step.
- **Validation Procedure:** contains tasks that specify a method to be used in the creation of the data mining process. The user can define if the next tasks should be included in an assessment/selection process (e.g., cross validation), or if the tasks should be created using all data.
- **Preprocessing:** contains tasks that apply transformations to attribute values (e.g., z-score normalization).
- **Feature Selection:** contains tasks to assess the relevance of features for selection (e.g., Relieff).
- **Model Creation:** contains tasks for the creation of models using different algorithms (e.g., Support Vector Machine (SVM)).
- **Model Evaluation:** contains tasks that specify the metrics to use for performance evaluation (e.g., recall and precision).

## 3.2 Architecture

The previous UI is part of a microservices architecture that we illustrate in Figure 3.

In this architecture, a user can access the UI through the UI Service that provides a web application written in ReactJS, from which further requests are done to our API Gateway that redirects the requests to different services accordingly.

The Tasks Service returns representations of data mining tasks that can be used to compose the sequential data mining workflow.

The User Service enables users to login with a username and a password and holds information related to users.

The Templates Service contains predefined templates of data mining workflows useful for certain data and business domains.

The Datasets Service stores uploaded datasets in a central file system (Network File System (NFS)) and also returns data from the NFS according to users' requests. The MongoDB in Datasets Service is used to store metadata related to uploaded datasets.

Then, we have the Workflows Service that translates sequential workflows sent by users to a representation that is understandable by Netflix Conductor[6]. The new representation is sent to the Conductor Service that employs Netflix Conductor, and becomes available to be processed by different Data Science services/workers. The Workflows Service is also contacted to return the status of workflows sent by users.

By using the Netflix Conductor technology we can organise the tasks in a certain sequence and the Data Science services can pull the scheduled tasks and work on them in parallel and independently, following a competing consumers pattern (Hohpe and

[6]https://netflix.github.io/conductor/

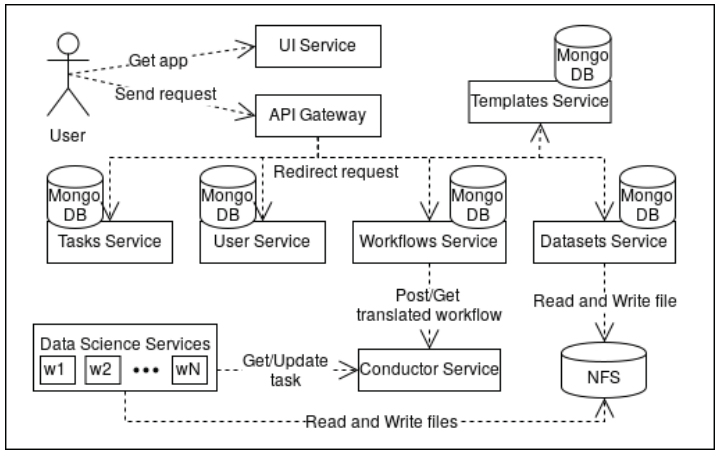


Figure 3: Current system's architecture.

Woolf, 2003). Netflix Conductor allows that tasks appearing ahead in a workflow's path are executed just after the prior tasks have been executed.

The Data Science Services are multiple fine grained services/workers that work on specific data science tasks pulled from the Conductor Service. These Data Science Services share files (e.g., datasets, models) between them by writing and reading to/from the NFS.

The communications between all the services presented in the architecture are performed using the HTTP protocol, mainly through REST APIs. All the services can be scaled out independently.

To better understand how individual data science tasks are processed in the system, in Figure 4 we present an example of a translation from a sequential workflow sent by the user (on the left), to its representation in Netflix Conductor (on the right). This translation abstracts users from the creation of complex workflows, which is an advantage over other systems such as Azure ML Studio, as abovementioned.

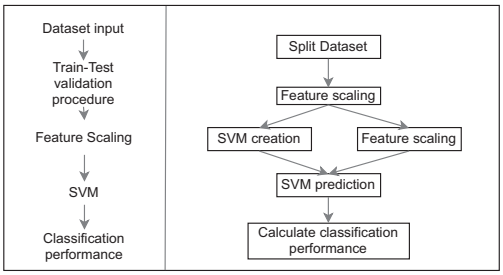


Figure 4: Example of a data mining workflow translation.

The sequential workflow sent by the user contains the location of the dataset to use, the procedure to evaluate the process (hold out / train-test method), a feature scaling task that is followed by a model creation task using the SVM algorithm, and finally there is a task to show the classification performance of the produced model.

Upon receiving the workflow, the Workflows Service translates it to the Netflix Conductor representation. In the new representation, the flow starts with a Split Dataset task (split original data into training and test sets), followed by a feature scaling task (applied to the training set). Then, an SVM creation task (applied to the processed training set) and a feature scaling task (applied to the testing set and using info from the previous feature scaling task) can be handled in parallel. The SVM prediction task (applied to the processed test set and using the model created before) appears next, and finally, we have a task to compute the classification performance of the model. It is normal that tasks appearing ahead in the workflow use data produced in preceding tasks.

# 4 EXPERIMENTS

In this section we present the tests done with a first prototype of the system deployed on a cluster in Google Kubernetes Engine[7]. For that we used 4 instances with 2 vCPUs and 7.5GB of RAM each.

## 4.1 Usability Tests

### 4.1.1 Setup

The usability tests provided a crucial role in evaluating the prototype and validating the paradigm of

[7]https://cloud.google.com/kubernetes-engine/

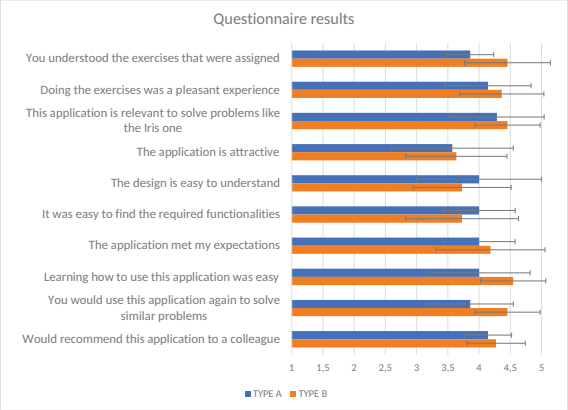


Figure 5: Average and standard deviation of the users' responses.

constructing data mining processes using sequential tasks. The tests consisted in having the users execute a few exercises using the interface and getting their feedback. This feedback was then used to evaluate the users' experience, the usability of the interface and the value that was provided to them, hence validating the concept.

We divided the users in two types:

- **Type A:** Users with no experience with data mining systems and no knowledge in data mining or programming languages (8 users).
- **Type B:** Users with experience in data mining systems (mainly Orange), with knowledge in data mining but without programming skills (11 users).

The usability tests started with a quick overview of the platform and its functionalities, which took less than 3 minutes. After this introduction and question answering, we gave the users a script with a few exercises estimated to be solved in less than 20 minutes. In the end we gave a questionnaire that the users had to fill about their experience, and their thoughts on the relevance of the system.

To keep the tests simple we decided to ask the users to make six exercises using the iris flower dataset (Anderson, 1936).

The exercises were simple and intertwined, making the user have a feeling of progress during their execution.

Briefly, the exercises that we asked them to perform were the following:

1. To scale the attributes of the dataset between the values 0 and 1.
2. To create an SVM model and to use the hold-out procedure to assess the model performance. Also verify the accuracy and f-measure of the produced model.
3. Same exercise as before, however including a feature scaling operation before model creation. This was conducted to verify whether the user was aware that tasks could be created and removed in the middle of a workflow previously created.
4. To perform feature selection using the Relieff algorithm and different numbers of features to see which attributes would have the most predictive capabilities.
5. To build an SVM model preceded by feature scaling using the teo best features discovered in the previous exercise and to use cross validation to validate the model.

#### 4.1.2 Results

After performing the tests we asked the users to fill a questionnaire, which allowed us to know how much the users liked the interface, their experience using the tool and if they found it useful. Each statement could be answered as: totally disagree, disagree, indecisive, agree and totally agree. To analyse the average response and the standard deviation we converted the answers to numbers, where number 1 translates to "totally disagre" and 5 to "totally agree".

As seen in Figure 5 the values are all above average. The most satisfactory results were that users found the interface easy to use, they would recommend it to colleagues and that they would use it again to solve related problems. The attractiveness of the interface, even though it was very positive, scored lower than the other metrics; this was expected since this is a prototype and that part was not a priority. The results acquired from type A users are lower than the ones from type B. This showed that the users with no experience (type A) had more difficulty using the interface which was expected, but surprisingly they found ea-

sier to find the required functionalities and the design simpler to understand.

Besides answering the questionnaire the users also had a place to write suggestions, critiques and what they liked the most in the application. This feedback reinforced what was discovered during the questionnaire and it was very satisfactory. None of the critiques were about the concept we aim to prove and the things they liked the most were inline with the objectives we tried to achieve when building the application.

### 4.2 Computational Performance Tests

Basic preliminary computational performance tests were done to assess how the system will behave with the current architecture. We executed tests using two randomly generated numerical datasets with a binary response class: Dataset 1 containing 10000 rows and 1001 columns (34.2 MB) and Dataset 2 with 20000 rows and 1001 columns (68.4 MB).

Using each dataset we created 10 times a Naïve Bayes model and evaluated its classification performance using 10-fold cross validation.

As a baseline, we performed the same experiments with H2O deployed in an equal cluster.

The results can be seen in Figure 6.

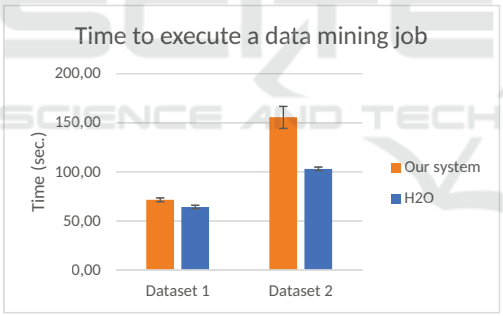


Figure 6: Tests performed with our system and H2O.

It can be seen that our system is slower in the preliminary tests, but this is nothing we were not expecting, as we are storing intermediate results in a centralised disk using NFS, while H2O stores them in memory. We will address this issue in the future.

## 5 CONCLUSION

We presented a service for non-programmers to perform data mining experiments employing good machine learning / data mining practices. We prototyped a cloud application following a microservices architecture with an interface that aims to achieve high usability metrics.

To evaluate a first prototype and validate the paradigm of visual programming using sequential tasks we made experiments with experienced and non-experienced users which provided us satisfactory feedback.

Future works will include not only more usability tests with experienced users to improve the user interface in aesthetics and functionality terms, but mainly the investment in optimising the current architecture, which might include exploring the storage of intermediate results in memory and other techniques that can produce results faster.

## ACKNOWLEDGEMENTS

This work was carried out under the project PTDC/EEI-ESS/1189/2014 Data Science for Non-Programmers, supported by COMPETE 2020, Portugal 2020-POCI, UE-FEDER and FCT.

## REFERENCES

Anderson, E. (1936). The species problem in iris. *Annals of the Missouri Botanical Garden*, 23:457–509.

Cao, L. (2017). Data science: A comprehensive overview. *ACM Comput. Surv.*, 50(3):43:1–43:42.

Cawley, G. C. and Talbot, N. L. (2010). On over-fitting in model selection and subsequent selection bias in performance evaluation. *Journal of Machine Learning Research*, 11(Jul):2079–2107.

Han, J., Pei, J., and Kamber, M. (2011). *Data mining: concepts and techniques*. Elsevier.

Hastie, T., Tibshirani, R., and Friedman, J. (2001). *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA.

Henke, N., Bughin, J., Chui, M., Manyika, J., Saleh, T., Wiseman, B., and Sethupathy, G. (2016). The age of analytics: Competing in a data-driven world. *McKinsey Global Institute*, 4.

Hohpe, G. and Woolf, B. (2003). *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Kranjc, J., Ora, R., Podpean, V., Lavra, N., and Robnik-ikonja, M. (2017). Clowdflows: Online workflows for distributed big data mining. *Future Generation Computer Systems*, 68:38 – 58.

Medvedev, V., Kurasova, O., Bernataviien, J., Treigys, P., Marcinkeviius, V., and Dzemyda, G. (2017). A new web-based solution for modelling data mining processes. *Simulation Modelling Practice and Theory*, 76:34 – 46. High-Performance Modelling and Simulation for Big Data Applications.

Miller, S. and Hughes, D. (2017). The quant crunch: How the demand for data science skills is disrupting the job market. *Burning Glass Technologies*.

Nielsen, J. (1994). Enhancing the explanatory power of usability heuristics. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '94, pages 152–158, New York, NY, USA. ACM.

# Correction to "A Data Mining Service for Non-Programmers", in Proceedings of the 10th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management - Volume 1: KDIR, 340-346, 2018, Seville, Spain

Artur Pedroso, Bruno Leonel Lopes, Jaime Correia, Filipe Araujo,
Jorge Cardoso, Rui Pedro Paiva
CISUC, Dept. of Informatics Engineering,
University of Coimbra, Portugal

October 10, 2018

After publishing our recent conference paper [1] we observed some mistakes. Fortunately, none of these influence our findings and results.

In Section 2 Related Work, we state that the cross validation procedure in Orange is just applied to the model creation phase and does not include prior operations like feature selection. After further research we found that Orange enables the application of preprocessing operations such as feature selection in the cross validation loop.

In the same section we also state that besides RapidMiner, none of the mentioned applications provide the inclusion of data mining experiments in a nested cross validation loop. However, tools like Weka also provide a method to include the data mining experiments in a nested cross validation loop.

In Section 3.1 User Interface, the symbols "{" and "}" that are used in the grammar, presented in EBNF notation, must be replaced with the symbols "[" and "]" respectively.

In Section 4.1.1 Setup, we say that 8 users Type A conducted the usability tests. However, the correct number is 7.

We apologise for the mistakes and for any inconvenience this may have caused. We want other researchers to use our work in the best possible way.

## References

[1] A. Pedroso, B. L. Lopes, J. Correia, F. Araujo, J. Cardoso, and R. P. Paiva, "A data mining service for non-programmers," in *Proceedings of the 10th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management - Volume 1: KDIR,*, INSTICC. SciTePress, 2018, pp. 340–346.