Benchmarking of Machine Learning Models for Tabular Scarce Data

1st Justus Kösters
Center for Applied Data Science
Bielefeld University of Applied Science
Gütersloh, Germany
justus.koesters@fh-bielefeld.de

2nd Marvin Schöne
Center for Applied Data Science
Bielefeld University of Applied Science
Gütersloh, Germany
marvin.schoene@fh-bielefeld.de

3rd Martin Kohlhase
Center for Applied Data Science
Bielefeld University of Applied Sciences
Gütersloh, Germany
martin.kohlhase@fh-bielefeld.de

***Abstract*—Due to their high costs and time requirements, companies are interested in minimizing their laboratory experiments during process or product design. For this, machine learning can be used to extract knowledge from the process or product to predict future designs. Due to the high costs and time requirements, data from laboratory experiments are scarce, so only machine learning algorithms with small hypothesis spaces are suitable to predict such data. In this paper, the performance of Linear and Logistic Regression, Decision Trees, Gaussian Processes and Support Vector Machines on respectively five real world datasets for classification and regression is compared. The Decision Trees have the best and Gaussian Processes the worst overall performance, but the Gaussian Processes show a great potential if adequate hyperparameters are selected. For the analyzed data and the chosen hyperparameters, the Gaussian Processes tend to overfit, whereas the Support Vector Machines tend to underfit. Linear and Logistic Regression offer a good tradeoff between complexity and performance, producing results comparable to Decision Trees.**

***Index Terms*—tabular scarce data, industrial design, supervised machine learning models**

I. Introduction

For the industrial design of processes or products, companies often carry out laboratory experiments. These experiments generate high-quality data enriched by domain-specific knowledge, making it profitable to use the knowledge in the data for future assistance in process or product design [1]. This can be done using *machine learning* [2], where the knowledge in the data is algorithmically aggregated into a data-based model, also called a machine learning (ML) model. As a result, future laboratory experiments can be reduced and domain-specific knowledge is preserved. While many ML algorithms nowadays require a large amount of labeled data, there are some applications where data collection is very expensive or time-consuming, e.g. laboratory experiments [3]. Data from laboratory experiments are therefore rather scarce, and proper ML models that can handle scarce data need to be identified. In order to expand knowledge about the process or product being designed, the ML model can be further trained using active learning, which additionally reduces further experiments while improving the aggregated knowledge within the ML model [4].

This work was funded by the Ministry of Economic Affairs, Innovation, Digitalisation and Energy of the State of North Rhine-Westphalia (MWIDE) within the project AI4ScaDa, grant number 005-2111-0015.

In this paper, potentially suitable ML models for tabular scarce data on real world datasets are benchmarked so that the best-performing ML models for an application to laboratory-collected data of a real industrial process can be identified. This is done for both regression and classification. Furthermore, since it is intend to use the results of this paper to develop new model-specific approaches to select the query in active learning, only ML algorithms that basically can be used for both regression and classification are compared. While many scientific papers on scarce data deal with semi-supervised and unsupervised ML methods [5], this paper only focuses on supervised ML methods.

II. Related Work and Fundamentals

In order to select potentially suitable ML models for applications with scarce data, it is necessary to consider the cause and composition of errors that models make and what influence the size of a dataset has on them (Section II-A). With this knowledge and results of previous work with scarce data, appropriate ML models can be selected for our benchmarking, which is described in Section II-B.

A. Bias-Variance Tradeoff with Scarce Data

In supervised learning, ML models aim to achieve sufficient generalization from training data so that unknown data with a comparable data distribution can be predicted with high accuracy. If there is no sufficient generalization, the models deal with over- and underfitting. In the case of overfitting, the hypothesis assumed about the data is too complex (e.g. too high polynomial degree). As a consequence, the model tends to learn not only the relationships of the data but also their noise [6]–[8]. Underfitting appears if the model poorly fits training data due to an oversimplified or inappropriate hypothesis. In this case, the model is not able to learn the relationships nor the noise in the data. The overall prediction error can be split into *bias* and *variance error*, which calls the arrangement between over- and underfitting *bias-variance tradeoff*. Bias appears with a mismatch between the given data with the model structure, while variance is a measure for the sensibility to variations in the training data [9]. The

bias and variance composition on an error can be received from a decomposition of the mean-squared-error

$$\mathrm{E}\Big[\big(h(\mathbf{x}) - \mathrm{E}[y \,|\, \mathbf{x}]\big)^2\Big] = \underbrace{\big(\mathrm{E}[h(\mathbf{x})] - \mathrm{E}[y \,|\, \mathbf{x}]\big)^2}_{\text{squared bias}} + \underbrace{\mathrm{E}\Big[\big(h(\mathbf{x}) - \mathrm{E}[h(\mathbf{x})]\big)^2\Big]}_{\text{variance}} \quad (1)$$

where $h(\mathbf{x})$ is the prediction of a ML model for a sample $\mathbf{x} = (x_1 \ldots x_M)$ containing M features with the corresponding label y. As the complexity of a model continues to increase above a certain complexity level, the bias is typically not increasing or decreasing, while the variance is instead increasing [10]. It can be inferred that simple model hypotheses tend to have higher bias and complex models have higher variance due to their more flexible predictions [11].

Previous studies [12], [13] on classification with *Decision Trees* for large datasets have shown that both the bias- and variance-error decrease with increasing training data size, which lowers the general prediction error. They analyzed the composition of the bias- and variance-error for different training dataset sizes ranging from $N = 32$ to over $N = 30000$ samples on 7 different datasets. The proportion of variance on the overall prediction error is at the maximum for smallest datasets. Depending on the analyzed datasets, the variance has a higher influence on the overall error than the bias if the amount of training data is small [13].

Combining the facts that

- the variance-error increases with higher model complexity [11] and
- the variance-error has a major influence on the error for small datasets [13]

shows, that models with low complexities should be used for predictions based on scarce data. Next, some potentially appropriate ML models for applications on scarce data are discussed.

B. ML Models for Scarce Data

There are only a few articles [14]–[17] that apply ML models to tabular scarce data. The articles [14] and [15] deal with regression tasks while [16] focusses on classification tasks. In [17], one classification and one regression task are mentioned respectively. Used ML models for regression are *Linear Regression*, *Gaussian Process Regression*, *Lasso-Regression*, *Ridge-Regression*, *Support Vector Machines* and *Neural Networks*. The classification tasks are solved with *Spatial Poisson Regression* and Decision Trees. Except Support Vector Machines and Neural Networks, these are simple models with low complex hypothesis spaces which corresponds to the results of Section II-A. Based on these results, the following four different types of ML models applicable to both classification and regression are selected for the benchmarking.

1) Linear and Logistic Regression: Both Linear and Logistic Regression are parametric models. In Linear Regression, the model $\hat{y}(\mathbf{x})$ is defined by a linear combination

$$\hat{y}(\mathbf{x}) = \beta_0 + \sum_{m=1}^{M} x_m \beta_m \quad (2)$$

with the offset β_0 and the M slopes β_m to weight each feature x_m of the sample $\mathbf{x}$. The $M + 1$ coefficients β_0 and $\boldsymbol{\beta} = (\beta_1 \ldots \beta_M)^{\mathrm{T}}$ are estimated by *Least Squares*, a linear optimization, which minimizes the residual sum of squares

$$RSS(\beta_0, \boldsymbol{\beta}) = \sum_{i=1}^{N} (y_i - \mathbf{x}_i \boldsymbol{\beta} - \beta_0)^2 \quad (3)$$

of the Linear Regression model [18]. The Logistic Regression model is a generalization of Linear Regression by a logistic discrimination function

$$\hat{y}(\mathbf{x}) = P(c_1|\mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{x}_i \boldsymbol{\beta} - \beta_0)} \quad (4)$$

to separate two different classes c_1 and c_2 by the conditional probability $P(c_1|\mathbf{x})$. Using multiple models of (4) and the *Softmax* function, Logistic Regression can be extended to multi-class predictions. Since a direct solution for the coefficients β_0 and $\boldsymbol{\beta}$ is not possible by Least Squares, the Logistic Regression model is iteratively fitted by a gradient descent algorithm maximizing the likelihood or log likelihood [19].

2) Gaussian Processes: Gaussian Processes [20] are *Bayesian* models and thus use conditional probabilities to describe process behavior. The main idea is to measure the similarity between data points using a kernel function. If two data points in the input space are very similar in the sense of this kernel, it follows that the corresponding output values are also very similar. The model behavior is influenced by hyperparameters such as the bandwidth of the kernel functions and the estimated noise level in the training data. While the hyperparameters for the kernel functions are determined by an optimizer that maximizes the marginal likelihood function, the noise level must be estimated in advance independently of the optimizer. As non-parametric models, Gaussian Processes offer a high degree of flexibility and at the same time allow to specify confidence intervals for the model predictions due to the underlying probability distributions. In their basic form, they can only be used for regression, but can be extended to classification using a logistic function similar to (3). In this case, the model output of the Gaussian Processes is incorporated into the exponential function of the logistic function as a latent variable. However, to model the probability that $\mathbf{x}$ belongs to a certain class, integrals must be approximated by *Laplace* approximation [20].

3) Decision Trees: Decision Trees are non-parametric supervised ML models that use a recursive partitioning of the data space performed by a *divide and conquer* algorithm [21]. They consist of nodes connected to their successors by edges, resulting in a top-down directional acyclic graph. The graph starts with the root node, which has following internal

nodes and leaf nodes. In every root node or internal node, the partitioning of the data is performed on an feature to smaller subsets. The leaf nodes contain local models that approximate the relationship of the data within the corresponding subset, and have no child nodes. The performance of a tree depends on the partitioning or splitting quality in the respective nodes, typically defined by a selected feature x_m and a threshold α_m at which the input space is axis-diagonally split within the subspace of x_m [24]. There are many popular algorithms to generate Decision Trees like CART [25], CHAID [26], ID3 [27], C4.5 [28] and GUIDE [29]. A problem that occurs in many tree algorithms and is only solved by GUIDE is the biased feature selection for splitting due to the amount of expressions of the features. The GUIDE algorithm also increases the performance by analyzing interactions between different features, is able to solve regression and classification tasks with using both numerical and categorical features simultaneously, and can estimate many different complex local models [29]. Therefore GUIDE is used in this benchmarking.

4) Support Vector Machines: Support Vector Machines are decision machines which do not provide posterior probabilities. They are used to solve regression and classification tasks. The aim of Support Vector Machines for classification is to create a decision boundary that separates binary classes. For multi-class classification, they work with an one-vs-one approach. The decision boundary is chosen through maximizing the *margin* which defines the smallest distance between the decision boundary and any samples. Therefore the so-called kernel trick is used, which applies kernels to the input data to map the Support Vector Machine inputs to a high-dimensional feature space in which a linear separation of the two classes is searched for. For the regression tasks, a Support Vector Machine searches for a best fitting line within a predefined threshold [30].

III. Experimental Analysis

To compare the performance of the four selected model types in both classification and regression, an extensive benchmarking on real world datasets is performed, whose setup is described in more detail in Section III-A. The results are presented in Section III-B, which are further discussed in more detail in Section III-C.

A. Experimental Set Up

In this benchmarking, ten open access datasets for respectively five classification and regression tasks are used. A description of the datasets according to size and amount of features is shown in Table I. Expect the *Titanic* dataset with $N = 712$, all datasets contain a small number of samples between $N = 103$ and $N = 308$. The number of features varies from $M = 4$ to $M = 24$ where only the datasets Titanic and *Automobile* containing categorical features in addition to the numerical features. The categorical features of these two datasets are *one-hot* encoded so that the categorical features can be used for each ML model, even if it would not be necessary for each model (e.g. for the GUIDE Decision Trees). The advantage of one-hot encoding is that weight based models can simply learn a weight for every categorical unique value. However, this also increases the dimensionality of the two datasets (up to $M = 32$ for the Automobile dataset) which leads to sparse matrices.

For every dataset 25 independent random train-test splits are created with a training size of $N_{\text{train}} = 0.8N$. Except the Decision Trees, all ML models are trained by the *scikit-learn v. 1.2.0* package for Python [22]. For classification *GaussianProcessClassifier* (GPC), *LogisticRegression* (LogR) and *SVC* and for regression *GaussianProcessRegressor* (GPR), *LinearRegression* (LinR) and *SVR* are used. The models are mostly trained and optimized with their default settings, which can be taken from the individual manuals. All kernel methods are using *Radial Basis Function Kernels* and only the maximum number of iterations for logR and the noise level of GPR are increased to 10000 and 1 respectively. The noise level adjustment is necessary due to the high noise level of the datasets and the fact that the implementation in Python only optimizes the kernel parameters and not estimate the noise level. The GUIDE Decision Trees for classification (DTC) and regression (DTR) are estimated by a compiled binary (version 37.3) from [23]. For prediction, the trees are using the most frequently occurring class in a leaf (classification) and multiple linear models (regression). Moreover, they are post-pruned by a 10-fold crossvalidation with a 0.25-SE-rule.

The models are tested against a specific metric in each run and averaged over the 25 independent runs. However, in order to analyze the models in therms of over- and underfitting, the metric is evaluated with both training and test data. The performance for classification is measured by the (multi-class) F_1-score, which is a harmonic mean from the confusion matrix and not heavily influenced by unbalanced datasets. The F_1-score is calculated by

$$F_1 = \frac{2TP}{2TP + FN + FP} \tag{5}$$

where TP are the true positive, FN the false negative and FP the false positive classifications. The F_1-score takes values between $[0, 1]$ [24]. To calculate the multi-class F_1-score, (5) is calculated separately for each class in a dataset and averaged. The performance of the regression models is compared by a normalized version of the mean absolute error

$$\text{nMAE}_{c,d}^{(\cdot)} = \frac{\text{MAE}_{c,d}^{(\cdot)}}{\max\limits_{\mathcal{C}}(\text{MAE}_{b,d}^{\text{test}})} \quad . \tag{6}$$

The mean absolute error $\text{MAE}_{c,d}^{(\cdot)}$ for model $c \in \mathcal{C}$ with $\mathcal{C} = \{\text{GPR}, \text{DTR}, \text{LinR}, \text{SVR}\}$ of dataset $d \in \{\text{DS } 6, \ldots, \text{DS } 10\}$ is normalized for both training and test data by the maximum mean absolute error $\text{MAE}_{b,d}^{\text{test}}$ across all four models $b \in \mathcal{C}$, so that $\text{nMAE}_{c,d}^{(\cdot)}$ also takes values in an interval $[0, 1]$. The dummy $(\cdot)$ is replaced by either train or test.

B. Results

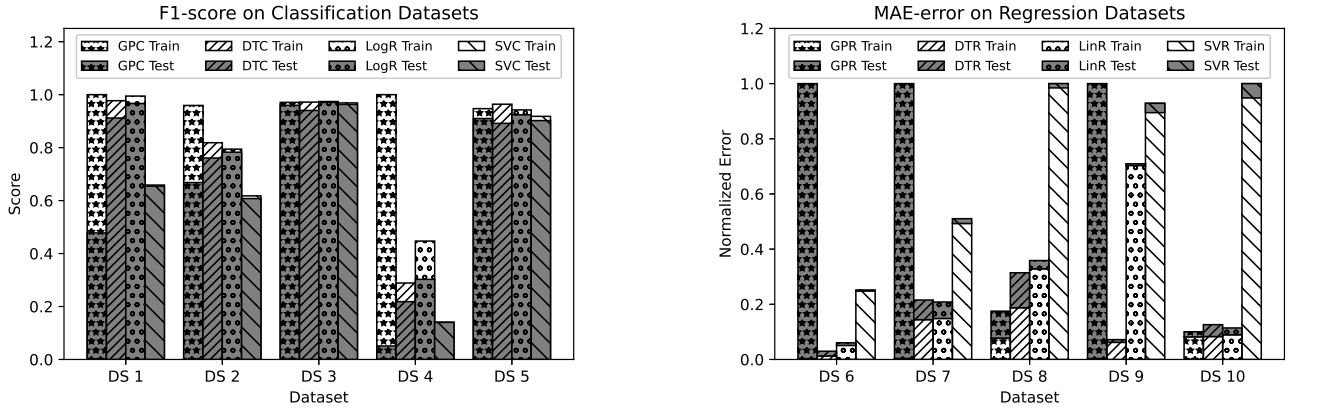
The results of the benchmarking are shown in Figure 1. There are two bar plots which display the respective prediction

TABLE I: Description of benchmarking datasets

dataset ID	dataset name	# numerical variables	# categorical variables	# target classes	# datapoints	source
Classification						
DS 1	Wine	13	0	3_{a}	178	[31]
DS 2	Titanic	4	3	2_{a}	712	[32]
DS 3	Iris	4	0	3_{b}	150	[33]
DS 4	Heart Disease	13	0	5_{a}	297	[34]
DS 5	Wheat Seeds	7	0	3_{b}	199	[35]
Regression						
DS 6	Body Fat	14	0	-	200	[36]
DS 7	Automobile	13	6	-	159	[37]
DS 8	Concrete Strength	7	0	-	103	[38]
DS 9	Yacht Hydrodynamics	6	0	-	308	[39]
DS 10	Tecator	24	0	-	240	[40]

$_{\mathrm{a}}$ Unbalanced dataset.

$_{\mathrm{b}}$ Balanced dataset.



(a) Results for the classification task. High values indicate good results.

(b) Results for the regression task. Low values indicate good results.

Fig. 1: Prediction quality of benchmarked ML models for the different datasets

scores for the ML models on the different datasets, where the light bars show the score for training data and the dark bars for test data. Due to the different scaling of the score for training and test data, the results of training data for classification are displayed in the background and for regression in the foreground.

Analyzing the classification results from Figure 1 (a), it can be recognized that each classifier on DS 3 and DS 5 achieves good generalization with comparable scores, which is defined by similar F_1-scores of both training and test data close to 1. On DS 1, the DTC and LogR classifiers perform well, while the GPC has issues with overfitting which can be recognized from a much higher training score than the test score. In contrast, the SVC has comparable but low scores near 0.6 for both training and test data. A similar behavior is observed at DS 2. DTC and LogR perform best at a F_1-score of about 0.8, but not as well as in DS 1 (F_1-score ≈ 0.95). GPC is again overfitting and SVC is not able to capture the relationships in the data well. On DS 4, all ML models perform poorly. The GPC has a high score for training data but a much lower score for test data and thus overfits dramatically.

The results in Figure 1 (b) describe the performance of the ML models for the regression tasks. In contrast to the F_1-score, the models perform best when the nMAE is minimal. Therefore, the GPR has poor performance at DS 6, DS 7, and DS 9 where it is purely based on overfitting, but it performs well at DS 10 and has a small error at DS 8 with a little overfitting. DTR and LinR also have a small error at DS 8 and DS 10 and even at DS 6 and DS 7. SVR is again unable to capture the relationships in the data well, resulting in more than twice the errors of DTR and LinR for all datasets. However, at DS 6, DS 7, and DS 9 SVR achieves smaller errors than GPR. DS 9 is most noticeable, where the LinR, along with SVM and GPR, has a high error compared to the DTR. After presenting the results in this section, the reasons will be addressed in the following section.

C. Discussion

As noted in the previous Section, the performance of all models is worst on DS 4. This can be caused by 5 different

target classes which are also unbalanced. These results are strengthened by a comparison with the scores achieved on the balanced datasets DS 3 and DS 5 with fewer classes. Despite DS 2 contains only two balanced classes and many samples, the ML models achieve comparably lower results. These results are suspected to be caused by one-hot encoding, generating 8 additional sparse features. The better results of DTC and LogR on the comparatively higher dimensional datasets DS 1, DS 2 and DS 4 can be explained by the naturally given feature selection of Decision Trees and the L2 regularization used in the Python implementation of LogR. It is surprising that LogR performs slightly better than DTC, which can be justified by the trivial local models of DTC. However, GUIDE can also train Decision Trees with more complex local models, e.g. local *Nearest-Neighbor* models [41]. Moreover, with GUIDE the one-hot encoding is not even necessary which could increase the performance on DS 2. The kernel models GPC and SVC don't perform on these datasets as expected. For SVC, performance might be improved by some adjustments of the kernel parameters, which, unlike GPC, are not optimized during training. In therms of GPC, the noise level incorporated into the kernel matrix could be responsible for the poor results. If the assumed noise level is too low, the model overfitts [18], which can be clearly recognized in the scores for DS 1 and DS 4. As far as we know, the noise level can only be adjusted for GPR and not for GPC.

The results for the regression tasks are similar to those for the classification tasks. Although the noise level was increased by a magnitude of ten, dramatic overfitting of the GPR occurs in three datasets. In contrast, for DS 8 and DS 10 GPR outperforms the other models, which demonstrates both the sensitivity of the noise level and the efficiency of GPR if this hyperparameter is chosen appropriate. Similar results were obtained in [42]. SVR is in general not compatible with DTR and LinR, and also not with GPR unless it is overfitted. It is assumed that the hyperparameter for the scaling of the internal L2 regularization is too high by default, so that the model is not flexible enough (underfitting). However, this has not been verified. DTR and LinR perform very similar except at DS 9, where one feature has a strong non-linear dependence on the target that cannot be represented by a linear model.

To sum up, Decision Trees trained with the GUIDE algorithm and Linear/Logistic Regression models perform well on these small datasets. Gaussian Processes can perform well, but are strongly affected by the given noise level, which makes them responsive to overfitting. The Support Vector Machines behave exactly the opposite way in this benchmarking and tend to underfit.

IV. Conclusion

In this paper, a benchmarking of ML models for classification and regression on real world scarce datasets was performed. The objective of the benchmarking was to identify the best-performing model type for future classification and regression applications to laboratory-collected data from a real industrial process. After theoretically deriving the requirements for ML models for scarce data from the bias-variance error decomposition, a literature survey on applications to tabular scarce data was conducted. Based on these results, the following ML models suitable for both classification and regression tasks were selected: (i) Linear/Logistic Regression, (ii) Decision Trees, (iii) Gaussian Processes and (iv) Support Vector Machines. The benchmarking was performed with ten real world datasets, respectively five classification and regression datasets, including 255 samples on average. For every prediction task (classification/regression) there is one dataset that contains one-hot encoded categorical features in addition to numerical features. In the benchmarking, Decision Trees trained with the GUIDE algorithm and Linear/Logistic Regression models achieved the best results for both classification an regression. Gaussian Processes only achieve good results if a certain hyperparameter, the noise level, is carefully adjusted to the data. For this reason, Gaussian Processes have overfitted to most datasets. The Support Vector Machines behaved the opposite way and tended to underfit, for which the cause could not be clearly identified.

Based on the results of the benchmarking, Decision Trees are taken for the applications to the industrial process. Apart from the good and robust performance in this benchmarking, they (i) do not need hyperparameter optimization, (ii) can incorporate categorical features without any preprocessing, (iii) are intrinsic interpretable and (iv) offer many advantages for model-specific active learning strategies due to their local model structure. Furthermore, the classification by Decision Trees with more complex local models (e.g. Logistic Regression models) is investigated and the benchmarking will be extended with more datasets to analyze the impact of dataset dimensionality and size on the different ML models. However, due to the very good performance on some datasets and the ability to explicitly represent model uncertainties, Gaussian Processes are also further explored for the application to the industrial process, especially with respect to the selection of an appropriate noise level.

References

[1] K. Guo et al., "Artificial intelligence and machine learning in design of mechanical materials." Materials Horizons, Vol. 8, pp. 1153-1172, 2021.

[2] P. Larranaga et al., "Industrial Applications of Machine Learning." Data Mining and Knowledge Series. Boca Raton, Florida: CRC Press, 2019.

[3] A. Adadi, "A survey on data-efficient algorithms in big data era." Journal of Big Data, Vol. 8, 2021.

[4] A. Tharwat and W. Schenck, "A Survey on Active Learning: State-of-the-Art, Practical Challenges and Research Directions." Mathematics, Vol. 11(4), 2023.

[5] G. Qia and J. Luo, "Small Data Challenges in Big Data Era: A Survey of Recent Progress on Unsupervised and Semi-Supervised Methods." IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 44(4), pp. 2168-2187, 2022.

[6] L. von Ruden et al., "Informed Machine Learning - A Taxonomy and Survey of Integrating Prior Knowledge into Learning Systems." IEEE Transactions on Knowledge and Data Engineering PP (99), 2021.

[7] K. Choo et al., "Machine Learning kompakt, Ein Einstieg für Studierende der Naturwissenschaften" Wiesbaden: Springer, 2020.

[8] J. Jiang, K. Lerman and E. Ferrara, "Zero-shot meta-learning for small-scale data from human subjects." arXiv preprint, arXiv:2203.16309, 2022.

[9] Z. Yang et al., "Rethinking Bias-Variance Trade-off for Generalization of Neural Networks" Proceedings 37th International Conference on Machine Learning, Vienna, 2020.

[10] S. Geman, E. Bienenstock and R. Doursat, "Neural networks and the bias/variance dilemma." Neural Computation, Vol 4, pp. 1–58, 1992.

[11] E. Briscoe and J. Feldman, "Conceptual Complexity and the Bias-Variance Tradeoff." Proceedings Annual Meeting of the Cognitive Science Society, Vol 28, 2006.

[12] D. Brain and G. Webb, "On the effect of data set size on bias and variance in classification learning." Proceedings Fourth Australian Knowledge Acquisition Workshop, pp. 117–128, 1999.

[13] D. Brain and G. Webb, "The need for low bias algorithms in classification learning from large data sets." Principles of Data Mining and Knowledge Discovery, Vol 6, pp.62–73, 2002.

[14] J. Lee et al., "Prediction model of band-gap for AX binary compounds by combination of density functional theory calculations and machine learning techniques." Physical Review B, Vol 93, 2016.

[15] Y. Zhang and C. Ling, "A strategy to apply machine learning to small datasets in materials science." Computational Materials, Vol 4:25, 2018.

[16] C. Squarzoni-Diaw et al., "Using a participatory qualitative risk assessment to estimate the risk of introduction and spread of transboundary animal diseases in scarce-data environments, A Spatial Qualitative Risk Analysis applied to foot-and-mouth disease in Tunisia 2014-2019." Transbound Emerg Dis, Vol 68, pp.1966–1978, 2021.

[17] T. Shaikhina et al., "Machine Learning for Predictive Modelling based on Small Data in Biomedical Engineering." IFAC-PapersOnLine, Vol 48-20, pp.469–474, 2015.

[18] O. Nelles, "Nonlinear System Identification." Springer International Publishing, 2020.

[19] E. Alpaydin, "Introduction to Machine Learning." The MIT Press, Cambridge, 2010.

[20] C. Rasmussen and C. Williams, "Gaussian Processes for Machine Learning." MIT Press, 2006.

[21] W. Loh, "Classification and Regression Trees." Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, Vol. 1, pp. 14-23, 2011.

[22] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python" Journal of Machine Learning Research, Vol. 12, pp. 2825-2830, 2011.

[23] W. Loh, "GUIDE Classification and Regression Trees and Forests (version 41.1)." Department of Statistics, https://pages.stat.wisc.edu/ loh/guide.html, 2023, Last accessed 30 March 2023.

[24] A. Géron, "Hands-On Machine Learning with Scikit-Learn, Keras and Tensor Flow: Concepts, Tools and Techniques to build Intelligent Systems." O'Reilly Media, Vol. 2, 2019.

[25] L. Breiman et al., "Classification and Regression Trees." Monterey: Wandsworth and Brools, 1984.

[26] J. Sonquist and J. Morgan, "The detection of interaction effects: A report on a computer program for the selection of optimal combinations of explanatory variables." Surevey Research Center, Institute for Social Research, University of Michigan, Vol. 35, 1964.

[27] J. Ross Quinlan, "Induction of decision trees." Machine Learning, Vol. 1(1), pp. 81-106, Springer, 1986.

[28] J. Ross Quinlan, "C 4.5: Programs for Machine Learning." Morgan Kaufmann, San Matex, California, 1993.

[29] W. Loh, "Regression Trees with Unbiased Variable Selection and Interaction Detection." Statista sinica, Vol. 12, pp. 361-386, 2002.

[30] C. Bishop, "Pattern recognition and machine learning." New York: Springer, 2006.

[31] M. Forina et al., "PARVUS, An extendible package for data exploration, classification and correlation." Via Brigata Salerno, 16147 Genoa, Italy: Institute of pharmaceutical and food analysis and technologies, 1998.

[32] J. Li, W. Cukierski, "Titanic - Machine Learning from Disaster." Kaggle, https://kaggle.com/competitions/titanic, 2012.

[33] R. Fisher, "The use of multiple measurements in taxonomic problems." Annals of eugenics, Vol. 7(2), pp. 179-188, 1936.

[34] R. Detrano et al., "International application of a new probability algorithm for the diagnosis of coronary artery disease." American Journal of Cardiology, Vol. 64, pp. 304-310, 1989.

[35] M. Charytanowicz et al., "A Complete Gradient Clustering Algorithm for Feature Analysis of X-ray Images." Information Technologies in Biomedicine, Springer: Berlin-Heidelberg, pp. 15-24, 2010.

[36] J. Hoeting, "ISEC2020: Learning Kaggle." Kaggle, https://kaggle.com/competitions/isec2020learnkaggle, 2020, Last accessed 27 March 2023.

[37] D. Kibler, D. Aha and M. Albert, "Instance-based prediction of real-valued attributes." Computational Intelligence, Vol. 5, pp. 51-57, 1989.

[38] C. Yeh, "Modeling slump flow of concrete using second-order regression and artificial neural networks." Cement and Concrete Composites, Vol. 29, pp. 474-480, 2007.

[39] J. Gerritsma, R. Onnink and A. Versluis, "Geometry, resistance and stability of the delft systematic yacht hull series." International Shipbuilding Progress, Vol. 28, pp. 276-297, 1981.

[40] J. Vanschoren, "OpenML tecator", https://www.openml.org/d/505, 2014, Last accessed 27 March 2023.

[41] W. Loh, "Improving the precision of classification trees." Annals of Applied Statistics, Vol. 3(4), pp. 1710-1737, 2009.

[42] T. Voigt, M. Kohlhase and O. Nelles, "Incremental DoE and Modeling Methodology with Gaussian Process Regression: An Industrially Applicable Approach to Incorporate Expert Knowledge." Mathematics, Vol. 9(19), 2021.