

A Practical Guide to Support Predictive Tasks in Data Science

José Augusto Câmara Filho¹, José Maria Monteiro¹, César Lincoln Mattos¹
and Juvêncio Santos Nobre²

¹*Department of Computing, Federal University of Ceará, Fortaleza, Ceará, Brazil*

²*Department of Statistics and Applied Mathematics, Federal University of Ceará, Fortaleza, Ceará, Brazil*

Keywords: Practical Guide, Prediction, Data Science.

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 guideline to support predictive tasks in data science. In addition to being useful for non-experts in Data Science, the proposed guideline can support data scientists, data engineers or programmers which are starting to deal with predictive tasks. Besides, we present a tool, called DSAdvisor, which follows 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.

1 INTRODUCTION

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, in order to build predictive models, these non-specialist professionals need to acquire knowledge in different areas of Computing and Statistics, making this task practically unfeasible. Another alternative is ask experienced data science professionals to help, which creates dependency instead of autonomy. In this context, the popularization of data science becomes an important research problem (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, pre-

dict the future, and create new services and products (Ozdemir, 2016). Data science makes it possible to identifying patterns hidden and obtain new insights hidden in these datasets, from complex machine learning algorithms.

The Data Science lifecycle has six stages: business grasp, data understanding, data preparation, modeling, evaluation, and deployment. 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 previous 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 and Statistics, making their use practically unfeasible for non-specialist professionals in data science.

In this paper, we propose a guideline to support

predictive tasks in data science. In addition to being useful for non-experts in Data Science, the proposed guideline can support data scientists, data engineers or programmers which are starting to deal with predictive tasks. In addition, we present 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 regression or classification tasks, extracting knowledge from their own data repositories. DSAdvisor acts like an advisor for non-expert users or novice data scientists.

The rest of this paper is organized as follows. Section 2 reviews related works. In section 3, the proposed guideline is laid out. The DSAdvisor is commented in section 4. Finally, in section 5 we present our conclusions and suggestions for future research.

2 RELATED WORKS

In this section we will discuss the main related works. For a better understanding, we organized the related works into two categories: supporting tools and practical guidelines.

2.1 Data Mining Tools

Traditional data mining tools help companies establish data patterns and trends by using a number of complex algorithms and techniques. 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 allows the selection of different data sources, data preprocessing 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 uses 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). DSAdvisor is an advisor for non-expert users or novice data scientists, which following the stages of the guideline proposed in this paper. DSAdvisor aims to encourage non-expert users to build machine learning models to solve regression or classification tasks, extracting knowledge from their own data repositories.

Even before the popularization of data Science, all these tools were developed to help with data mining tasks. These tools differ regarding tool usability, type of license, the language in which they were developed, support for data understanding, and missing values handle. The most widely used tools include KEEL, Knime, Orange, RapidMiner, Tanagra, and Weka. The table 1 provides a comparison between these tools and the DSAdvisor.

In the other hand, AutoML tools enable you to automate some machine learning tasks. Although it be important to automate all machine learning tasks, that is not what AutoML does. Rather, it focuses on a few repetitive tasks, such as: hyperparameter optimization, feature selection, and model selection. Examples of these tools include: AutoKeras, Auto-WEKA, Auto-Sklearn, DataRobot, H2O and MLBox.

2.2 Guidelines

A guideline is a roadmap determining the course of a set of actions that make up a specific process, in addition to a set of good practices for the performance of these activities (Dictionary, 2015). Some guidelines have been proposed to manage general data mining tasks.

In (Melo et al., 2019), the authors presented a practical guideline to support the specific problem of predict change-proneness classes in oriented object software. In addition, they applied their guideline over a case study using a large imbalanced dataset extracted from a wide commercial software. It is important to highlight that, in this work, we extend the

Table 1: General characteristics of data mining software. Adapted from (Hasim and Haris, 2015).

Softwares list					
Software	Usability	License	Language	Data Understanding	Missing values handle
DSAdvisor	High	GPL	Python	Perform	Intermediate
KEEL	High	GPL	Java	Perform	Basic
KNIME	Low	Outra	Java	Perform	Basic
RapidMiner	High	GPL	Java	Partially performs	Basic
Orange	Low	GPL	C++, Python	Partially performs	Basic
Weka	Low	GPL	Java	Partially performs	Basic

guideline proposed in (Melo et al., 2019) to the more general data science context.

(Luo et al., 2016) highlight the flexibility of the emerging machine learning techniques, however, there is uncertainty and inconsistency in the use of such techniques. Machine learning, due to its intrinsic mathematical and algorithmic complexity, is often considered “black magic” that requires a delicate balance of a large number of conflicting factors. This, together with inadequate reporting of data sources and the modeling process, makes the research results reported in many biomedical articles difficult to interpret. It is not uncommon to see potentially spurious conclusions drawn from methodologically inadequate studies, which in turn undermines the credibility of other valid studies and discourages many researchers who could benefit from adopting machine learning techniques. In the light of this, guidelines are proposed for the use of predictive models in clinical settings, ensuring that activities are carried out correctly and reported.

3 THE PROPOSED GUIDELINE FOR PREDICTIVE TASKS

This section describes the proposed guide to support regression and classification tasks, which is organized into three phases: exploratory analysis, data preprocessing and building predictive models. Each one of these phases will be detailed next.

3.1 Phase 1: Exploratory Analysis

The first phase of the proposed guideline aims to analyze a dataset, provided by the user, and next, describe and summarize it. Figure 1 illustrates this phase, which 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.

So, the guide indicates the use of different descriptive statistics, such as the number of lines (count), mean, standard deviation (std), coefficient of variation (cv), minimum (min), percentiles (25%, 50%, 75%) and maximum (max) for numerical variables, and count, number of distinct values (unique), the most frequent element (top) and the most common value’s frequency (freq) for categorical variables. Besides, the guideline recommend the use of different strategies for identifying and showing missing values.

Moreover, the proposed guideline suggests different methods to assess the distribution of variables, such as Cramér Von Mises (Cramér, 1928), D’Agostino’s K-squared (D’Agostino, 1970), Lilliefors (Lilliefors, 1967), Shapiro-Wilk (Shapiro and Wilk, 1965) and Kolmogorov-Smirnov (Smirnov, 1948). All these methods serve to determine the veracity of a hypothesis (Hirakata et al., 2019). In this guide we want to state whether a variable follows (H0) or not (H1) a normal distribution, and each method has its own way of calculating and returning a result to identify the most suitable hypothesis. Each test considers two hypotheses about the variable under study. One is called the null hypothesis (H0), which is assumed to be true until proven otherwise. The second is called the alternative hypothesis (H1), which represents a statement that the parameter of interest differs from that defined in the null hypothesis, so that the two hypotheses are complementary. The hypotheses used by the tests are: i) H0: The variable follows a normal distribution and ii) H1: The variable does not follow a normal distribution.

Finally, the guide indicates the use of different correlation coefficients based on the data distribution. Spearman’s correlation coefficients (Spearman, 1961) will be displayed for all pair of numerical data. If a pair of variables (columns or features) follow a normal distribution, the Pearson’s correlation coefficients (Pearson, 1895) must be computed and shown. If the user dataset contains categorical data, the guide suggest to display Cramer’s V. (Cramér, 1928).

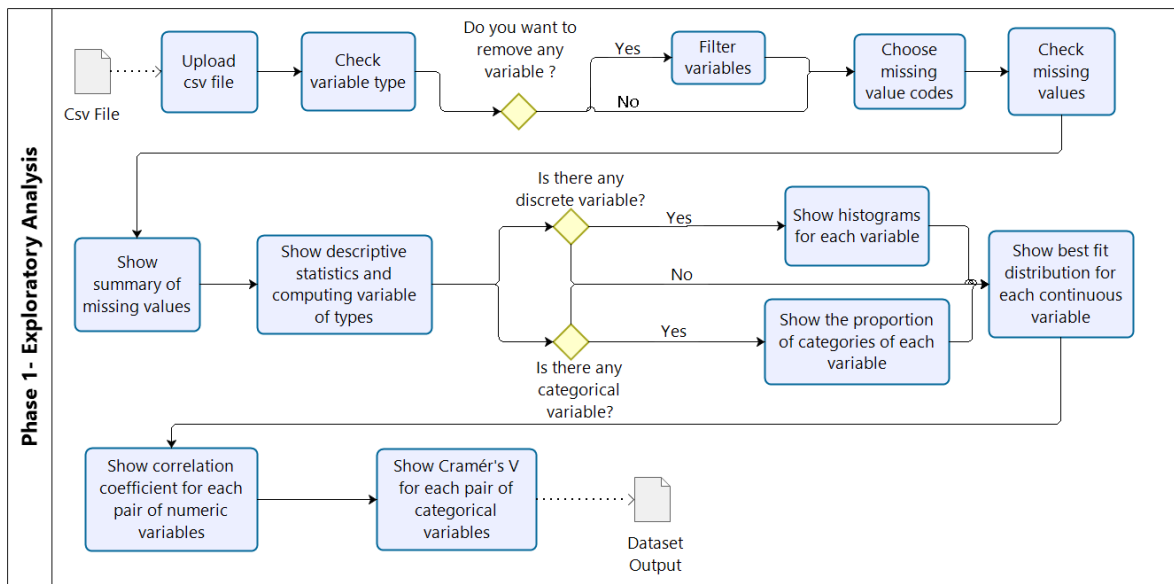


Figure 1: Phase 1 - Exploratory Analysis.

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. Figure 2 illustrates the activities that make up the second phase of the proposed guide.

3.2.1 Problem Setup

In this step, the predictive task type (regression or classification) must be defined. Besides, the dependent variable should be identified. Moreover, the proportion of the training and test sets have to be specified. Other options for the third phase are the predictive algorithms list, score function for optimizing GridSearchCV and metrics list for Results Presentation. The list of predictive algorithms, the list of metrics and the score function will be according to the type of problem to be treated, classification's or regression's problem.

3.2.2 Choose Normalization Techniques

Verifying that the variables are on the same scale is an essential step at this point. For example, two variables may be expressed in different ranges, such as integers and the interval between 0 and 1. Therefore,

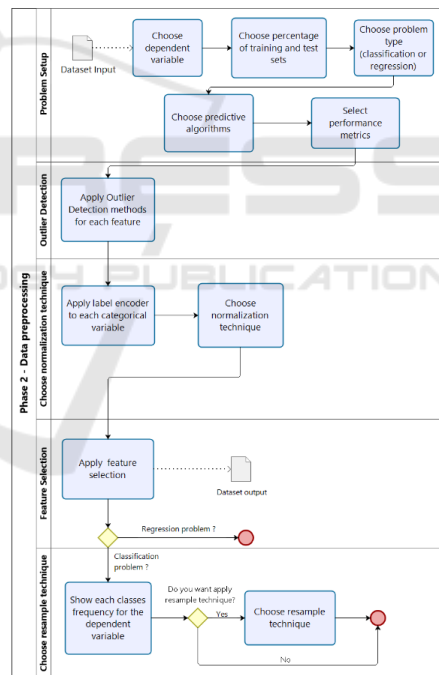


Figure 2: Phase 2 - Data preprocessing.

it is necessary to normalize all variables in the dataset. For instance, for the activation function in the neural network it is recommended that the data be normalized between 0.1 and 0.9 instead of 0 and 1 to avoid saturation of the sigmoid function (Basheer and Hajmeer, 2000). The normalization techniques used in this guideline are z-score and min-max normalization.

3.2.3 Outlier Detection

Outliers are extreme values that deviate from other observations on data (i.e., an observation that diverges from an overall pattern on a sample). Detected outliers are candidates for aberrant data that may otherwise adversely lead to model misspecification, biased parameter estimation and incorrect results. It is therefore important to identify them prior to creating the prediction model (Liu et al., 2004).

A survey to distinguish between univariate vs. multivariate techniques and parametric (*Statistical*) vs. nonparametric procedures was done by (Bengal, 2005). Detecting outliers is possible when multivariate analysis is performed and the combinations of variables are compared with the class of data. In other words, an instance can be a multivariate outlier but a usual value in each feature, or it can have values that are outliers in several features, but the whole instance might also be a usual multivariate value (Escalante, 2005).

There are two main techniques to detect outliers: interquartile range (a univariate parametric approach) and adjusted boxplot (a univariate nonparametric approach). Next, we describe each one of these techniques.

Interquartile Range. The interquartile range (IQR) is a measure of statistical dispersion, often used to detect outliers. The IQR is the length of the box in the boxplot (i.e., $Q3 - Q1$). Here, outliers are defined as instances below $Q1 - 1.5 * IQR$ or above $Q3 + 1.5 * IQR$.

Adjusted Boxplot. Note that the boxplot assumes symmetry because we add the same amount to $Q3$ as we subtract from $Q1$. In asymmetric distributions, the usual boxplot typically flags many regular data points as outlying. The skewness-adjusted boxplot corrects that by using a robust measure of skewness in determining the fence (Hubert and Vandervieren, 2008). In this new approach, outliers are defined as instances such that if $medcouple (MC) \geq 0$, they are below $Q1 - 1.5e^{-4mc} * IQR$ or above $Q1 + 1.5e^{3mc} * IQR$; if not, they are below $Q1 - 1.5e^{-3mc} * IQR$ or above $Q1 + 1.5e^{4mc} * IQR$. To measure the skewness of a univariate sample (x_1, \dots, x_n) from a continuous unimodal distribution F , we use the MC , and $Q2$ is the sample median (Brys et al., 2004). It is defined as in equation 1:

$$MC = med_{x_i \leq Q2 \leq x_j} h(x_i, x_j) \quad (1)$$

Outliers are one of the main problems when building a predictive model. Indeed, they cause data scientists to achieve suboptimal results. To solve that, we need effective methods to deal with spurious points. If it is

obvious that the outlier is due to incorrectly entered or measured data, you should drop the outlier.

3.2.4 Feature Selection

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. It plays a role in compressing the data processing scale, where the redundant and irrelevant features are removed (Cai et al., 2018). Feature selection technique can pre-process learning algorithms, and good feature selection results can improve learning accuracy, reduce learning time, simplify learning results, reduction of dimensional space and removal of redundant, irrelevant or noisy data (Ladha and Deepa, 2011).

Feature selection methods fall into three categories: filters, wrappers, and embedded/hybrid methods. Filter method takes less computational time for selecting the best features. As the correlation between the independent variables is not considered while selecting the features, this leads to selection of redundant features (Venkatesh and Anuradha, 2019). Wrapper are brute-force feature selection methods that exhaustively evaluate all possible combinations of the input features to find the best subset. Embedded/hybrid methods combine the advantages of both approaches, filters and wrappers. A hybrid approach uses both performance evaluation function of the feature subset and independent test. (Veerabhadrapa and Rangarajan, 2010).

3.2.5 Choose Resample Techniques for Imbalanced Data

Imbalanced data problem occurs in many real-world datasets where the class distributions of data are asymmetric. It is important to note that most machine learning models work best when the number of instances of each class is approximately equal (Longadge and Dongre, 2013). The imbalanced data problem causes the majority class to dominate the minority class; hence, the classifiers are more inclined to the majority class, and their performance cannot be reliable (Kotsiantis et al., 2006).

Many strategies have been generated to handle the imbalanced data problem. The sampling-based approach is one of the most effective methods that can solve this problem. The sampling-based approach can be classified into three categories, namely: Over-Sampling (Yap et al., 2014), Under-Sampling (Liu et al., 2008), and Hybrid Methods (Gulati, 2020). In this guide we recommend the use of over-sampling and under-sampling.

Over-sampling (OS). Over-sampling raises the weight of the minority class by replicating or creating new minority class samples. There are different over-sampling methods; moreover, it is worth noting that the over-sampling approach is generally applied more frequently than other approaches.

- **Random Over Sampler:** This method increases the size of the dataset by the repetition of the original samples. The point is that the random over-sampler does not create new samples, and the variety of samples does not change (Li et al., 2013).
- **Smote:** This method is a statistical technique that increases the number of minority samples in the dataset by generating new instances. This algorithm takes samples of the feature space for each target class and its nearest neighbors, and then creates new samples that combine features of the target case with features of its neighbors. The new instances are not copies of existing minority samples (Chawla et al., 2002).

Under-sampling (US). Under-sampling is one of the most straightforward strategies to handle the imbalanced data problem. This method under-samples the majority class to balance the class with the minority class. The under-sampling method is applied when the amount of collected data is sufficient. There are different under-sampling models, such as Edited Nearest Neighbors (ENN) (Guan et al., 2009), Random Under-Sampler (RUS) (Batista et al., 2004) and Tomek links (Elhassan and Aljurf, 2016), which are the most popular.

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 $N - 1$ 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, $MSE_1, MSE_2, \dots, MSE_k$. 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 in 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. Figure 3 illustrates the activities that make up the third phase of the proposed guide.

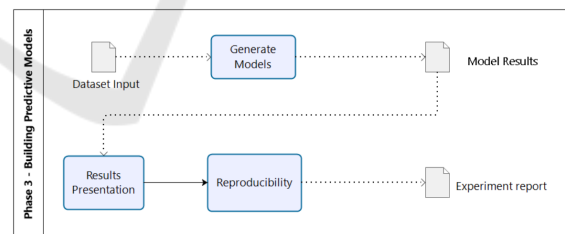


Figure 3: Phase 3 - Building Predictive Models.

4 DSAdvisor

DSAdvisor is an advisor for non-expert users or novice data scientists, which following the stages of the guideline proposed in this paper. DSAdvisor aims to encourage non-expert users to build machine learning models to solve predictive tasks (regression or classification), extracting knowledge from their own data repositories. This tool was developed in CSS3, HTML5, Flask (Grinberg, 2018), JavaScript

and Python (van Rossum, 1995).

DSAdvisor is an open source tool, developed using the Python programming language. So, this makes possible to reuse the large number of Python APIs and Toolkis currently available. Besides, DSAdvisor provides different resources to support data understanding and missing values detection. In addition, DSAdvisor guides the user on the task of outlier detection showing the number of instances, the percentage of outliers found and the total of outliers. 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. Furthermore, the DSAdvisor tool helps the user on the feature selection task, running the following filters methods: Chi Squared, Information Gain, Mutual Info, F-Value and Gain Ratio. In order to tackle the imbalanced data problem, DSAdvisor supports three alternatives: Oversampling, Undersampling and Without resampling techniques. To assess which type of problem is being addressed (classification or regression), DSAdvisor asks some questions to the user. Next, based on the user responses, DSAdvisor suggests the most appropriate problem type. In case of choosing classification, the pre-selected algorithms will be logistic regression, naive bayes, support vector machine, decision tree, and multi layer perceptron. In case of choosing regression, the pre-selected algorithms will be linear regression, support vector machine, multi layer perceptron, radial basis function. After this, DSAdvisor suggests the most suitable performance metrics. Finally, the user has the option to download the execution log of all the options selected previously, the results obtained whether they are tables or graphs, and the settings of all used algorithms. This option ensures that the experiments performed in the DSAdvisor tool are reproducible by other users.

5 CONCLUSIONS AND FUTURE WORKS

In this paper, we propose a guideline to support predictive tasks in data science. In addition, we present a tool, called DSAdvisor, which follows 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 intend to carry out usability tests and interviews with non-expert users, in order to evaluate DSAdvisor.

REFERENCES

- 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.
- Basheer, I. and Hajmeer, M. (2000). Artificial neural networks: fundamentals, computing, design, and application. *Journal of Microbiological Methods*, 43(1):3 – 31. Neural Computing in Microbiology.
- Batista, G., Prati, R., and Monard, M.-C. (2004). A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explorations*, 6:20–29.
- Ben-Gal, I. (2005). *Outlier Detection*, pages 131–146. Springer US, Boston, MA.
- Brys, G., Hubert, M., and Struyf, A. (2004). A robust measure of skewness. *Journal of Computational and Graphical Statistics*, 13(4):996–1017.
- 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*.
- Cai, J., Luo, J., Wang, S., and Yang, S. (2018). Feature selection in machine learning: A new perspective. *Neurocomputing*, 300:70–79.
- Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357.
- 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.
- Cramér, H. (1928). On the composition of elementary errors: First paper: Mathematical deductions. *Scandinavian Actuarial Journal*, 1928(1):13–74.
- D’Agostino, R. B. (1970). Transformation to normality of the null distribution of g_1 . *Biometrika*, pages 679–681.
- 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.
- Dictionary, C. (2015). Cambridge dictionaries online.
- Elhassan, T. and Aljurf, M. (2016). Classification of imbalance data using totem link (t-link) combined with random under-sampling (rus) as a data reduction method.
- Escalante, H. J. (2005). A comparison of outlier detection algorithms for machine learning. *Programming and Computer Software*.
- Grinberg, M. (2018). *Flask web development: developing web applications with python*. ” O’Reilly Media, Inc.”.
- Guan, D., Yuan, W., Lee, Y.-K., and Lee, S. (2009). Nearest neighbor editing aided by unlabeled data. *Information Sciences*, 179(13):2273–2282.

- Gulati, P. (2020). Hybrid resampling technique to tackle the imbalanced classification problem.
- 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.
- Hirakata, V. N., Mancuso, A. C. B., and Castro, S. M. d. J. (2019). Teste de hipóteses: perguntas que você sempre quis fazer, mas nunca teve coragem. *Teste de hipóteses: perguntas que você sempre quis fazer, mas nunca teve coragem. Vol. 39, n. 2, 2019, p. 181-185.*
- Hubert, M. and Vandervieren, E. (2008). An adjusted boxplot for skewed distributions. *Computational statistics & data analysis*, 52(12):5186–5201.
- 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.
- Kotsiantis, S., Kanellopoulos, D., Pintelas, P., et al. (2006). Handling imbalanced datasets: A review. *GESTS International Transactions on Computer Science and Engineering*, 30(1):25–36.
- Ladha, L. and Deepa, T. (2011). Feature selection methods and algorithms. *International Journal on Computer Science and Engineering*.
- Li, H., Li, J., Chang, P.-C., and Sun, J. (2013). Parametric prediction on default risk of chinese listed tourism companies by using random oversampling, isomap, and locally linear embeddings on imbalanced samples. *International Journal of Hospitality Management*, 35:141–151.
- Lilliefors, H. W. (1967). On the kolmogorov-smirnov test for normality with mean and variance unknown. *Journal of the American statistical Association*, 62(318):399–402.
- Liu, H., Shah, S., and Jiang, W. (2004). On-line outlier detection and data cleaning. *Computers & Chemical Engineering*, 28(9):1635 – 1647.
- Liu, X.-Y., Wu, J., and Zhou, Z.-H. (2008). Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(2):539–550.
- Longadge, R. and Dongre, S. (2013). Class imbalance problem in data mining review. *arXiv preprint arXiv:1305.1707*.
- Luo, W., Phung, D., Tran, T., Gupta, S., Rana, S., Karmakar, C., Shilton, A., Yearwood, J., Dimitrova, N., Ho, T. B., et al. (2016). Guidelines for developing and reporting machine learning predictive models in biomedical research: a multidisciplinary view. *Journal of medical Internet research*, 18(12):e323.
- Melo, C. S., da Cruz, M. M. L., Martins, A. D. F., Matos, T., da Silva Monteiro Filho, J. M., and de Castro Machado, J. (2019). A practical guide to support change-proneness prediction. In *ICEIS (2)*, pages 269–276.
- 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.
- Pearson, K. (1895). Notes on regression and inheritance in the case of two parents proceedings of the royal society of london, 58, 240-242.
- 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.
- Sandve, G. K., Nekrutenko, A., Taylor, J., and Hovig, E. (2013). Ten simple rules for reproducible computational research. *PLoS Comput Biol*, 9(10):e1003285.
- Shapiro, S. S. and Wilk, M. B. (1965). An analysis of variance test for normality (complete samples). *Biometrika*, 52(3/4):591–611.
- Smirnov, N. (1948). Table for estimating the goodness of fit of empirical distributions. *The annals of mathematical statistics*, 19(2):279–281.
- Spearman, C. (1961). The proof and measurement of association between two things.
- van Rossum, G. (1995). Python tutorial. Technical Report CS-R9526, Centrum voor Wiskunde en Informatica (CWI), Amsterdam.
- Veerabhadrapa and Rangarajan, L. (2010). Bi-level dimensionality reduction methods using feature selection and feature extraction. *International Journal of Computer Applications*, 4.
- Venkatesh, B. and Anuradha, J. (2019). A review of feature selection and its methods. *Cybernetics and Information Technologies*, 19(1):3–26.
- Yap, B. W., Abd Rani, K., Abd Rahman, H. A., Fong, S., Khairudin, Z., and Abdullah, N. N. (2014). An application of oversampling, undersampling, bagging and boosting in handling imbalanced datasets. In *Proceedings of the first international conference on advanced data and information engineering (DaEng-2013)*, pages 13–22. Springer.