# Structured Data Preparation Pipeline for Machine Learning-Applications in Production

Frye, Maik[1], Schmitt, Robert Heinrich[2]

*[1]Fraunhofer Institute for Production Technology IPT, Steinbachstraße 17, 52074, Aachen, Germany*
*[2]Laboratory for Machine Tools WZL RWTH Aachen University, Cluster Production Engineering 3A 540, Aachen 52074, Germany*

***Abstract* – The application of machine learning (ML) is becoming increasingly common in production. However, many ML-projects fail due to the existence of poor data quality. To increase its quality, data needs to be prepared. Through the consideration of versatile requirements, data preparation (DPP) is a challenging task, while accounting for 80 % of ML-projects duration [1]. Nowadays, DPP is still performed manually and individually making it essential to structure the preparation in order to achieve high-quality data in a reasonable amount of time. Thus, we present a holistic concept for a structured and reusable DPP-pipeline for ML-applications in production. In a first step, requirements for DPP are determined based on project experiences and detailed research. Subsequently, individual steps and methods of DPP are identified and structured. The concept is successfully validated through two production use-cases by preparing data sets and implementing ML-algorithms.**

***Keywords – Artificial Intelligence, Machine Learning, Data Preparation, Data Quality***

## I. INTRODUCTION

Due to developments towards a networked, adaptive production, an ever increasing amount of data is generated enabling comprehensive data analyses. For analysing data, machine learning (ML) and artificial intelligence (AI) are commonly used [2]. ML-methods enable the training of AI-systems. These technologies have already proven the potential for process optimization in many application areas [3]. ML and AI continue to gain popularity because of the ability to handle complex interrelationships and recognize patterns from data [4].

However, the implementation of ML and AI reveals versatile challenges, while ensuring sufficient data quality is accounted to be one of the greatest challenge [5]. Poor data quality results in poor analysis' results, which is also known as garbage in, garbage out (GIGO) principle [6]. According to a survey, 77 % of companies assume that poor results are due to inaccurate and incomplete data [7]. Insufficient data quality also significantly affects businesses. Based on Gartner's research, "the average financial impact of poor data quality is $ 9.7 million per year" [8]. Consequently, poor data quality is one of the main reasons for the failure of ML and AI-projects [9].

The challenge in ensuring high data quality are many different influencing factors and requirements. On the one hand, basic prerequisites for data analysis must be met, such as the correct assignment of process and product quality data via unique identifiers. On the other hand, properties of data sets as well as ML-algorithms require target-oriented DPP.

Due to the requirements, the process of DPP takes about 80 % of the total project duration. In general, the selection of DPP-methods for one use-case differs from another use-case, which leads to a non-reproducible DPP-pipeline, in which preparation is performed both manually and individually. For these reasons, we present a comprehensive concept for a structured and reusable DPP-pipeline for ML-applications in production. In a first step, requirements for DPP are determined based on project experiences and detailed research. Subsequently, individual steps and methods of DPP are identified and structured. The concept will be validated through two different production use-cases by preparing concrete data sets and implementing ML-algorithms.

The paper is structured as follows. In the following chapter, literature is reviewed with regard to available DPP-methods and existing approaches to structuring DPP. Thirdly, the methodology is presented, which is explained in detail in the fourth chapter and evaluated on the basis of two production use-cases. The paper concludes with a final conclusion and an outlook.

## II. RELATED RESULTS IN THE LITERATURE

In this section, the literature is reviewed according to existing DPP-methods and concepts for structuring DPP.

### *A. Existing DPP-Methods*

Hundreds of methods exist to prepare data for a subsequent training of ML-algorithms. Garcia et al. 2015 classified several methods into data integration, cleaning, normalization and transformation [10]. Similarly, in Han et al. 2012, different methods were presented and assigned to categories of cleaning, integration, reduction, transformation and discretization [11]. Kotsiantis et al. 2007 emphasized the necessity of high data quality and presented DPP-methods specifically dedicated to supervised learning algorithms [5].

Libraries used for preparation provide a wide range of DPP-methods. Sklearn, for instance, offers comprehensive documentation in a predefined structure [12]. Further, Sklearn contributions, such as categorical-encoders, extend the number of available DPP-methods [13]. Besides that, there are libraries that focus on specific data types, such as tsfresh for time series or OpenCV for image data [14, 15]. However, many existing methods are not covered by libraries, which leads to a rare use in production.

### B. Structuring DPP

There are already both generic and application-oriented approaches to structuring DPP. Generic approaches provide general design rules and methods for DPP such as data transformation. These approaches are often available in form of cheat sheets, which are, however, rather aimed at the application of ML-models than at DPP [16–18]. General design rules do not address a specific domain, while the assistance is independent of applications. A structured DPP is therefore not enabled.

On the other hand, there are application-oriented approaches that take domain-specific requirements into account. One example is the prediction of depression, in which selected DPP-methods are implemented consecutively [19]. The same applies to cost estimation of software projects as well as gesture recognition [20, 21]. However, only a very limited as well as rigidly predefined selection of DPP-methods is considered. Thus, these efforts can only be assessed as partially structured DPP-pipeline and do not refer to production environments.

Consequently, numerous methods exist, which are available through different libraries. However, no approach could be found, how to structure DPP for production purposes.

## III. DESCRIPTION OF THE METHOD

Based on the presented research gap, this paper presents a pipeline for structured DPP for ML-applications in production. The concept consisting of eight iterative steps can be taken from Fig. 1.

Based on available production data, requirements of the given use-case are determined. The next step is to determine data quality, from which DPP-methods to be applied are derived. DPP-steps are divided into integration (step 3) up to augmentation and balancing (step 7). In these steps, the large number of DPP-methods is classified and methods most frequently used in production are highlighted. After each step, quality checks (QC) of the data are performed. ML-algorithms are applied in step eight after a final quality check. In the following, each step of the concept will be presented in detail.

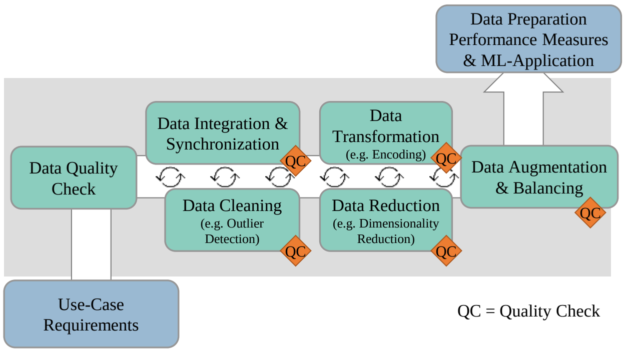

*Fig. 1. Concept for Structured Data Preparation Pipeline*

### A. Use-Case Requirements

In a first step, requirements are determined, since the selection of DPP-methods is highly dependent on present use-cases. Use-cases in application areas such as "Product", "Process" and "Machines & Assets" reveal different, versatile requirements for DPP [3]. DPP is influenced by data set characteristics, ML-algorithm properties, external and use-case specific requirements.

With respect to the data set, numerous different properties influence the selection of DPP-methods. Criteria to be considered are structured in Fig. 2. These characteristics can be classified into general, data set and target-related requirements.

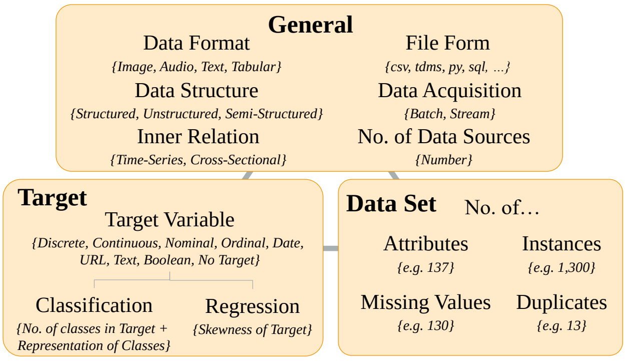

*Fig. 2: Overview of criteria to be considered regarding data set characteristics*

General characteristics cover information about data format (e.g. image) or number of data sources. In addition, inner-relations of the data, either time-series or cross-sectional, impacts DPP. With regard to the target variable, it is essential to know the label balance in case of classification and data skewness for regression tasks. In addition, data set characteristics comprise shape of the data set, duplicates as well as missing values.

Depending on which ML-algorithm is selected and implemented, DPP needs to be designed. Exemplarily, while tree-based algorithms are capable of handling categorical data, artificial neural networks require numerical data. External characteristics to be considered comprise the operating system, programming language and libraries to be used. Aspects such as RAM-usage, disk memory and time budget available play major roles especially for memory-intensive operations during DPP. Requirements that are derived from use-cases influence DPP, however, depend highly on given circumstances and are not simply reproducible. The output of the first step is a transparency about the requirements on DPP.

### *B. Data Quality Check*

While the requirement determination provides an indication of which criteria need to be taken into account, its values are identified by performing an initial data quality check. The goal is to assess accuracy, uniformity completeness, consistency and currentness of the data [22]. First, general information such as the number of sources, format and inner-relation of the data need to be determined by loading data of different sources. Then, the quality of data set and target variable can be checked. Exemplarily, a common tool for determining quality of tabular data sets is pandas profiling, which also calculates correlations of each attribute and provides an overview, which attributes to be rejected [23]. Moreover, measures of location and dispersion are calculated. The output of an initial data quality check is the knowledge about the DPP-steps to be performed.

### *C. Data Integration & Synchronization*

Based on knowledge about data quality, data is integrated enabling an efficient and performant DPP. It comprises the integration of information from different data sources with different data structures into a uniform data base. Data acquired in production is either time series or cross-sectional data. Inner-relations of the data highly influences the integration. Two main integration procedures exist. While a horizontal integration adds further attributes such as new sensors to the data set, for vertical integration, instances are concatenated to the data set when more data is being generated over processing times. Data integration requires production expert knowledge about existing data sources and structures.

In production, time-series data is often acquired that requires synchronization of sensors with different sampling rates, latencies or delays of measurement start. If two independent sensors exhibit different start times of measurement, one time series is shifted relative to the referenced time series. Relative time shifts also apply in case of latency, i.e. the time difference caused by the transmission medium. Further, a sampling rate change is performed to eliminate asynchrony caused by different sensor sampling rates. In this step, a general sampling rate is defined, which is applied to all sensor data sets. The determination of the general sampling rate can be based on most frequent, lowest or highest and self-selected sampling rate. The selected sampling rate decides whether sensor data sets are reduced or augmented.

Finally, it is checked whether performed methods yield the desired success by performing data quality checks. In case of integration, this is achieved by printing data set's shape and comparing time stamps. The output of this step is an integrated data set ready for further preparation.

### *D. Data Cleaning*

Starting with an integrated data set, data generally needs to be cleaned. Cleaning can be classified into missing data, outlier and noisy data handling.

In the vast majority of real-world production data sets, missing values, outliers and noisy data are present, which leads to loss in efficiency and poor performance of data analysis. Reasons range from equipment errors over incorrect measurements to wrong manual data entries. Depending on whether data is missing completely at random (MCAR), missing at random (MAR) or missing not at random (MNAR), missing data can be handled. Missing data can either be ignored, deleted or imputed. Ignoring missing data leads to an unbiased modelling, yet can only be applied if percentage of missing values is low. Missing data can be removed by deleting rows or columns or performing pairwise deletion. Eliminating missing values by deletion can be considered if enough instances or attributes exist in order not to lose too much information. Most often used approach in handling missing values is imputation, since meaningful information is maintained. Especially in production, information is maintained, if only few data sets are available as historical data. The following list shows an excerpt of possible imputation methods

- Univariate: mean, mode, median, constant
- Multivariate: linear & stochastic regression
- Interpolation: linear, last & next observation
- ML-based: k-nearest neighbour, k-means clustering
- Multiple imputation,
- Expectation maximization

Consequently, MCAR-data can be ignored if number of missing values does not exceed threshold value, deleted in case of many missing values and imputed if missing data is spread over many attributes. MNAR-data need to be avoided since it has the potential to ruin analysis, whereas MAR-data should be imputed. Quality of the resulting data set needs to be eventually checked.

In addition, outliers can have hazardous impact on modelling. Outliers are extreme values that deviate from other observations and can be classified into global, contextual and collective outliers. The detection of outliers can be through univariate or multivariate statistical methods like Boxplots or Scatter Plots. Further detection approaches are nearest neighbour or ML-based. Handling outliers are in principle comparable to missing data handling, i.e. outliers can be ignored, deleted or imputed.

Besides missing data and outliers, noise can be observed in production data sets such as duplicates, inconsistent or unimportant values as well as very volatile data. Duplicates, constant values and correlations between the features need to be removed, since attributes bring no further information for modelling.

### *E. Data Transformation*

Once data is integrated and cleaned, it needs to be transformed. In real world data sets, data comes in different data types (e.g. different machine names, temperatures in -5°C or 5°C), ranges and distributions (e.g. binomial, multimodal). Moreover, numerical data may exhibit high cardinality.

For unifying data types and to improve analysis, data is encoded. It can be distinguished between classic, Bayesian and contrast encoders. Among others, classic encoder range from OneHot over label to Hashing or Binary encoders. Using Label encoders is meaningful for ordinal data, whereas OneHot encoders should be applied in case of nominal data. However, if cardinality of nominal attribute is high, too many dimensions may be added to the data set. In these cases, Hashing or Binary encoders should be applied. Commonly used encoders are Bayesian-based such as Target or LeaveOneOut. These methods are considering the target variable and its distribution.

Data can be in different ranges. For instance, data is represented in spindle speed with revolutions per minute as unit. Values can range from 800 rpm to 1,400 rpm, whereas the work piece temperature is from 0°C to 200°C. ML-algorithms may assess higher numbers as more important. Thus, feature scaling is required to ensure that attributes are on same scales. Common methods for feature scaling in production are Z-Score Standardization, rescaling by using Min-Max-Scaler or Robust Scaler. Thereby, many methods can also be applied in different DPP-steps. For instance, Z-Score Standardization is both used for outlier detection and feature scaling.

Usually, normal distributions are desired for modelling. However, production data is often present in skewed distribution. For normalizing skewed distributions, Square Root, Cube Root or log-transform are methods to be chosen. If distributions are highly skewed, Box-Cox or Yeo-Johnson transformations are selected.

Lastly, numerical attributes with high cardinality can be discretized, i.e. high amount of instances that can be combined without losing meaningful information. Data discretization aims to mapping numeric values to reduced subset of discrete or nominal values. Most popular approaches for discretizing data are binning methods based on either Equal Width or Equal Frequency. Finally, the effectiveness of each method is verified by a data quality check. The output is a transformed data set.

### *F. Data Reduction*

As more sensors are connected, more data is generated and more instances and features are added to data sets. Adding more features will end up in data sets being sparse. As dimensions grow, dimension space increases exponentially, which is also stated as curse of dimensionality. After a certain point, adding new features or sensors in production degrades the performance of ML-algorithms resulting in the necessity of reducing the number of dimensions.

One approach is to perform dimensionality reduction. Based on existing features, a new set of features is created that maintain a high percentage of original information. Popular methods are Principal Component Analysis (PCA) or Linear Discriminant Analysis. For applying PCA, a previous feature scaling is required. Besides component-based reduction techniques such as PCA, dimensionality can be decreased based on projections. Methods range from Locally Linear Embedding over Multidimensional Scaling to t-distributed Stochastic Neighbour Embedding (t-SNE). Furthermore, autoencoders represent a ML-based method for reducing the number of attributes.

Another approach is to select features. Instead of creating a reduced number of features out of existing ones, specific features are selected or features are removed from the data set. Methods can be classified into filter, wrapper and embedded approaches. Attributes can be filtered based on features with low variances or high correlation between features. In the wrapper approach, features are selected by identifying the impact of a certain feature on the performance of a baseline model that is trained. Forward Feature Selection, Backward Feature Elimination as well as Recursive Feature Elimination represent common methods for performing wrapper approaches. Lastly, embedded approaches perform feature selection through regularization or the computation of feature importance.

Besides selecting features, instances can also be selected to reduce the number of observations. One challenge is to select stratified and representative samples. Models trained on representative data samples can easily be scaled up. It can be distinguished between filter and wrapper approaches. However, since the number of instances is huge in reality, both filter and wrapper methods take too long for being competitive alternatives in production leading to manual sampling as commonly used approach. Lastly, data quality is checked. The output is a reduced data set in features and instances.

### *G. Data Augmentation & Balancing*

For given data sets, the number of features or instances can also be too low, leading to the requirement of augmenting data in order to enlarge the data set and increase its variation. In tabular data sets, features can be added through domain specific knowledge. Based on existing features, new features can be derived providing ML-models with new meaningful information. For instance, products, quotients or powers can be computed between attributes. Moreover, two or more columns can be concatenated into one. Besides augmenting features, instances can be augmented through inserting random noise into the data set.

In classification tasks, the classes are usually not uniformly distributed. If the product quality is predicted to be in or out of specification, usually the vast majority is labelled as in-specification. This class underrepresentation poses a major challenge, since algorithms benefit from balanced data. In case of high underrepresentation, algorithms may not learn patterns but only guess major output classes, i.e. products that are in-specification. Thus, data is balanced, which can be classified in over-, under- and hybrid sampling. In oversampling through e.g. SMOTE or ADASYN, synthetic samples from the minority class are created. For undersampling, Tomek Links or Cluster Centroids represent common methods. Hybrid sampling combines over- and undersampling. Exemplarily, minority classes can be oversampled and afterwards, majority classes undersampled. Quality checks verify methods performance leading to an augmented and balanced data set ready for modelling.

### *H. DPP-Performance Measures & ML-Application*

Besides a final data quality check, data quality is determined by assessing the performance of trained and tested ML-models. Consequently, ML-models are implemented to eventually determine the quality of the prepared data set. For this, a suitable ML-algorithm needs to be selected and built. The final ML-model performance is assessed according to proper metrics. Output of this step is an assessed ML-model performance and a transparency about final data quality.

## IV. RESULTS AND DISCUSSIONS

The structured DPP-pipeline is validated and evaluated based on two production use-cases. In a first use-case, the focus is put on manufacturing of semiconductors. Based on process and environment data within a process chain, the goal is to predict, whether products are in or out of specification at the end. The tabular data set comprises 1,567 observations and 594 features. By performing pandas profiling, 4 % missing values are identified and a rejection of 210 features is recommended due to high correlations and high amount of missing values per feature. Besides this, 254 features are numerical, whereas 130 attributes are categorical. Features reveal different ranges, while the target variable is imbalanced in a ratio of 14:1 (Step B). Based on use-case description and data quality check, necessary steps of DPP can be derived, while unnecessary steps are automatically ignored. Missing data is removed if ratio of missing values is higher than 40 %, otherwise, data is imputed. Noisy, duplicate values and columns with only one unique value are dropped (Step D). Further, data is transformed by applying standard scaler and target encoding (Step E). Subsequently, the number of features are removed through feature selection using backward feature elimination (Step F). Through the imbalance, the data set is balanced based on SMOTE (Step G). A tested random forest classifier achieved an F1-score = 0.9764. It was found that the less DPP is performed, the worse final the model results are. No preparation leads to an F1-score = 0.4721. Additionally, it was shown that the duration of data preparation was significantly reduced. Moreover, different preprocessing levels were benchmarked to quantify the performance of each DPP-method. Thereby, balancing and reduction offered the highest leaps in model performance, whereas basic cleaning steps were necessary to train the model.

The second use-case deals with predictive maintenance, in which tool wear is predicted based on process data that is acquired in various experiments. Through an initial data quality check, three different time-series data sets with different time stamps and frequencies are integrated, resulting in a data set of 14,517,520 instances and 12 numerical attributes. Attributes contain outliers and unscaled data with different frequencies (Step B). Based on these findings, methods to be performed can directly be detected. Initially, data sets are integrated and synchronized based on highest sampling rate to maintain information (Step C). Missing values are linearly interpolated, whereas outliers are detected using Interquartile Range (IQR) and finally removed from the data set (Step D). Since data is in different ranges, features are scaled (Step E). Further attributes are reduced based on low variance filter (Step F). No augmentation or balancing is performed. The tested Gradient Boosting performance achieved an F1-Score = 0.9923. Given the assumption that Gradient Boosting does not overfit and is implemented correctly, data quality can be assessed as high. Different stages of DPP were tested, while no preparation lead to an error in modelling, since data sets were not on uniform data base. Here, it was shown that data reduction offers the highest performance gain.

In conclusion, two data sets with versatile requirements reveal the potential of the pipeline to structuring DPP. It was found that steps E and F of the pipeline can have high impact on final performance. However, many versatile use-cases are still missing for proving the strength. Generally, it was found during the validation that level of preparation comprises a general trade-off. Data quality remains on a low level in case of too little preparation. Too much preparation introduces a bias, since real conditions are no longer represented. The wrong use of DPP-methods introduces further noise, meaning that a proper dose of DPP is required. Another trade-off was detected in training and testing. Much preparation lead to good training results, however, requires the certainty of being able to prepare data in real world production use-cases accordingly. Less DPP enables a faster and less error-prone pipeline. In addition, through conditionality and various requirements, DPP itself remains iterative. More use-cases are required to quantify the strength of the DPP-pipeline.

## V. CONCLUSIONS AND OUTLOOK

Data quality is first and foremost in data-driven analysis. Production data sets comprise many data quality issues making it essential to prepare data. Nowadays, preparation is performed manually, unstructured and takes the vast majority of time in ML-projects. For these reasons, we presented a comprehensive concept for a structured and reusable DPP-pipeline for ML-applications in production. The concept consists of eight iterative steps, starting with the identification of requirements for DPP. Based on a data quality check, the DPP-methods to be performed are determined. DPP-steps can be classified into integration & synchronization, cleaning, transformation, reduction and augmentation & balancing. The prepared data set is finally verified based on ML-model performance. The DPP-pipeline was successfully validated based on two production use-cases with versatile requirements by assessing the final model performance.

Future work will focus on benchmarking hundreds of DPP-methods in the production sector. The automation of the DPP-pipeline represents a great potential for further improvement of DPP. Expert systems can be used to select most appropriate DPP-methods for given use-cases.

## VI. ACKNOWLEDGMENTS

The research in this paper was partially supported by the European Commission through the H2020 project EPIC (https://www.centre-epic.eu/) under grant No. 739592.

## REFERENCES

[1] **Press, G.:** Cleaning Big Data: Most Time-Consuming, Least Enjoyable Data Science Task, Survey Says. Forbes Inc. https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/#4db1b036f637, 2016.

[2] **Reavie, V.:** Do You Know The Difference Between Data Analytics And AI Machine Learning? Forbes Inc. https://www.forbes.com/sites/forbesagencycouncil/2018/08/01/do-you-know-the-difference-between-data-analytics-and-ai-machine-learning/#1de233a35878, 2018

[3] **Krauß, J., Dorißen, J., Mende, H., Frye, M., Schmitt, R.H.:** Machine Learning and Artificial Intelligence in Production: Application Areas and Publicly Available Data Sets, in: Wulfsberg, J.P., Hintze, W., Behrens, B.-A. (Eds.), Production at the leading edge of technology. Springer Berlin Heidelberg, Berlin, Heidelberg, 2019, pp. 493–501.

[4] **Bendiek, S.:** Artificial Intelligence in Europe - Germany, Outlook for 2019 and Beyond: How 307 Major Companies Benefit from AI. Microsoft. https://cloudblogs.microsoft.com/industry-blog/de-de/government/2019/05/17/artificial-intelligence-in-europe-germany-outlook-for-2019-and-beyond/, 2019.

[5] **Kotsiantis, S.B., Kanellopoulos, D., Pintelas, P.E.:** Data Preprocessing for Supervised Learning. In :World Acadamy of Science, Engineering and Technology, 2007.

[6] **Wolff, M.:** Garbage In, Garbage Out: The Importance of Good Data. Medium. https://medium.com/@marybrwolff/garbage-in-garbage-out-the-importance-of-good-data-ce1bb775468e, 2020.

[7] **Tancer, B.**, The 2014 Digital Marketer: An Experian Marketing Services Benchmark and Trend Report. Experian. https://www.experian.co.uk/assets/marketing-services/reports/ report-digital-marketer-2014.pdf, 2013.

[8] **Moore, S.:** Poor quality data weakens an organization's competitive standing and undermines critival business objectives. Gartner. https://www.gartner.com/smarterwithgartner/how-to-stop-data-quality-undermining-your-business/, 2018.

[9] **Vyas, K.:** Why 85% of the Artificial Intelligence Projects Fail? CustomerThink. https://customerthink.com/ why-85-of-the-artificial-intelligence-projects-fail/, 2019.

[10] **García, S., Luengo, J., Herrera, F.:** Data Preprocessing in Data Mining. Springer Switzerland, DOI: 10.1007/978-3-319-10247-4, 2015.

[11] **Han, J., Kamber, M., Pei, J.:** Data Mining: Concepts and Techniques, 3rd ed. Morgan Kaufmann, ISBN-10: 9780123814791, Waltham, MA, USA, 2012.

[12] **Pedregosa, F., Varoquaux, G., Gramfort, A.:** Scikit-learn: Machine Learning in Python: 6. Dataset transformations. scikit learn. https://scikit-learn.org/stable/datatransforms.html, 2020.

[13] **McGinnis, W.:** Categorical Encoding Methods. GitHub. https://github.com/scikit-learn-contrib/category_encoders, 2020.

[14] **Christ, M.:** tsfresh. tsfresh. https://tsfresh.readthedocs.io/en/latest/, 2020.

[15] **Heinisuo, O.-P.:** opencv-python 4.4.0.42: Project description. Python Package Index. https://pypi.org/project/opencv-python/, 2020.

[16] **Willems, K.:** Collecting Data Science Cheat Sheets. towards data science. https://towardsdatascience.com/collecting-data-science-cheat-sheets-d2cdff092855, 2017.

[17] **Willems, K.:** Keras Cheat Sheet: Neural Networks in Python: Make your own neural network with this Keras cheat sheet to deep learning in Python for beginners, with code samples. DataCamp. https://www.datacamp.com/community/blog/keras-cheat-sheet, 2017.

[18] **Willems, K.:** Scikit-Learn Cheat Sheet: Python Machine Learning: A handy scikit-learn cheat sheet to machine learning with Python, including code examples. DataCamp. https://www.datacamp.com/community/blog/ scikit-learn-cheat-sheet, 2017.

[19] **Iliou, T., Konstantopoulou, G., Ntekouli, M., Lymperopoulou, C., Assimakopoulos, K., Galiatsatos, D., Anastassopoulos, G.:** ILIOU Machine Learning Preprocessing Method for Depression Type Prediction, in: Angelov, P., Filev, D., Kasabov, N. (Eds.), Evolving Systems. An Interdisciplinary Journal for Advanced Science and Technology, Issue 1. Springer, 2019, pp. 29–39.

[20] **Alharbi, N., Liang, Y., Wu, D.:** A Data Preprocessing Technique for Gesture Recognition Based on Extended Kalman-Filter, in: IEEE/ACM International Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE), Philadelphia, PA, USA, 2017.

[21] **Huang, J., Li, Y.-F., Xie, M.:** An empirical analysis of data preprocessing for machine learning-based software cost estimation, in: Ruhe, G. (Ed.), Information and Software Technology. Elsevier, 2015, pp. 108–127.

[22] **Batini, C., Scannapieca, M.:** Data Quality: Concepts, Methodologies and Techniques. Springer, 2006.

[23] **Brugmann, S.** Pandas Profiling. GitHub. https://github.com/pandas-profiling/pandas-profiling, 2020