Dsadvisor: A Tool to Support Predictive Tasks in Data Science

José Augusto Câmara Filho[1], **José Maria Monteiro**[1]

[1]Universidade Federal do Ceará
Fortaleza – CE – Brasil

augusto95@alu.ufc.br, monteiro@dc.ufc.br

Abstract. *Currently, professionals from the most diverse areas of knowledge need to explore their data repositories in order to extract knowledge and create new products or services. Several tools have been proposed in order to facilitate the tasks involved in the Data Science lifecycle. However, such tools require their users to have specific (and deep) knowledge in different areas of Computing and Statistics, making their use practically unfeasible for non-specialist professionals in data science. In this paper, we propose a tool, which aims to encourage non-expert users to build machine learning models to solve predictive tasks, extracting knowledge from their own data repositories. More specifically, DSAdvisor*[1] *guides these professionals in predictive tasks involving regression and classification.*

1. Introdução

Due to a large amount of data currently available, arises the need for professionals of different areas to extract knowledge from their repositories to create new products and services. For example, cardiologists need to explore large repositories of electrocardiographic signals in order to predict the likelihood of sudden death in a certain patient. Likewise, tax auditors may want to explore their databases in order to predict the likelihood of tax evasion. However, the volume and variety of data far exceed human capacity for manual analysis. In response, complex algorithms have been developed which allow identifying patterns hidden in these datasets. The convergence of these phenomena has driven the development and popularization of data science [Provost and Fawcett 2013].

Data science is a multidisciplinary area involving the extraction of information and knowledge from large data repositories [Provost and Fawcett 2013]. It deals with the data collection, integration, management, exploration and knowledge extraction to make decisions, understand the past and the present, predict the future, and create new services and products [Ozdemir 2016]. Data science makes it possible to obtain new insights hidden in these datasets. To extract knowledge from the data, we must be able to (i) understand yet unsolved problems with the use of data mining techniques, (ii) understand the data and their interrelationships, (iii) extract a data subset, (iv) create machine learning models in order to solve the selected problem, (v) evaluate the performance of the new models, and (vi) demonstrate how these models can be used in decision-making [Chertchom 2018]. The complexity of the above tasks explains why only highly experienced users can master the entire Data Science lifecycle. On the other hand, several tools have been proposed in order to support the tasks involved in the Data Science lifecycle. However, such tools require their users to have specific (and deep) knowledge in different areas of Computing

[1]DSAdvisor's video https://tinyurl.com/59bvafhu

and Statistics, making their use practically unfeasible for non-specialist professionals in data science.

In this paper, we propose a tool, called DSAdvisor, which aims to encourage non-expert users to build machine learning models to solve regression or classification tasks, extracting knowledge from their own data repositories. DSAdvisor acts like an advisor for non-expert users.

The rest of this paper is organized as follows. Section 2 briefly reviews related works. The implementation of DSAdvisor is illustrated in section 3. Finally, in section 4 we present our conclusions and suggestions for future research.

2. Related Works

Traditional data mining tools help companies establish data patterns and trends by using a number of complex algorithms and techniques. Some of these tools are installed on the desktop to monitor the data and highlight trends, and others capture information residing outside a database [Ramamohan et al. 2012]. As example of such tools, we can cite: KEEL, Knime, Orange, RapidMiner and WEKA [Hasim and Haris 2015].

KEEL (Knowledge Extraction based on Evolutionary Learning) is a software that facilitates the analysis of the behavior of evolutionary learning in different approaches of learning algorithm such as Pittsburgh, Michigan, IRL (iterative rule learning) and GCCL (genetic cooperative-competitive learning) [Alcalá-Fdez et al. 2009].

Knime is a modular environment that enables easy integration of new algorithms, data manipulation and visualization methods. It's interface is configurable allowing the selection of different methods. Specifically, one can select data sources, data prepossessing steps, machine learning algorithms, as well as visualization tools. To create the workflow, the user drag some nodes, drop onto the workbench, and link it to join the input and output ports.

The Orange tool has different features which are visually represented by widgets (e.g. read file, discretize, train SVM classifier, etc.). Each widget has a short description within the interface. Programming is performed by placing widgets on the canvas and connecting their inputs and outputs [Demšar et al. 2013].

RapidMiner provides a visual and user friendly GUI environment. This tool use the process concept. A process may contain subprocesses. Processes contain operators which are represented by visual components. An application wizard provides prebuilt workflows for a number of common tasks including direct marketing, predictive maintenance, sentiment analysis, and a statistic view which provides many statistical graphs [Jovic et al. 2014].

Weka offers four operating options: command-line interface (CLI), Explorer, Experimenter and Knowledge flow. The "Explorer" option allows the definition of data source, data preparation, run machine learning algorithms, and data visualization [Hall et al. 2009].

In this paper, we propose a tool called DSAdvisor, which aims to encourage non-expert users to build machine learning models to solve regression or classification tasks, extracting knowledge from their own data repositories. DSAdvisor acts like an advisor

for non-expert users and follow a well-defined guideline.

3. DSADVISOR

In order to evaluate the guideline proposed in this paper, we built a tool called DSAdvisor to encourage non-expert users to build machine learning models to solve predictive (regression or classification) tasks. DSAdvisor was developed in Flask [Grinberg 2018] and Python. Besides, DSAdvisor follows all stages of the guideline proposed in [Filho et al. 2021].

3.1. Phase 1: Exploratory Analysis

The first phase of the DSAdvisor aims to analyze a dataset, provided by the user, and next, describe and summarize it. This first phase comprises the following activities: uploading the data, checking the type of variables, removing variables, choosing missing value codes, exhibiting descriptive statistics, plotting categorical and discrete variables, analyzing distributions, and displaying correlations [Filho et al. 2021]. Due to space limitations, we will highlight just some of these activities.

DSAdvisor uses the following tests to check the normality (whether the data follow a normal distribution or not) of each column (feature): Cramér Von Mises, D'Agostino's K-square, Lilliefors, Shapiro-Wilk. The Kolmogorov-Smirnov test is used to assess which distributions is the one that most closely matches the column distribution (best fit). The result of each test will be shown on the screen along with a histogram combined with two probability functions, the first being the normal distribution and the other the closest approximation by the Kolmogorov-Smirnov test. For each variable, the user will have the option to choose between which of the distributions seems to fit a certain variable, as shown in the Figure 1.

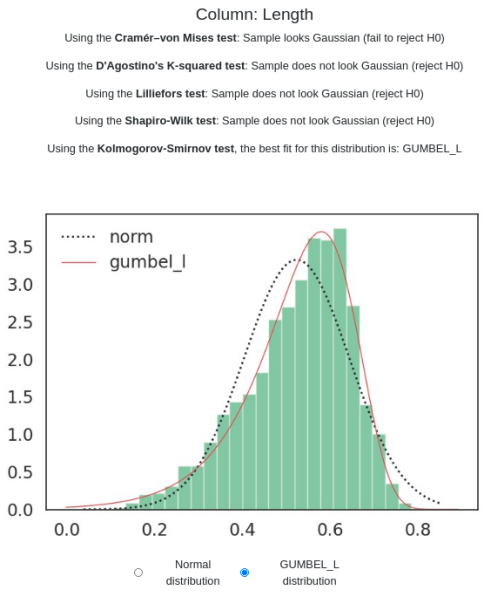

Figura 1. An Example of Distribution Analysis ("Best Fit") Screen applied to the column "Length"from Abalone Dataset.

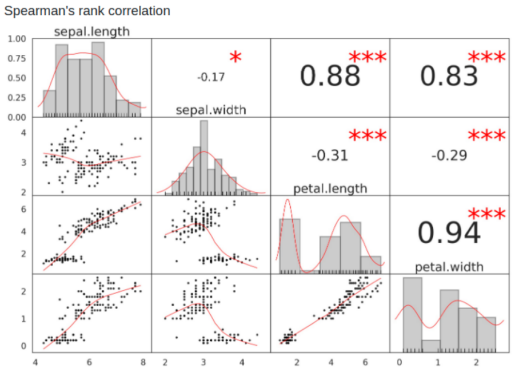

Figura 2. Correlation Matrix Applied to the Iris Flower Dataset.

In order to show the correletion between the dataset features, DSAdvisor uses a combined graph called correlation matrix (Figure 2). This graph shows the Spearman's correlation coefficient for each pair of numeric variables and the Pearson's correlation coefficient for each pair of numeric variables that follows a normal distribution.

3.2. Phase 2: Data Preprocessing

Data preprocessing is an essential component to solve many predictive tasks. The purpose of the second phase of the proposed guide is to prepare the data in order to use it to build predictive models. This phase includes activities related to outlier detection, data normalization, choose the independent variable, selection of attributes, data balancing, feature selection, and division of training and testing sets [Filho et al. 2021]. Due to space limitations, we will highlight just some of these activities.

There are two main techniques to detect outliers: interquartile range (a univariate parametric approach) and adjusted boxplot (a univariate nonparametric approach). DSAdvisor uses these methods in order to detect outliers, as shown in Figure 3. If the user wishes to know precisely what these values are, they can go to the outliers table option to check the position and value of the outliers for each variable.

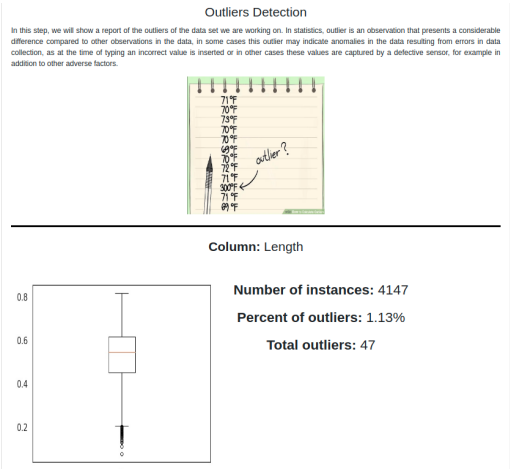

Figura 3. An Example of the Outlier Detection screen applied to the column "Length"from Abalone Dataset.

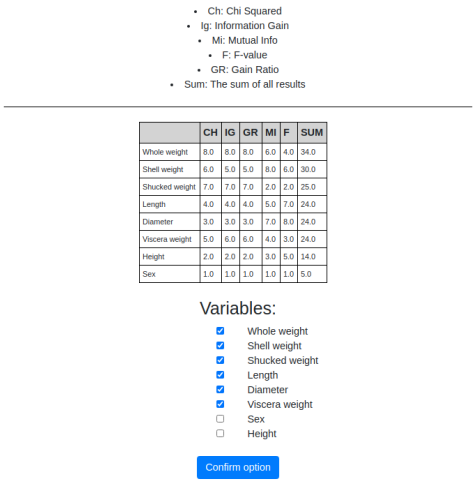

Figura 4. An Example of the Feature Selection screen applied to the Abalone Dataset.

Feature selection is referred to the process of obtaining a subset from an original feature set according to certain feature selection criterion, which selects the relevant features of the dataset. Feature selection methods fall into three categories: filters, wrappers, and embedded/hybrid methods. The DSAdvisor tool will execute the following filters on each variable: Chi Squared, Information Gain, Mutual Info, F-Value and Gain Ratio. For indicating which variables are most relevant to proceeding with the tool, the following heuristic is implemented (Figure 4), depending on the result of each variable in each method they will be ranked from the highest score to the lowest, a sum column will be added to show the sum of each variable in each method and, given the highest value divided by two, the threshold will be given, which for values above it will be marked and those below it will not. However, the tool leaves the final decision to the user to choose which ones to select.

3.3. Phase 3: Building Predictive Models

This phase aims to generate predictive models and analyze their results. For this purpose we introduce pipeline, pipeline is a Sklearn class [Pedregosa et al. 2011] to sequentially apply a list of transformations and final estimator on a dataset. Pipeline objects chain multiple estimators into a single one. This is useful since a machine learning workflow typically involves a fixed sequence of processing steps (e.g., feature extraction, dimensionality reduction, learning and making predictions), many of which perform some kind of learning. A sequence of N such steps can be combined into a pipeline if the first $N1$ steps are transformers; the last can be either a predictor, a transformer or both [Buitinck et al. 2013]. For evaluating statistical performance in pipeline, we use a GridSearch with K-Fold Cross Validation. Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample. K-fold Cross-Validation involves randomly dividing the set of observations into k groups, or folds, of approximately equal size. The first fold is treated as a validation set, and the method is fit on the remaining $k - 1$ folds. For example, using the mean squared error as score function, MSE_1, is then computed on the observations in the held-out fold. This procedure is repeated k times; each time, a different group of observations is treated as a validation set. This process results in k estimates of the test error, MSE_1, MSE_2,...MSE_k. The K-Fold Cross-Validation estimate is computed by averaging these values [James et al. 2013]. In our case, we focus on making predictions in the data set for classification and regression tasks, our models will have the algorithms chosen by the user according to the task to be performed. In order to compare the models' performance, it is necessary to use suitable metrics. After running the pipeline, just take the metrics previously chosen by the user to be calculated and presented to the user in an explanatory way about each selected metric.The last step in this phase consists on ensuring the experiment's reproducibility in order to verify the credibility of the proposed study. [Olorisade et al. 2017] have evaluated studies in order to highlight the difficulty of reproducing most of the works in state-of-art. Some authors have proposed basic rules for reproducible computational research, as [Sandve et al. 2013], based on these rules we save all the decisions made by the user, the results obtained whether they are tables or graphs, the seed and settings of the algorithms used, all in a final document. The activities that make up this phase are described in [Filho et al. 2021].

4. CONCLUSIONS AND FUTURE WORKS

In this paper, we propose a tool, called DSAdvisor, which following the stages of the proposed guideline. DSAdvisor aims to encourage non-expert users to build machine learning models to solve predictive tasks, extracting knowledge from their own data repositories. More specifically, DSAdvisor guides these professionals in predictive tasks involving regression and classification. As future works we intent to carry out usability tests and interviews with non-expert users, in order to evaluate DSAdvisor.

Referências

Alcalá-Fdez, J., Sanchez, L., Garcia, S., del Jesus, M. J., Ventura, S., Garrell, J. M., Otero, J., Romero, C., Bacardit, J., Rivas, V. M., et al. (2009). Keel: a software tool to assess evolutionary algorithms for data mining problems. *Soft Computing*, 13(3):307–318.

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., et al. (2013). Api design for machine learning software: experiences from the scikit-learn project. *arXiv preprint arXiv:1309.0238*.

Chertchom, P. (2018). A comparison study between data mining tools over regression methods: Recommendation for smes. In *2018 5th International Conference on Business and Industrial Research (ICBIR)*, pages 46–50. IEEE.

Demšar, J., Curk, T., Erjavec, A., Gorup, Č., Hočevar, T., Milutinovič, M., Možina, M., Polajnar, M., Toplak, M., Starič, A., et al. (2013). Orange: data mining toolbox in python. *the Journal of machine Learning research*, 14(1):2349–2353.

Filho, J. A. C., Monteiro, J. M., Mattos, C. L. C., and Nobre, J. S. (2021). A practical guide to support predictive tasks in data science. In Filipe, J., Smialek, M., Brodsky, A., and Hammoudi, S., editors, *Proceedings of the 23rd International Conference on Enterprise Information Systems, ICEIS 2021, Online Streaming, April 26-28, 2021, Volume 1*, pages 248–255. SCITEPRESS.

Grinberg, M. (2018). *Flask web development: developing web applications with python*. "O'Reilly Media, Inc.".

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18.

Hasim, N. and Haris, N. A. (2015). A study of open-source data mining tools for forecasting. In *Proceedings of the 9th International Conference on Ubiquitous Information Management and Communication*, pages 1–4.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An introduction to statistical learning*, volume 112. Springer.

Jovic, A., Brkic, K., and Bogunovic, N. (2014). An overview of free software tools for general data mining. In *2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 1112–1117. IEEE.

Olorisade, B. K., Brereton, P., and Andras, P. (2017). Reproducibility in machine learning-based studies: An example of text mining.

Ozdemir, S. (2016). *Principles of data science*. Packt Publishing Ltd.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

Provost, F. and Fawcett, T. (2013). Data science and its relationship to big data and data-driven decision making. *Big data*, 1(1):51–59.

Ramamohan, Y., Vasantharao, K., Chakravarti, C. K., Ratnam, A., et al. (2012). A study of data mining tools in knowledge discovery process. *International Journal of Soft Computing and Engineering (IJSCE) ISSN*, 2(3):2231–2307.

Sandve, G. K., Nekrutenko, A., Taylor, J., and Hovig, E. (2013). Ten simple rules for reproducible computational research. *PLoS Comput Biol*, 9(10):e1003285.