Performance Analysis of Machine Learning Models in Predictive Maintenance

1st Miguel Lopes
Industrial Engineering and Management
Faculdade de Engenharia da Universidade do Porto
Porto, Portugal
up202006670@up.pt

2nd Rodrigo Costa
Industrial Engineering and Management
Faculdade de Engenharia da Universidade do Porto
Porto, Portugal
up202006773@up.pt

***Abstract*—This paper presents an analysis of Machine Learning (ML) techniques for predictive maintenance, utilizing nine algorithms with the goal of anticipating equipment breakdowns to reduce both corrective and preventive maintenance expenditures. The performance of these models was evaluated based on their ability to accurately predict failures and minimize false alarms. In the end, the outcomes for each model are detailed, along with a comparative analysis of the top-performing models for this specific dataset.**

***Index Terms*—Predictive Maintenance, Machine Learning, Classification, Models**

I. Introduction & Problem Definition

During the fourth industrial revolution, predictive maintenance has become crucial in sustainable manufacturing. It involves using digitalized machine maintenance to anticipate when machinery needs servicing, helping streamline decision-making processes in manufacturing by utilizing vast datasets. The main challenge lies in accurately predicting when maintenance will be needed. [1]

The integration of predictive maintenance capabilities serves to ameliorate machine downtime, expenditure, oversight, and production quality. Typically, this approach embodies a meticulously outlined workflow commencing with project comprehension and data acquisition, culminating in the decision-making phase. [2]

In a nutshell, the main goal of this project is to perform a comparative analysis of the performance of several machine learning algorithms. This will be done based on a synthetic dataset that aims to echo real predictive maintenance information.

II. Background Theory

A. Machine Learning

Learning, involves acquiring or adjusting behaviors, values, knowledge, skills, or preferences. But, unlike humans who learn from experiences, machines rely on data. At its core, machine learning (ML) is a subset of artificial intelligence where computers autonomously learn and adapt. ML aims to enhance accuracy by modifying actions based on outcomes. [3]

Tom Mitchell stated: A computer program demonstrates learning from experience E in relation to a task T and performance measure P if its performance on task T, as gauged by performance measure P, enhances over time with accumulated experience E. [4]

B. Feature Engineering

In machine learning, features act as numeric representations of different aspects in raw data, serving as a vital bridge between data and models. Feature engineering encompasses the extraction of these features from raw data and their conversion into formats suitable for machine learning models. This pivotal step in the pipeline can significantly streamline the modeling process, leading to better outcomes. Practitioners widely recognize that a substantial amount of effort is dedicated to feature engineering and data cleansing tasks. [5]

C. Supervised Algorithms

Supervised learning is a subset of ML and AI, relies on labeled datasets to train algorithms for accurate data classification or outcome prediction. Through the iterative process of feeding input data into the model and adjusting weights during cross-validation, the model is fine-tuned until it fits appropriately.

Opposing to unsupervised learning where algorithms are trained on unlabeled data without explicit guidance. Instead of predicting specific outcomes, unsupervised learning focuses on finding patterns, structures, or relationships within the data.

D. Classification

In supervised learning, particularly in classification problems, algorithms aim to mimic a function that assigns vectors to different classes. They learn this by studying input-output examples of the function. Inductive machine learning involves extracting rules from instances, often from a training set, or more broadly, creating a classifier that can generalize to new instances. [6]

E. Model Selection and Evaluation

Measuring the performance of a machine learning model conventionally involves training on one dataset, testing on another, and assessing accuracy. However, this simplistic approach overlooks nuances and prompts a deeper inquiry into the importance of performance estimates. These estimates provide insights into a model's ability to generalize to new data, crucial for real-world applications.

Given the iterative nature of machine learning, experimentation is inevitable, often involving the adjustment of internal parameters known as hyperparameters. This experimentation yields various models with differing performance levels. Hence, devising methods to estimate and compare their performance is crucial for selecting the most effective model for future predictions. [7]

Selecting the right model for machine learning involves considering various factors to ensure effectiveness and suitability for the task. These include accuracy, interpretability, computational efficiency, scalability, robustness, flexibility, resource constraints, domain knowledge, and ethical considerations.

1) Hyperparameter tuning: Hyperparameter tuning is the process of identifying the most effective hyperparameter configuration for a machine learning model, crucial for optimizing performance on a given task. Various techniques exist for tuning hyperparameters, with common methods including grid search, random search, and Bayesian optimization.

F. Models Explanation

During the development of this project, several classification models were compared. These models vary significantly in their underlying algorithms, assumptions, and capabilities, offering diverse options for addressing classification tasks in machine learning. The bases of each one of them is explained below.

1) One Rule: The One Rule (OneR) algorithm creates rules for each attribute in a dataset and selects the one with the lowest error rate as the main rule. Each attribute's most frequent class is used to create a rule, based on which class occurs most often for that attribute value. This algorithm is a basic benchmark for comparing predictive power among different algorithms because of its simplicity and focus on a single attribute. [8]

2) Multinomial Logistic Regression: It is a multivariable approach allows modeling of diverse variables, even on different measurement scales, making the model flexible. It focuses on estimating coefficients' significance. Design variables, or dummy variables, encode categorical variables for logistic regression analysis, with each representing a level of the nominal scale variable to capture its effect on the outcome. [9]

3) K Nearest Neighbors: The standard kNN method predicts the label of a test data point based on the majority rule among its k nearest neighbors in the feature space. Challenges include determining data point similarity and selecting the optimal k value. Approaches like Euclidean, Mahalanobis, and Minkowsky distances, and their variants, aim to refine similarity measurement to improve kNN classification accuracy and effectiveness. [10]

4) Naive Bayes: Bayesian statistics centers on two key probability operations: conditioning (Bayes Theorem) and marginalization. By integrating these operations with empirical observations, it derives valid inferences iteratively. Following Bayesian principles involves fully specifying a model as a joint distribution encompassing all relevant variables, even those lacking definite prior knowledge, by assigning prior distributions to them. [11]

Despite its simplicity and the "naive" assumption of feature independence, Naive Bayes often performs surprisingly well and can serve as a baseline model for more complex classification algorithms.

5) Decision Trees: A decision tree serves as a classifier by dividing the instance space recursively. It's composed of nodes forming a rooted tree, with the root node having no incoming edges and others having one incoming edge each. Internal nodes, or test nodes, split the space based on attribute values, usually one attribute per test. Conditions may involve ranges for numeric attributes. Leaf nodes represent classes or probabilities of target values. Instances are classified by traversing the tree from root to leaf based on test outcomes. Analysts use decision trees to predict responses and understand customer behavior, labeling nodes by tested attributes and branches by corresponding attribute values. [12]

6) Random Forest: Random forest is an ensemble classifier composed of independent decision trees, utilizing randomization. Each tree is built using a random subset of features and samples from the training data to ensure diversity and reduce overfitting. The construction involves training through bootstrap sampling and randomly selecting feature subsets for splitting, enhancing tree diversity. The trees are then combined to form the random forest model, used for classifying unknown instances. By aggregating predictions from multiple trees, random forest offers robust and accurate classification. [13]

7) Neural Networks: In a conventional neural network (NN), interconnected neurons generate real-valued activations. Input neurons are activated by stimuli sensed by sensors, while weighted connections from previous activations stimulate other neurons, some of which may influence the environment. Learning involves adjusting weights to achieve desired behaviors, like steering a car. Problem nature and neuron interconnections dictate the network's computational stages, altering activations nonlinearly through extended chains of stages. [14]

8) Support Vector Machines: The Support Vector Machine (SVM) stands as a cutting-edge method for classification endeavors. To learn, it engages in identifying a collection of parameters through the resolution of a Convex Constrained Quadratic Programming (CCQP) problem, a task for which numerous efficient methodologies have been introduced. [15]

9) Extreme Gradient Boosting: XGB, short for Extreme Gradient Boosting, represents a variant of gradient boosting, falling within the category of supervised learning algorithms. Its primary aim is to enhance both the efficacy and efficiency of machine learning models. By harnessing the power of boosting, XGB amalgamates numerous tree models, each with relatively lower classification accuracy, to produce a single high-accuracy model. [16] XGBoost constructs a robust predictive model by combining the forecasts of multiple weak learners, typically decision trees. Employing a boosting approach, it iteratively improves the ensemble model's accuracy by ensuring that each weak learner rectifies the errors made by its predecessors. It also functions well even with incomplete

datasets because of its strong mechanism for handling missing data during training.

III. Methodology & Data

In order to develop a high-quality project, it was decided to follow the CRISP-DM approach. It stands for Cross-Industry Standard Process for Data Mining, and serves as a well-established industry framework for steering data mining endeavors. According to IBM, it has six phases: Business Understanding, Data Understanding, Data Preparation, Modelling, Evaluation, and Deployment. The life cycle of CRISP-DM is illustrated in Figure 1 of the Appendices. [17]

A. Business Understanding

Corrective Maintenance (CM) strategies have evolved due to the costs associated with unexpected production halts. Preventive Maintenance (PM) emerged as a proactive approach, involving periodic examinations and component replacements before critical failure. However, PM's scheduled replacements may lead to premature or delayed actions, affecting maintenance costs and potentially causing severe consequences.

The emergence of Industry 4.0 technologies and the Internet of Things (IoT) has led to the adoption of condition-based maintenance (CBM). This approach automates traditional manual inspections using sensors and devices to measure and monitor various physical parameters of industrial equipment. Interventions are triggered based on sensor readings, with actions taken when values surpass predefined thresholds. [18]

B. Data Understanding

Based on the analysis of the dataset consisting of 10000 observations, each containing 10 distinct attributes (UID, Product ID, Type, Air Temperature, Process Temperature, Rotational Speed, Torque, Tool Wear, Target, and Failure Type), we can draw several relevant conclusions.

Firstly, it's evident that the dataset exhibits a significant imbalance, with only a small percentage of observations corresponding to some type of failure (3.39%)-Figure 2. Additionally, we identified that the Rotational Seep and Torque variables initially contained outliers (Figures 6 and 4), which my affect the accuracy of our analyses.

It was observed that failures tend to occur, on average, under significantly higher air temperature conditions (Figure- 5), when the equipment is subject to higher levels of Tool Wear, or when machines have high Torque values. This indicates the importance of these factors in failure prediction. Furthermore, failures related to heat dissipation are associated with higher air temperature conditions - figure 3.

Further analysis revealed a negative correlation between Torque and Rotational Speed, while a positive correlation was observed between Air Temperature and Process Temperature (Figure 7). Further information is stated in section III-C3

C. Data Preparation

In this topic, all the steps regarding the transformation of the dataset for the application of the machine learning algorithms will be mentioned and explained.

1) Data Selection: Since the working dataset didn't contain many features, all of them were selected for the analysis, except for the *UID* and the *productID*, since they correspond to individual values for observation and product, respectively, and won't add relevant information to the models.

2) Data Cleaning: Regarding **missing values**, a preliminary treatment was done, transforming any possible "?", "-", or "", into *NA*'s. Then, by counting how many *NA*'s existed, it was possible to concluded that no missing values existed.

An analysis revealed inconsistencies: 18 observations with *FailureType* "Random Failure" had *Target* set to 0, contrary to the dataset's pattern; these were corrected to 1. Additionally, nine observations with *Target* as 1 and *FailureType* "No Failure" lacked further explanation and were consequently removed from the dataset.

Looking for **outliers** is a crucial step before developing machine learning algorithms. These consist of data points that deviate significantly from the rest of the dataset, and can distort results and compromise the integrity of the models. Several approaches were used:

- **Z-Score method**: This entails a statistical assessment of a score's correlation with the mean within a set of scores. A Z-score of zero indicates equivalence to the average, while negative or positive values signify deviations below or above the average by a certain number of standard deviations [19]. The Z-Score of each observation is calculated as follows:

$$z_i = \frac{x_i - \overline{x}}{s} \tag{1}$$

  Specifically, outliers are detected by calculating the Z-score, with a threshold of 3 being commonly used. Any data point with an absolute Z-score exceeding 3 is flagged as an outlier.
- **Interquartile Range**: The interquartile range (IQR) is computed as the difference between the third quartile (Q3) and the first quartile (Q1):

$$IQR = Q3 - Q1 \tag{2}$$

  This technique divides the dataset into quartiles, resulting in four equal parts. [20] Subsequently, the lower and upper boundaries are established as follows:

$$LowerBoundary = Q1 - (1.5 * IQR) \tag{3}$$

$$UpperBoundary = Q3 - (1.5 * IQR) \tag{4}$$

  Values under the lower boundary, and over the upper boundary are considered as outliers.
- **Local Outlier Factor (LOF)**: Is based on computing the outlying degree for each data point by assessing the average ratio between its local reachability density and the minimum number of nearest neighbors. Each data point is then assigned a LOF score according to specific equations outlined in the reference. [21] One notable advantage of LOF is its capability to discern local density and pinpoint local outliers.

3) Data Construction: In order to avoid the creation of many dummy variables, **encoding** was used to transform the categorical variable *Type* into a numerical one so that more models could be fed with this feature. The procedure was the following: $L \rightarrow 0$, $M \rightarrow 0.5$, and $H \rightarrow 1$.

As mentioned before, due to the high correlation between the temperature variables, and between *Torque* and *RotSpeed*, as can be seen in Figure 7, two more variables were **created**:

$$TempDif = ProcessTemperature - AirTemperature \tag{5}$$

$$Power = Torque * RotSpeed \tag{6}$$

Subsequently, another dataset was created applying the Z-Score **normalization** method, since it's been proven to increase models accuracy by simplifying the data and putting into identical ranges. [22]

4) Data Formatting: Even though there were eight variables available for analysis and training, some of them might diminish the quality of the models. **Feature selection** offers an efficient solution to address this issue by eliminating irrelevant and redundant data. This process can lead to reduced computation time, enhanced learning accuracy, and a clearer comprehension of the learning model or dataset. [23] The applied methods were the following:

- MRMR (Minimum Redundancy Maximum Relevance): The MRMR feature selection algorithm chooses features by balancing their relevance to the target variable with their redundancy to each other. It calculates relevance and redundancy scores for each feature, selecting those with high relevance and low redundancy.
- Stepwise Backwards Feature Selection: works by iteratively removing features from the model until a stopping criterion is met. It starts with a model that includes all features and evaluates the impact of removing each feature individually. The feature whose removal results in the smallest decrease in model performance is eliminated in each iteration. This process continues until the model's performance no longer improves or until a predefined number of features is reached.

Then, the data was **split** 70% into training data, and 30% into testing data to evaluate the performance of different models, since this ratio is one of the most common to use, and it has been proven to be the one that leads to best results in classification problems [24]. In order to do hyperparameter tuning, a 10-fold cross validation technique was implemented.

Since the working dataset was highly unbalanced (3.39% of minority class), several **balancing** were tested and carried out.

Both Undersampling, SMOTE and Oversampling techniques were conducted to delete observations/create synthetic observations and decrease/increase the number of failures in the dataset. Only like this, it is possible to feed the model with enough information to predict possible events of the failure types with fewer observations registered in the original dataset.

D. Modelling

In the development of this project, nine different machine learning algorithms mentioned in Section II-F were compared. Firstly, a lot of ONE-R models were tested in order to evaluate different oversampling techniques (oversampling and SMOTE), and the prediction of different target variables (Target and FailType). Then, when performing the Naive Bayes model, undersampling vs. oversampling vs. unbalanced, and normalized variables vs. not-normalized variables were explored. Here it was stipulated that from this moment on, only normalized variables will be taken into account, both on oversampled and unbalanced datasets.

So, for all the other models, the feature selection scenarios were compared, taking into account the results from tables III and IV, and the hyperparameters were found for each of the scenarios using a 10-fold cross validation process. The hyperparameters grid values can be found in table I of the appendices. The only exceptions to this were the XGBoost and the Neural Networks, in order to save computational time.

E. Evaluation

1) Accuracy: Accuracy represents the proportional of correct predictions made by a model over the total number of predictions. In the context of the project at hand, the reliance solely on accuracy as a performance metric may not be advisable. This caution arises from the inherent imbalance within the dataset, where one class significantly outweighs the others.

2) Recall: Recall is calculated as the ratio of true positives (instances correctly predicted as positive) to the sum of true positives and false negatives (instances incorrectly predicted as negative but are actually positive.

3) AUC Score: This metric represents the probability that a model ranks a Positive higher than a Negative, characterizing the trade-off between True positives and false alarms.

4) Precision: Precision is calculated as the proportion of true positives among all instances predicted to be positive. In this way, precision measures the quality of positive predictions, providing an insight into how reliable the positive classifications made by the model are.

5) Weighted F1 Score: The F1 Score is calculated as a harmonic mean of the precision and recall scores for each type of failure (excluding "no failure"), taking into account their support in the dataset to assign different weights. This metric evaluates the models' ability to predict the correct type of failure and also takes into account the trade-off between this and false alarms.

IV. Results & Discussion

A. Feature Selection

The results, from both MRMR (Table III) and Stepwise Backwards Feature Selection (Table IV), were broken down into scenarios of variables for the models to take into account. These scenarios are presented in tables II of the appendices.

B. Outlier Detection

After all techniques, some numerical outliers could be identified. But when performing a graphical analysis, it was figured out that these don't echo "unreal" values. Let's take Figure 6 as an example. If it was decided to remove the observations in red, information related to failures would be excluded, since a lot of the failures happen in extreme values of the variables.

After a global analysis, it was decided that there were no real outliers, since no additional information about wrong measurements was available.

C. Models Evaluation

1) One-R: The One-R model was used as the first approach to the work. For this reason, we only looked at the accuracy values in this first model to serve as a benchmark.The results are shown in Table V.

2) k-Nearest Neighbor: As seen in table VI lower values of neighbors produces better results.

3) Decision Trees: From table VII, we can conclude that the best quality criterion is gini, the minimum samples required to split an internal node is smaller on the unbalanced datasets, and higher on balanced ones. The tree which lead to the best performance is illustrated in figure 8.

4) Multiple Logistic Regression: In the MLR model, the unbalanced approach resulted in the best weighted F1 metric, using the feature scenario 1 from table II.

5) Naive Bayes: Once again, lower hyperparameter values result in better results, as shown in the table VIII. Scenarios 1 and 2 stood out in this model.

6) Random Forest: In the unbalanced dataset, the number of features per node was higher (3) than in the balanced (1). The number of trees was stood in the lowest grid value for both of the datasets, which might indicate that a new refinement is needed with lower values in this parameter.

7) Support Vector Machine: The best Gamma value for unbalanced data was 0.01, while in balanced it was 0.1. As far as cost is concerned, in both cases the model performs better with higher values.

8) XGBoost: In the development of this model, the hyperparameters were defined for the first feature scenario only, and generalized for all the others. On the unbalanced dataset, higher ETA values, and smaller depth, gamma, and child weight were preferred. On the balanced dataset (which was the best model developed), smaller ETA and child weight values were selected, as well as higher gamma and depth. The results are shown in table XI

9) Neural Networks: Once again, this model was run for the first feature scenario only, in order to save computational time. This doesn't guarantee that an optimal model was found. In addition, only the hidden layers' hyperparameter was tested, and it was found to conduct a better performance when established in 3, as illustrated in table XII.

It is worth mentioning that for the majority of these models, there should be some refinement on these values before implementing the models. In some features, the values chosen for the grid are very spaced-out, not allowing to conclude if they are actually a global-optimum.

The performance results of each of the models in their best hyperparameter-scenario combination can be found in table XIII of the appendices.

V. Conclusions

Random Forest (Balanced) attains the highest accuracy, indicating its proficiency in correctly classifying instances across multiple failure types. This suggests that Random Forest effectively balances between the correct identification of all types.

Support Vector Machines (Balanced) demonstrates the highest AUC, implying its effectiveness in distinguishing between different classes. This suggests that SVM provide a good balance between true positive rate and false positive rate across the multiple failure types.

XGBoost (Balanced) demonstrates superior performance across multiple metrics -Figure 9 . It achieves the highest mean recall, indicating its ability to capture a high proportion of positive instances, crucial for correctly identifying each type. Additionally, it obtains the highest mean precision, implying confidence in its predictions while minimizing false positives. These also lead to the highest weighted F1 score, reflecting a balanced performance between precision and recall across all classes, making it the most robust and reliable in identifying failure types.

REFERENCES

[1] M. Achouch, M. Dimitrova, K. Ziane, S. Sattarpanah Karganroudi, R. Dhouib, H. Ibrahim, and M. Adda, "On Predictive Maintenance in Industry 4.0: Overview, Models, and Challenges," Applied Sciences, vol. 12, no. 16, p. 8081, 2022.

[2] T. Zonta, C. A. da Costa, R. d. R. Righi, M. J. de Lima, E. S. d. Trindade, and G. P. Li, "Predictive maintenance in the Industry 4.0: A systematic literature review," Computers & Industrial Engineering, vol. 150, p. 106889, 2020.

[3] J. Alzubi, A. Nayyar, and A. Kumar, "Machine Learning from Theory to Algorithms: An Overview," in Journal of Physics: Conference Series, vol. 1142, Dec. 2018, p. 012012. DOI: 10.1088/1742-6596/1142/1/012012.

[4] T. M. Mitchell, Machine learning, vol. 1, no. 9. McGraw-hill New York, 1997.

[5] A. Zheng and A. Casari, "Feature engineering for machine learning: principles and techniques for data scientists," O'Reilly Media, Inc., 2018.

[6] F. Y. Osisanwo et al., "Supervised machine learning algorithms: classification and comparison," International Journal of Computer Trends and Technology (IJCTT), vol. 48, no. 3, pp. 128-138, 2017.

[7] S. Raschka, "Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning," 2020.

[8] T. Soman and P. O. Bobbie, "Classification of arrhythmia using machine learning techniques," WSEAS Transactions on computers, vol. 4, no. 6, pp. 548-552, 2005.

[9] D. W. Hosmer Jr., S. Lemeshow, and R. X. Sturdivant, "Chapter 2: The Multiple Logistic Regression Model," 1st ed., 22 March 2013.

[10] S. Zhang et al., "Learning k for knn classification," ACM Transactions on Intelligent Systems and Technology (TIST), vol. 8, no. 3, pp. 2, 2017.

[11] M. Seeger, "Bayesian modelling in machine learning: A tutorial review," 2006.

[12] L. Rokach and O. Maimon, "Decision Trees," in Data Mining and Knowledge Discovery Handbook, O. Maimon and L. Rokach, Eds. Boston, MA: Springer US, 2005, pp. 165–192.

[13] Q. Ren, H. Cheng, and H. Han, "Research on machine learning framework based on random forest algorithm," AIP Conference Proceedings, vol. 1820, no. 1, p. 080020, 2017.

[14] J. Schmidhuber, "Deep learning in neural networks: An overview," Neural Networks, vol. 61, pp. 85-117, 2015.

[15] D. Anguita, A. Ghio, N. Greco, L. Oneto, and S. Ridella, "Model selection for support vector machines: Advantages and disadvantages of the Machine Learning Theory," in The 2010 International Joint Conference on Neural Networks (IJCNN), Barcelona, Spain, 2010.

[16] J. Mateo, J.M. Rius-Peris, A.I. Maraña-Pérez, A. Valiente-Armero, A.M. Torres, "Extreme gradient boosting machine learning method for predicting medical treatment in patients with acute bronchiolitis," Biocybernetics and Biomedical Engineering, vol. 41, no. 2, pp. 792-801, 2021.

[17] IBM, "CRISP-DM Help Overview". IBM. https://www.ibm.com/docs/en/spss-modeler/saas?topic=dm-crisp-help-overview (05/05/2024)

[18] P. Nunes, J. Santos, E. Rocha, "Challenges in predictive maintenance – A review," CIRP Journal of Manufacturing Science and Technology, vol. 40, pp. 53-67, 2023.

[19] B. C. Nwodo, D. M. Abdulmalik, O. A. Abisoye, and A. S. Bashir, "Outlier Detection in Multivariate Time Series Data using a Fusion of K-Medoid, Standardized Euclidean Distance and Z-Score," in International Conference on Information Technology in Education and Development, Academia in Information Technology Profession (AITP 2020), 2020.

[20] H. P. Vinutha, B. Poornima, and B. M. Sagar, "Detection of Outliers Using Interquartile Range Technique from Intrusion Dataset," in Information and Decision Sciences, S. C. Satapathy, J. M. R. S. Tavares, V. Bhateja, and J. R. Mohanty, Eds. Singapore: Springer Singapore, 2018, pp. 511-518.

[21] O. Alghushairy, R. Alsini, T. Soule, and X. Ma, "A Review of Local Outlier Factor Algorithms for Outlier Detection in Big Data Streams," Big Data and Cognitive Computing, vol. 5, no. 1, p. 1, 2021.

[22] M. A. Imron and B. Prasetyo, "Improving algorithm accuracy k-nearest neighbor using z-score normalization and particle swarm optimization to predict customer churn," Journal of Soft Computing Exploration, vol. 1, no. 1, pp. 56-62, 2020.

[23] J. Cai, J. Luo, S. Wang, and S. Yang, "Feature selection in machine learning: A new perspective," Neurocomputing, vol. 300, pp. 70-79, 2018.

[24] I. Muraina, "Ideal dataset splitting ratios in machine learning algorithms: general concerns for data scientists and data analysts," in 7th International Mardin Artuklu Scientific Research Conference, pp. 496-504, 2022.

APPENDICES

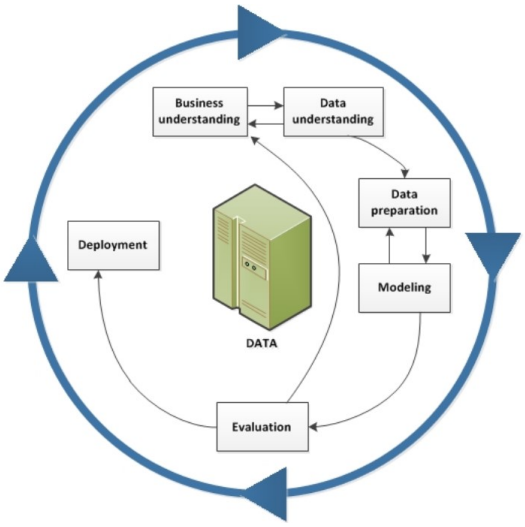

Fig. 1. CRISP-DM Life Cycle - IBM

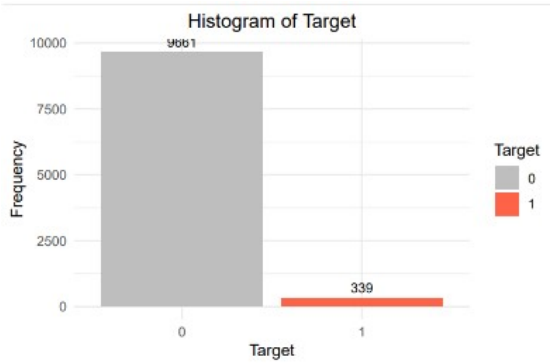

Fig. 2. Bar Plot ”No Failure” vs ”Failure”

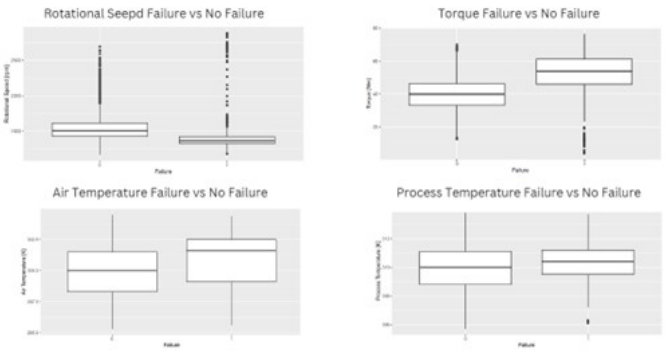

Fig. 3. Box Plots

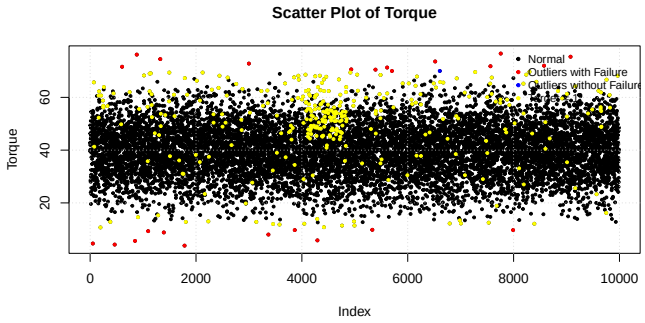

Fig. 4. Scatter Plot for the Analysis of Torque Outliers

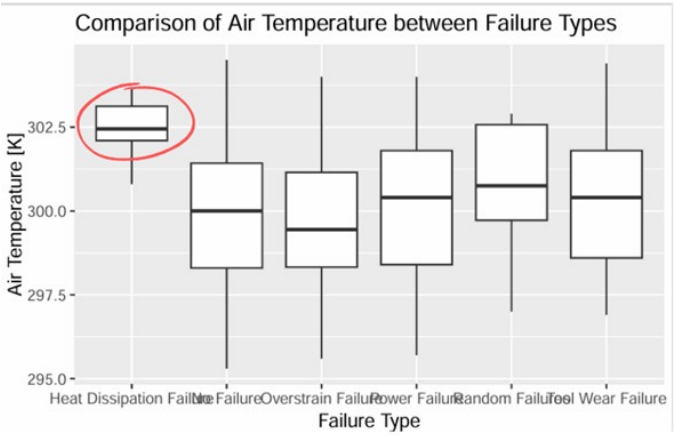

Fig. 5. Air Temperature by Failure Type

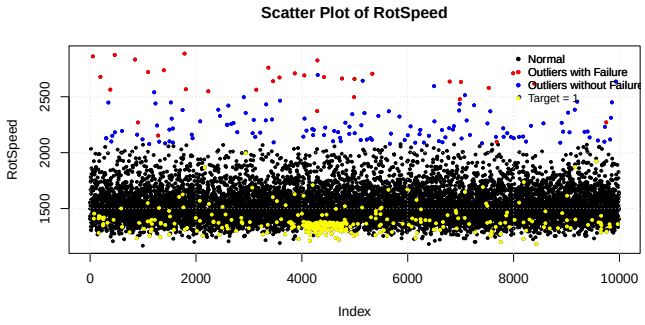

Fig. 6. Scatter Plot for the Analysis of RotSpeed Outliers

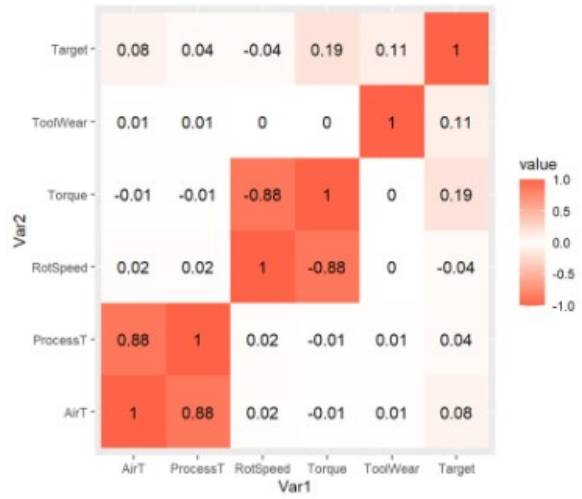

Fig. 7. Correlation Heatmap

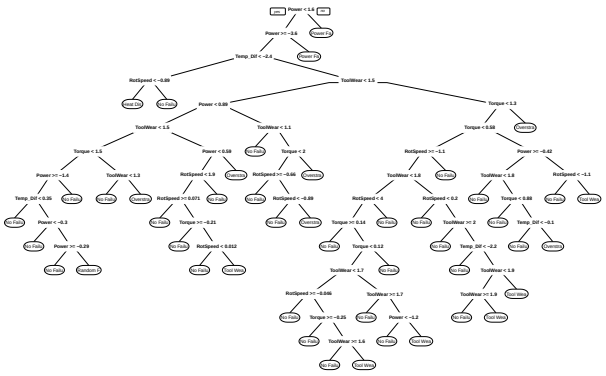

Fig. 8. Design of the tree with the best performance

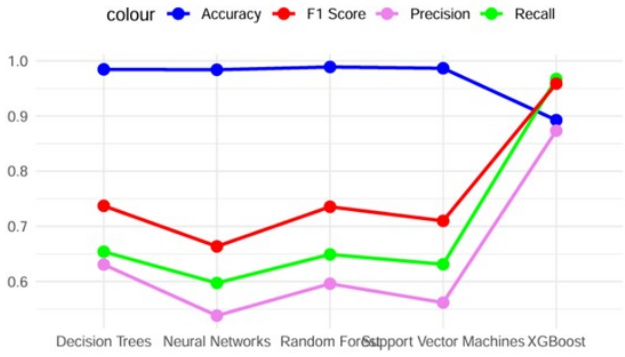

Fig. 9. Comparison betwen 5 models

TABLE I
HYPERPARAMETERS GRID VALUES

Model	Hyperparameters Grid
NB	**threshold**: 0, 5, 10, 15, 20 **gamma**: 0.1, 0.2, 0.3, 0.4, 0.5
KNN	**number of neighbors**: 1 to 40
DT	**min. number of observations per leaf**: 2 to 20 **criterion**: information, gini **complexity**: 0.01, 0
RF	**number of trees**: 250, 500, 750, 1000 **number of features**: 1, 3, 5, 7
SVM	**gamma**: 1, 0.1, 0.01, 0.001, 0.0001 **cost**: 0.01, 0.1, 1, 10, 100, 1000
XGB	**depth**: 4, 8 **number of rounds**: 10000 **minimum child weight**: 1, 5 **eta**: 0.001, 0.05 **gamma**: 0, 5
NN	**hidden layers**: 1, 3 **learning rate**: 0.1 **stepmax**: 10 000 000

TABLE II
SCENARIOS FROM FEATURE SELECTION

Scenario	Variables
1	AirT, ProcessT, RotSpeed, Torque, Type, Power, Temp_Diff, Toolwear, FailType
2	RotSpeed, Torque, Type, Power, Temp_Diff, Toolwear, FailType
3	AirT, ProcessT, Type, Power, Temp_Diff, Toolwear, FailType
4	AirT, ProcessT, RotSpeed, Torque, Type, Temp_Diff, Toolwear, FailType
5	AirT, ProcessT, RotSpeed, Torque, Type, Power, Toolwear, FailType
6	Type, Power, Temp_Diff, Toolwear, FailType
7	RotSpeed, ToolWear, Temp_Diff, Power, Torque, FailType
8	Type, RotSpeed, ToolWear, Temp_Diff, FailType
9	Type_Num, Torque, ToolWear, Temp_Diff, FailType

TABLE III
MRMR FOR TARGET VARIABLE AND DIFFERENT FAILURE TYPES

Target	Selction	Torque	ToolWear	Temp_Dif	is_type_L
	Score	0,00404	0,0087	0,002	0,00039
is_failType_Heat	Selction	Temp_Dif	Torque	is_type_L	ToolWear
	Score	0,02495	0,07426	-0,00021	-0,00535
is_failType_Overstrain	Selction	ToolWear	Torque	is_type_L	AirT
	Score	0,01473	0,01001	0,00204	-0,00456
is_failType_Power	Selction	Power	is_type_M	Temp_Dif	ToolWear
	Score	0,03957	-0,00034	-0,00146	-0,00548
is_failType_ToolWear	Selction	ToolWear	is_type_L	Torque	AirT
	Score	0,01159	-0,00045	-0,00147	-0,00466
is_failType_Random	Selction	Power	is_type_M	Temp_Dif	ToolWear
	Score	0,00063	-0,00018	-0,00119	-0,00548

TABLE IV
STEPWISE BACKWARDS FEATURE SELECTION RESULTS

FEATURE	COEFFICIENTS	PVALUES
Intercept	-8,319574869	0
AirT	1,503516706	0
ProcessT	-1,022702906	0
RotSpeed	2,298776517	0
Torque	6,079538721	0
Toolwear	0,8854987358	0
Power	-3,106176433	0
Type_L	0,4339605688	0,001888673182

TABLE V
RESULTS FOR ONE-R

Model	Normalized	Predict	ACCURACY
One-R	Yes	Random	0,564010
One-R	No	Random	0,544298
One-R	Yes	Toolwear	0,946347
One-R	No	Toolwear	0,932556
One-R	Yes	Power	0,911254
One-R	No	Power	0,911405
One-R	Yes	Overstrain	0,778982
One-R	No	Overstrain	0,780897
One-R	Yes	Heat	0,812426
One-R	No	Heat	0,937095
One-R	Yes	No failure	0,937095
One-R	No	No failure	0,777494

TABLE VI
HYPERPARAMETERS FOR K-NN

MODEL	DATA	SCENARIO	NORMALIZED	K
K-NN	Balanced	2	YES	1
K-NN	Unbalanced	7	YES	5

TABLE VII
HYPERPARAMETERS FOR DECISION TREES

MODEL	DATA	SCENARIO	NORMALIZED	CRITERION	CP	MIN_OBJ
DT	Unbalanced	7	YES	gini	0	2
DT	Balanced	2	YES	gini	0	10

TABLE VIII
HYPERPARAMETERS FOR NAIVE BAYES

MODEL	DATA	SCENARIO	NORMALIZED	LAPLACE	THRESHOLD
Naive Bayes	Unbalanced	2	YES	0	0,1
Naive Bayes	Balanced	1	NO	0	0,1

TABLE IX
HYPERPARAMETERS FOR RANDOM FOREST

MODEL	DATA	SCENARIOS	NORMALIZED	MTRY	NTREE
Random Forest	Unbalanced	2	Yes	3	250
Random Forest	Balanced	2	Yes	1	250

TABLE X
HYPERPARAMETERS FOR SUPPORT VECTOR MACHINES

MODEL	DATA	SCENARIO	GAMMA	COST
SVM	Unbalanced	2	0,01	1000
SVM	Balanced	7	0,1	1000

TABLE XI
HYPERPARAMETERS FOR XGBOOST

MODEL	DATA	SCENARIO	DEPTH	CHILD_WEIGHT	ETA	GAMMA
XGBoost	Balanced	6	8	1	0,001	5
XGBoost	Unbalanced	7	4	1	0,05	0

TABLE XII
HYPERPARAMETERS FOR NEURAL NETWORKS

MODEL	DATA	SCENARIOS	NORMALIZED	LAYERS
Neural Networks	Unbalanced	1	Yes	3
Neural Networks	Balanced	1	Yes	3

TABLE XIII
PERFORMANCE METRICS RESULTS

MODEL	DATA	SCENARIO	NORMALIZED	ACCURACY	AUC	MEAN_RECALL	MEAN_PRECISION	WEIGHTED_F1
K-NN	Balanced	2	YES	0,9276184123	0,7602709956	0,6151587507	0,4478964324	0,5194097372
K-NN	Unbalanced	7	YES	0,9759839893	0,6851834828	0,434067985	0,5458815685	0,5140356683
DT	Unbalanced	7	YES	0,9846564376	0,8479248562	0,6539485767	0,6306945062	0,7371882086
DT	Balanced	2	YES	0,9082721815	0,8566956414	0,5828632317	0,6804177023	0,6980752888
MLR	Unbalanced	1	YES	0,9816544363	0,7820483823	0,5905147208	0,5006439154	0,615857261
MLR	Balanced	2	YES	0,5823882588	0,8444684316	0,4039451021	0,7001080022	0,4994220924
Naive Bayes	Unbalanced	2	YES	0,95496998	0,8145499424	0,3666654216	0,5386613281	0,4437940379
Naive Bayes	Balanced	1	NO	0,5356904603	0,8140047829	0,2952159691	0,7697041024	0,3365664228
Random Forest	Unbalanced	2	YES	0,9889926618	0,8599221438	0,6490260604	0,5959019856	0,7354497354
Random Forest	Balanced	2	YES	0,9543028686	0,8445530187	0,5737796117	0,6815712804	0,7137087493
SVM	Unbalanced	2	YES	0,9866577718	0,8223874217	0,6311467784	0,5616382456	0,7099080945
SVM	Balanced	7	YES	0,8368912608	0,8744102544	0,4483587974	0,7788194787	0,5788444376
Neural Networks	Unbalanced	1	YES	0,9839893262	0,8000203101	0,5972159924	0,5380492184	0,6635796918
Neural Networks	Balanced	1	YES	0,02835223482	0,869358004	0,2853696318	0,6287878788	0,566751258
XGBoost	Balanced	6	YES	0,8925950634	0,8428726603	0,9670967742	0,8735483871	0,9588339083
XGBoost	Unbalanced	7	YES	0,9856571047	0,8482760321	0,5882352941	0,5882352941	0,8100840336