15TH IEA HEAT PUMP CONFERENCE
MAY 26–29, 2026 | HOFBURG VIENNA 2026
DECARBONISATION THROUGH INNOVATION

Machine Learning-Based Fault Detection in a R-290 Residential Heat Pump Unit

Daniele Scapin[a*], Matteo Galenda[a], Riccardo Pengo[a], Chiara Corazzol[a], Mirco Rampazzo[b]

[a]Carel Indstries S.p.A., Via dell'Industria, 11, Brugine, Padova, 35020 Italy
[b]DEI UNIPD, Via Gradenigo 6/b, Padova, 35131, Italy

Abstract

Anomaly and fault detection techniques are gaining increasing importance in the HVAC sector due to their ability to improve system reliability. While historically used in industrial applications, where costs were related to the plants, recent advances in digital integration have enabled their deployment even in smaller HVAC systems, such as residential heat pump units. Detecting early signs of inefficient operation can help predict faults, thereby reducing costs and user discomfort due to system failures, and preserving long-term performance. In this work, a comparative analysis is presented of various supervised fault detection algorithms applied to a typical R-290 Air-to-Water (A/W) heat pump application. A Matlab/Simscape-based virtual simulation environment of a R-290 heat pump unit is used to replicate both normal and faulty conditions. The simulation model allows for the generation of consistent and repeatable datasets, overcoming the limitations of time-consuming laboratory data collection. The algorithms under investigation include Support Vector Machines (SVM), Histogram-based Gradient Boosting (HistGB) and Neural Networks (NN). Results provide a performance overview of each method in simulated fault scenarios, offering insight into their suitability for real-world deployment and potential integration into field-ready HVAC monitoring systems.

Selection and/or peer-review under the responsibility of the organizers of the 15th IEA Heat Pump Conference 2026.

Keywords: Heat Pumps;Machine Learning; Modelling; Fault Detection

1. Introduction

The global heat-pump market is expanding rapidly, driven by electrification, stricter energy-efficiency standards, and policy incentives for decarbonization. As installations increase across residential, commercial, and industrial sectors, service demands and lifetime maintenance costs are expected to rise substantially with growing system complexity, and a limited pool of skilled technicians. Therefore, having an embedded indications for self-diagnosis would be a fundamental requirement for a mass distributed product.
These solutions reduce downtime and service visits, preserve user privacy, and enable predictive-maintenance strategies that lower operational expenditures and extend equipment life.

Fault detection and diagnosis (FDD) of HVAC system have become increasingly important, as early identification of abnormal operation can boost system reliability, cut energy consumption, and maintain occupant comfort [1]. Recent research has indicated that Artificial Intelligence (AI) and Machine Learning

*Corresponding author. Tel.: +39. 049.9716611
E-mail address: daniele.scapin@carel.com

(ML) methods, ranging from simple classifiers to advanced deep learning techniques, are among the most effective ways to improve anomaly detection and predictive maintenance in HVAC systems [2] [3].

Despite increasing interest, existing datasets and benchmarks are largely designed for large commercial facilities or central-plant equipment, and they underrepresent small residential units. Publicly available labeled datasets are valuable resources but do not represent the full range of fault modes relevant to residential A/W heat pumps [4]. Simulation-based approaches have therefore become more popular over time, as they allow researchers to generate reproducible and consistent datasets with explicitly labeled fault and nominal operating conditions at less cost and with less time spent on experimental campaigns. Among existing tools, Matlab/Simscape provides an integrated modelling environment that supports dynamic HVAC component simulation and controlled fault injection [5]. Specifically, over 1 million different datapoints have been generated from our model, containing healthy, evaporator fouling, undercharge and overcharge conditions.

To obtain well-validated simulated datasets reflecting real system behavior, the choice of refrigerant and model of the system must be properly motivated. For instance, recent research has demonstrated the ability to simulate R-290 heat pump systems and confirm their steady state and transient behavior with experimental data, showing that simulations can effectively capture relevant operating features important to FDD [6] [7]. Training FDD models within simulation environments before deployment on physical hardware offers multiple advantages: it enables safe, cost-effective data generation, allows systematic testing across a wide range of fault scenarios, and accelerates model development cycles. Once validated in simulation, models can be transferred to real systems, inheriting simulation-driven robustness while reducing the risks and costs associated with extensive real-world testing.

This renders simulation environments a top choice for anomaly detection and fault classification research in domestic use. Among the many ML algorithms that exist, SVM and NN remain popular for supervised FDD assignments. SVMs possess the power in high-dimensional space [8] and NNs can model nonlinear interactions [9]. Comparative studies have shown that these algorithms are competitive baselines in HVAC FDD and can test performance under a broad set of simulated fault conditions [10] [11].

The Histogram-based Gradient Boosting algorithm (Hist-GB in *scikit-learn*) is an efficient implementation of gradient boosting that significantly reduces training time for medium and large datasets. With features such as automatic handling of missing values, support for categorical variables, monotonic constraints, and fast histogram-based training, it is a strong choice for analysing both simulated and real HVAC datasets. [12].

Empirically, benchmarking and survey research reveal no one clear winner for table problems. Older methods like the SVM remain highly competitive on small-to-medium sized datasets because of their interpretability, low tuning cost and stability. NN can be trained to learn highly nonlinear, complex dynamics if sufficient data and regularization are available, while gradient-boosted trees (including their histogram-based relatives) perform at state of the art levels consistently using low training cost and out of the box strong behavior.

2. Simscape Model data generation

The Simscape model is extensively described in the article “Dynamic Modelling and Simulation of Faults in a R-290-based Air-to-Water Heat Pump System*”* [6]. To sum up briefly, it models a real domestic A/W heat pump of a nominal size of 7 kW. Each simulated component was calibrated to mimic its real-world counterpart, so the simulated components do not share identical geometric and technical parameters with the originals. When manufacturers did not provide specific values, we used Simscape’s defaults. Other parameters (particularly geometries), were measured or estimated by us, introducing a degree of human uncertainty that should be considered when interpreting results. With this validated model we can generate a range of datasets, including anomalous conditions. By tuning component parameters within Simscape, we simulate realistic fault scenarios. This study examines evaporator fouling, refrigerant undercharge, refrigerant overcharge, and the healthy system state.

The fouling effects are well described in the thesis done by Y. Alkurdi [13], where the most appreciable effects are a reduction of the air mass flow rate and a drop of the heat exchanger global thermic coefficient “K”. The most common fault occurring in the evaporator is the presence of dust and dirt covering it, causing an

Evaporator Fouling (EF). To replicate the fault in Simscape, the fan performance characteristic is altered so that it produces equivalent effects on the refrigerant and the air streams.
To reproduce the Undercharge (UC) and Overcharge (OC), the refrigerant mass is diminished and augmented by 0.2 kg with respect to the nominal charge.

2.1. **Dataset generation**

A total of 4 simulations has been executed: Healthy (H), EF, UC, OC. Every simulation contains about 1 million points, collected varying the following variables with a sinusoidal wave:

Table 1. Variables conditions

Variable	Unit	Offset	Amplitude	Frequency (rad/s)
External Temperature	°C	5	15	$2 \cdot 10^{-3}$
External Humidity	%	60	30	$1 \cdot 10^{-2}$
SuperHeat SetPoint	K	8	6	$2 \cdot 10^{-5}$
Water Outlet SetPoint	°C	32	3	$1 \cdot 10^{-3}$
Water Inlet	°C	28	3	$2 \cdot 10^{-4}$
Water Mass Flow rate	kg/s	0.4	0.1	$1 \cdot 10^{-5}$
Fan Speed	rad/s	90	50	$5 \cdot 10^{-4}$

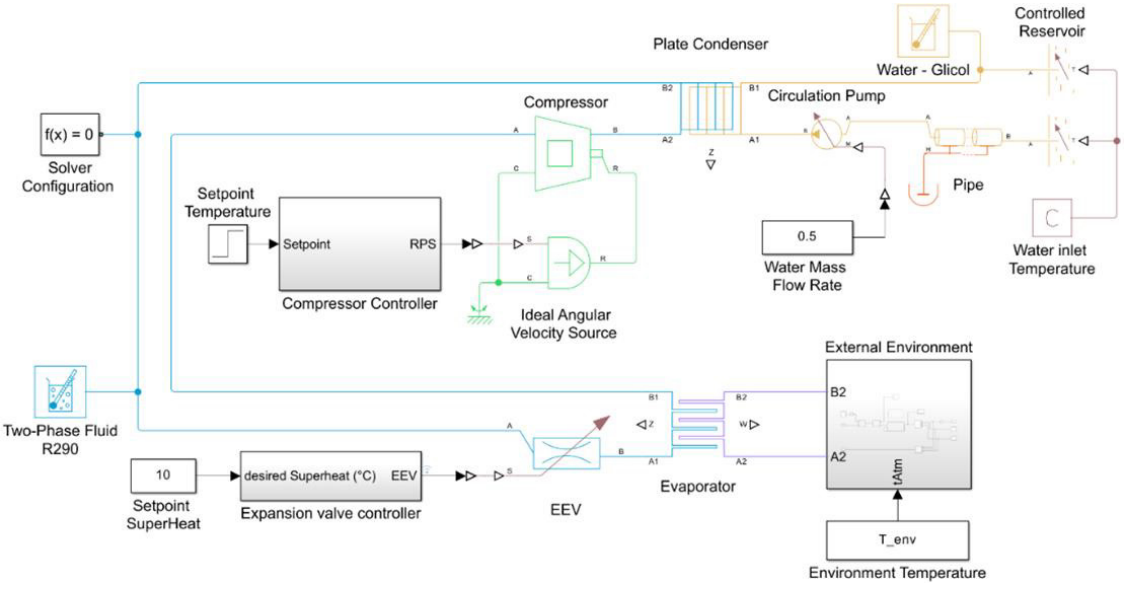

Figure 1. Heat Pump Simscape model

3. Dataset pre-processing

The preprocessing pipeline performs:

i. deterministic cleaning
ii. stratified train/validation/test split
iii. data augmentation by additive Gaussian noise
iv. feature scaling

The following paragraphs explain each operation justifying the choices for a large dataset ($\approx 4 \cdot 10^6$ samples).

3.1. Deterministic cleaning

The first block removes physically implausible or invalid records and reduces redundant precision:

- **Lower/upper bounds and plausibility checks.** Rows with non-positive or very low SuperHeat (SH) (< 2), low SubCooling (SC) (< 0.5), are discarded because those values are physically dangerous or impossible conditions for the real machine. The SH is also constrained to lie within ±1 unit of the SH Setpoint. Enforcing such bounds removes measurement errors and gross sensor faults before modeling, which is standard practice in engineering datasets to avoid contaminating the training distribution with invalid samples [14].
- **Rounding to one decimal place.** With the rounding, the algorithms learn to deal with a more realistic scenario (more like the real probes and sensors measurement precision). This type of preprocessing can enhance the performance of a little as our case.
- **Duplicate removal**. Removing exact duplicates is a simple and low-risk way to reduce dataset size and avoid over-representing repeated measurement [15]. The duplicate removal is performed after the rounding, thus removing more points than before the approximation.

After these operations, the dataset was reduced to a total of $3 \cdot 10^6$ points.

3.2. Train / validation / test split

Train/validation/test split is the standard practice of dividing the labeled dataset into three disjoint parts:

- **Train:** used to fit model parameters.
- **Validation:** used to make modeling decisions (hyperparameters, architecture, early stopping, feature selection).
- **Test:** a final holdout is used only once to estimate how the chosen model will perform on truly new data. The test set is held out until the very end to provide an unbiased estimate of how your chosen model will perform on truly new data.

Data splitting is a crucial step in training machine learning models, as it helps prevent overfitting, avoids data leakage during model development, and provides an unbiased estimate of performance through an untouched test set.
In this study, a conventional 60/20/20 split (training/validation/test) is applied. Given the large amount of available data, a smaller proportion can be allocated to training, while the remaining data are reserved for validation and testing to obtain a more reliable assessment of generalization performance.

3.3. Data augmentation by additive Gaussian noise

Adding zero-mean Gaussian noise to input features is employed as a straightforward data-augmentation and regularization technique that can improve generalization by forcing models to be robust to small measurement perturbations. As demonstrated in [16] training with additive input noise is formally equivalent, for certain learners, to Tikhonov (L2) regularization. This procedure is particularly appropriate when the modelled system is observed via sensors subject to noise.
The noise standard deviation is computed separately for each feature and scaled by 7%. This value was selected as a moderate perturbation level large enough to introduce meaningful variability and encourage robustness, yet small enough to avoid distorting the original feature distributions or altering the underlying physical relationships in the data. Empirically, perturbations in the range of 5–10% are commonly adopted in

multivariate sensor datasets, as they approximate typical measurement uncertainty while maintaining model stability. Using 7% therefore provides a balanced trade-off between regularization strength and data fidelity.

3.4. Feature scaling (Min–Max normalization)

Features with very different numeric ranges (e.g. pressures, temperatures and rps) distort distance calculations, gradients, and regularization and that makes training slower, less stable, or biased toward large-scale features. Scaling puts features on a common numeric order so algorithms behave correctly and efficiently.

Min–Max rescales each feature to a prespecified interval (default [0,1]). Linear models and gradient-based optimizers (e.g. NN) but also the SVM benefit from bounded feature ranges. Tree-based learners on the other hand, are invariant to monotonic rescaling. Nevertheless, applying a single preprocessing pipeline to all models simplifies the experimental setup and avoids mistakes when models are combined.

4. Algorithms Description

This section presents the three evaluated algorithms, with a summary of their advantages and disadvantages. To provide an initial indication of training complexity, the training time for each algorithm is reported. The reported durations correspond to the laptop used for experimentation and are therefore hardware-dependent: more powerful systems require less time, while lower-end systems require more. Consequently, the times should be interpreted as relative indicators of computational cost rather than absolute benchmarks.

4.1. Support Vector Machine (SVM)

SVM finds a boundary between classes that maximizes separation in feature space; with kernel functions it can separate non-linear patterns. [17]

Pros	Cons
• Effective in higher-dimensional spaces and with smaller datasets. • Often robust and provides good baselines without very large training sets.	• Less interpretable than trees. • Training and hyperparameter tuning can be expensive for very large datasets. • Does not naturally provide well-calibrated probability outputs (useful for risk scoring) without extra steps.

The SVM Serves as a classical ML baseline, particularly informative for smaller subsets of labeled scenarios or for tightly constrained feature sets. The training time for this algorithm is about 15s.

4.2. Histogram-based Gradient Boosting (HG-Boost)

An efficient form of gradient-boosted decision trees that train by grouping values into histogram bins; it builds many small trees sequentially where each new tree corrects previous errors. [18] [19]

Pros	Cons
• Strong accuracy with relatively low tuning. • Fast training on medium-to-large datasets; built-in support for missing values and categorical features. • Produces feature importances and partial dependence plots for interpretability.	• Less interpretable than a single decision tree, though more interpretable than deep networks. • Ensembles can be memory-heavy if extreme performance requires many trees.

Hist-GB is an excellent mid-ground: near state-of-the-art on tabular data while still being production friendly. Its training time is about 90s.

4.3. Feedforward Neural Networks (NN)

Networks of interconnected layers that learn complex non-linear relationships among inputs. They learn representations through many parameters (weights). [20]

Pros	Cons
• Highly flexible: capable of modelling complex, non-linear interactions between sensors and operating conditions. • Good choice when large amounts of labeled data are available.	• Require more data, computation time, and careful regularization to avoid overfitting. • Black box: harder to interpret and explain to technicians without additional tools. • Training and hyperparameter tuning are more involved; deployment on low-power controllers may be harder.

The NN can capture highly non-linear dynamics when abundant simulated data are available as this study case. [21] The model took about 1h to be trained.

Table 2. Algorithms sum up

Algorithm	Interpretability	Training cost	CPU and memory	Data needs
Support Vector Machine	High	Low	Low	Low
HistGB	Medium	Low	Low	Low
Neural Network	Low	High	High	High

5. Results

Since this is a classification problem, Confusion Matrix (CM) is adopted as a classic ML classification results indicator, with some statistic scores. A CM is a contingency table for classification problem, where each row corresponds to the actual class and each column corresponds to the predicted class. The element in row *i*, column *j* is the count of samples whose true class is *i* and that were predicted as *j*. In the diagonal positions *ii,* we have the correctly classified elements, while off it we have the false positives and false negatives (in the case of Fault detection in heat pump, they can be seen as false alarms and missed alarms). In our case, we have a total of 4 classes, thus we have a 4x4 CM. The description and use of CM and standard classification metrics follow common data-mining practice. [22]
Regarding the statistic scores (reported in the bar plot), the precision, recall, accuracy, balanced accuracy and F1-score are adopted.
In the bar plots are represented the averaged scores for each class since printing them for each class is a repetition of what is contained inside the CMs.

Figure 2 shows that, in the case studied, it is almost impossible to obtain a perfect separation among the 4 classes. This is caused either by the algorithm capabilities and the data themselves. More specifically, NN does best because it learns complex, high-order interactions.
The FDD likely depends on non-linear interactions and conditional patterns between sensors/variables (e.g., combinations of temperature, pressure, rps) and NN model those interactions directly. Simpler models cannot do it automatically and they need manually engineered interaction features, like enthalpies, powers, mass flow rates or other variables. NN has enough capacity to separate classes, giving very good results.
The model produces high recall on the harder classes (e.g., Healthy and Evaporator Fouling show high recall with NN), meaning they find decision boundaries that capture those class regions well.

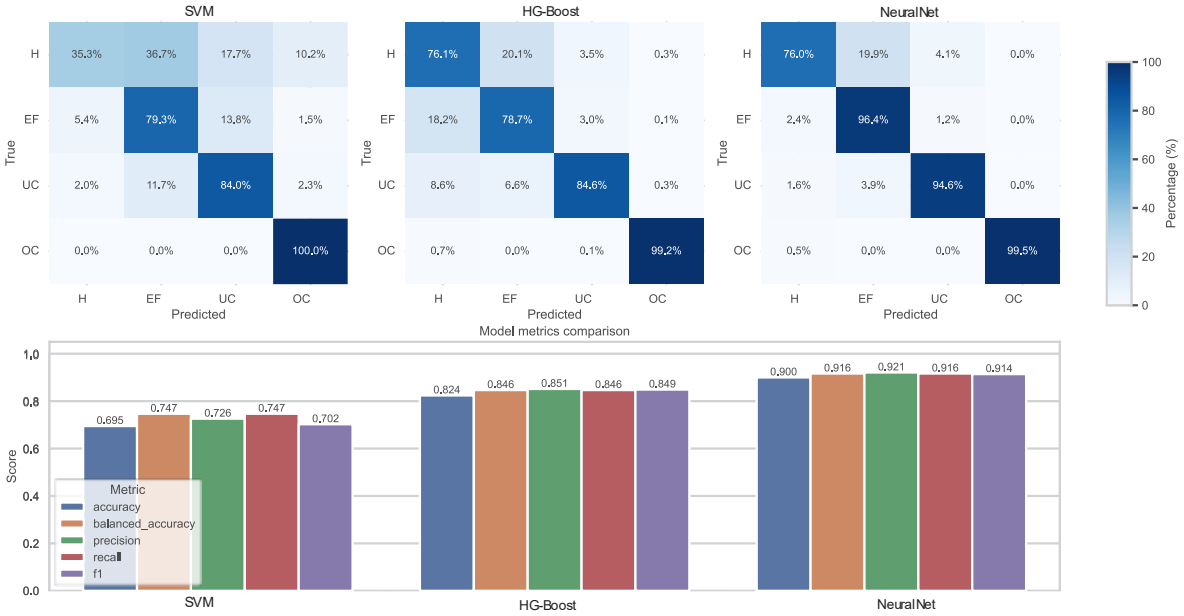

Figure 2. Confusion Matrices and scores

HG-Boost captures nonlinear relationships from numerical features with minimal preprocessing, which explains its strong performance on tabular data. However, histogram-based tree ensembles have lower representational capacity than models that learn dense feature representations (for example NN with learned embeddings) and therefore may underperform when many subtle cross-feature interactions exist or when learned embeddings provide a clear advantage. Despite this, HG-Boost often remains the preferred choice when robustness, speed, and limited preprocessing are priorities.

The SVM has the worst performance in our case (F1 score = 0.70). This can be caused by the linearity assumption. Indeed, basic SVM assumes linear separability (or separability after simple feature transforms) features. Looking at its respective confusion matrix, we have a lot of false positives and negatives, indicating that the true decision boundary is highly non-linear and the linear model simply cannot separate class 0 well.

Here are three reported examples of misclassified data with the NN model. The first is a wrong prediction of EF while the true value is H. The second reports an UC while the truth is EF. The last one classifies EF as a UC point.

Table 3. Data Examples

Discharge Pressure [barg]	Condensing Temperature [°C]	Discharge Temperature [°C]	Suction Temperature [°C]	Suction Pressure [barg]	Evaporation Temperature [°C]	Compressor speed [rps]	External Temperature [°C]
10.4	32.4	61.1	3.4	2.4	-10.5	76	4.3
10.4	32.3	61.8	-5.9	2	-14.5	73	-5.4
10.7	33.2	62.1	3.6	2.4	-10.2	51	6.4

6. Conclusions

The sim-to-real gap is mitigated by embedding the ML models within a data-validation and preprocessing pipeline that removes non-physical or faulty sensor measurements prior to inference. While simulation-based training cannot fully represent all real-world conditions, this approach improves robustness in practical deployments, with future work aiming to address this aspect through the inclusion of real operational data.
Overall, the three algorithms comparison indicates that model complexity and the ability to capture non-linear feature interactions are the primary determinants of performance on the heat-pump fault-classification task. The Neural Network achieved the highest performance (F1 = 0.91), confirming that deep networks are effective at modelling complex feature dependencies. HG-Boost also produced strong, competitive F1 score (F1 = 0.85), showing that both deep learners and boosted ensembles can recover the physical relationships among sensor variables. The linear Support Vector Machine classifier performed worst because of its linear separability assumption. Across all methods, refrigerant undercharge and overcharge were the easiest classes to detect; evaporator fouling was the most difficult and accounted for a large share of misclassifications. Although deep models deliver superior accuracy, their greater computational cost and deployment complexity remain important practical drawbacks. Consequently, selection for residential and industrial application must balance detection performance against interpretability, transparency, operational and computational constraints like HG-Boost.
To conclude, FDD in simulation environment is essential to design the right solution as a combination of hardware and software for the final application. This framework assures sustainability in the long term for both the manufacturer and the customer.

References

[1] I. Mateti, I. Štajduhar, I. Wolf and Sandi, "A review of data-driven approaches and hvac systems.," 2023.

[2] Y. Jian Bi, H. Wang, E. Yan, C. Wang, K. Yan, L. Jiang and Bin, "AI in HVAC fault detection and diagnosis: A systematic review," vol. 3, 2024.

[3] C. Zhelun, O. Zheng, W. Jin, P. Ojas, Y. Tao, L. Xing, L. Guanjing, M. Shohei, L. Seungjae, S. Chou, R. Chiosa, M. Piscitelli, A. Capozzoli, F. Hengel and A. Kührer, "A review of data-driven fault detection and diagnostics for building HVAC systems," vol. 339, 2023.

[4] J. Granderson, G. Lin, Y. Chen, A. Casillas, J. Wen, Z. Chen, P. Im, S. Huang and J. Ling, "A labeled dataset for building HVAC systems operating in faulted and fault-free states," 2023.

[5] MathWorks, "Simscape Faults Interface".

[6] D. Scapin, M. Rampazzo, R. Pengo, C. Corazzol and W. Muvegi, "Dynamic Modelling and Simulation of Faults in a R290-based Air-to-Water Heat Pump System," 2025.

[7] W. Yanjun, J. Lin, W. Jianhua, C. Wang, T. Zhao and Yuting, "Simulation and experimental study on dynamic characteristics of R290 split heat pump during start-up," vol. 236, 2024.

[8] C. Cortes and V. Vapnik, "Machine Learning," 1995.

[9] C. Bishop and N. Nasrabadi, "Pattern Recognition and Machine Learning," 2006.

[10] W. Tun, K. Wong and S. Ling, "Advancing Fault Detection in HVAC Systems: Unifying Gramian Angular Field and 2D Deep Convolutional Neural Networks for Enhanced Performance," 2023.

[11] A. Alghanmi, A. Yunusa-Kaltungo and R. Edwards, "A Comparative Study of Faults Detection Techniques on HVAC Systems".

[12] F. Pedregosa, G. G. A. Varoquaux, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher and M. Perrot, "Scikit-learn: Machine Learning in Python," 2011.

[13] Y. Alkurdi, "Anomaly Detection in Heat Pumps: Experimental Setup, Testing, and Data Analysis," 2023.

[14] R. Erhard and H. D. Hong, "Data Cleaning: Problems and Current Approaches".

[15] T. p. d. team, "pandas-dev/pandas: Pandas," 2020. [Online].

[16] C. Bishop, "Training with Noise is Equivalent to Tikhonov Regularization," vol. Neural Computation, 1995.

[17] C. Cortes and V. Vapnik, "Support-vector networks," vol. 20, 1995.

[18] L. Breiman, J. Friedman, R. Olshen and C. Stone, "Classification and Regression Trees (1st ed.)," 1984.

[19] J. R. Quinlan, Programs for Machine Learning}, 1993.

[20] Y. Gorishniy, I. Rubachev, V. Khrulkov and A. Babenko, "Revisiting Deep Learning Models for Tabular Data," vol. abs/2106.11959, 2021.

[21] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf and E. Yang, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," 2019.

[22] H. Jiawei, K. Micheline and P. Jian, "Data Mining Concepts and Techniques," 2012.