Towards Better Product Quality: Identifying Legitimate Quality Issues through NLP & Machine Learning Techniques

Rakhshanda Jabeen[1,2], Morgan Ericsson[2] and Jonas Nordqvist[3]

Abstract—**Manufacturers of high-end professional products are committed to delivering outstanding customer-quality experiences. They maintain databases of customer complaints and repair service jobs data to monitor product quality. Analyzing the text data from service jobs can help identify common problems, recurring issues, and patterns that impact customer satisfaction, and aid manufacturers in taking corrective actions to improve product design, manufacturing processes, and customer support services. However, distinguishing legitimate quality issues from a brief, domain-specific text in service jobs remains a challenge. This study aims to automate the classification of technical service repair job data into legitimate quality issues or non-issues to assist individuals in the quality field department in a large company. To achieve this goal, we developed a comprehensive pipeline based on natural language processing and machine learning techniques including raw text preprocessing, dealing with imbalance class distribution, feature extraction, and classification. In this study, We evaluate several feature extraction and machine learning classification methods and perform the Friedman test followed by Nemenyi post-hoc analysis to find the best-performing model. Our results show that the passive-aggressive classifier achieved the highest average accuracy of 94% and 89% average macro F1-score when trained on TF-IDF vectors.**

I. INTRODUCTION

With the rapid growth of unstructured data in electronic text formats, natural language processing (NLP) — a subfield of linguistics, computer science, and AI, has emerged as a vital field of research. NLP enables machines to understand and interpret human language. Companies are beginning to recognize the economic value of their text data repositories, including social media platforms and internal document collections, for informed decision-making [1].

The text classification task is one of the most essential tasks in NLP. It involves the automatic categorization of text documents into predefined classes based on their content using machine learning (ML) methods. The process generally includes several steps, including preprocessing (which involves tokenization, stopwords and noise removal, and lemmatization [2]), feature extraction (which involves converting natural language into numerical vectors for mathematical computation), and finally, modeling the data using an appropriate machine learning algorithm for classification. These techniques have a wide range of applications in various industries, such as healthcare, the Internet of Things (IoT), security, spam filtering, digital marketing, and sentiment analysis [3, 4].

This research aims to address the challenge faced by a multinational professional appliance manufacturing company, in the manual categorization of service repair data of machines to filter the legitimate quality issues in service jobs. Technical service agents are responsible for resolving customers' issues and providing a text description of the resolution. The quality field department then manually assesses and classifies the service jobs to determine if it is a genuine quality issue and requires attention at the production and design levels. However, with a growing volume of data in multiple languages, manual classification has become increasingly complex. Therefore, an automatic solution for the classification process is necessary to save time and resources while ensuring consistency and accuracy.

Wang et al. [5] proposed a medical triage system that uses NLP and ML methods to classify questions of patients and text related to their symptoms and to provide suggestions on which consulting room to choose. The system can potentially alleviate the burden on the hospital triage system by helping with disease diagnosis. Additionally, ML-based applications have been found to perform equally or better than individual clinicians, resulting in reduced time and resource requirements for the task [6].

Text classification techniques have also been used to detect spam emails due to the increase in the volume of emails. Spam filters can be implemented at various levels, such as client-level and email servers. Researchers have proposed a significant body of work to automatically and efficiently classify emails as spam or non-spam using NLP and ML methods. Such applications have proven to be effective in reducing the negative impact of spam emails, including wasted time and resources, financial losses, and phishing attacks [7, 8].

NLP and automatic text classification have numerous applications in the financial industry, including fraud detection, stock market predictions, investment recommendation, financial risk management, etc. [9]. Nair et al. [10] proposed a method that uses sentiment analysis of news headlines extracted from the Cryptopanic API to predict the price of Bitcoin. This system has the potential to help novice and professional traders make more profitable investment decisions and reduce the risks associated with cryptocurrency trade.

The successful application of text classification in various industries motivates us to propose an ML-based solution to automatically classify the industrial domain-specific text data effectively. The goal is to automate the classification process of technical service repair jobs data as either legitimate quality issues or not. This research work is focused on answering the following research questions:

1) Propose and implement a comprehensive preprocessing protocol that can effectively address the challenges asso-

[1]Electrolux Professional AB, Sweden
[2]Department of Computer Science and Media Technology, Linnaeus University, Växjö, Sweden
[3]Department of Mathematics, Linnaeus University, Växjö, Sweden

ciated with the dataset, particularly the presence of very brief, multilingual, and domain-specific text data with an imbalanced class distribution.
2) Determine the best-performing feature extraction method for the classification task and investigate its impact on the performance of the classifiers.
3) Evaluate the effectiveness of different machine learning classifiers in classifying service jobs as either genuine quality issues to be approved or non-quality issues to be rejected.

The organization of the paper is as follows. Section II provides an overview of related work, while SectionIII presents a detailed description of the problem. Section IV describes the methodology used to achieve the research objectives, including the description of the data set, text preprocessing techniques, feature extraction approaches, classification algorithms, and performance metrics used to evaluate the classification model. Sections V and VI present the results obtained and their discussion, respectively. Finally, Section VII concludes the work.

II. RELATED WORK

Numerous research studies have been conducted in NLP and ML, aiming to understand human language and address various real-world challenges. Among these efforts, text classification has been one of the primary focuses [11].

For data augmentation in text classification tasks, Wei et al. [12] proposed easy data augmentation (EDA) techniques, including synonym replacement, random insertion, random swap, and random deletion while preserving the original class labels. Their results showed improved classification accuracy on several benchmark datasets and reduced overfitting, especially when trained on smaller datasets. Fromme et al. [13] introduced ContextGen to address the challenge of low-resource domain-specific text classification tasks. They adapt the GPT-2 text generation model to generate domain-specific text samples and then assign labels to these generated samples using BERT to augment training input.

The study by Parmar et al. [14] compares the performance of five classifiers, including Support Vector Machines (SVM), multinomial Naive Bayes, Decision tree, Random forest, and k-nearest neighbors (k-NN), to categorize defects and issues reported in customer complaint messages in an industrial setting. They used the TF-IDF vectorizer for feature extraction and found that SVM achieved the highest accuracy of 63.02% among all classifiers.

Ohata et al. [15] proposed a modular pipeline for a technical support system to automatically categorize incoming customer issues and recommend appropriate solutions based on textual descriptions of the issues. One of the challenges associated with this study is the predominance of short messages and the presence of domain-specific technical terms in the dataset. The authors evaluated and compared various text representation techniques and ML classification methods and found that the Random Forest classifier achieved the highest accuracy of 72.7% and a weighted F1-score of 69.2%.

Hoffimann et al. [16] proposed an automatic classification approach to categorize the daily reports of drilling engineers into three classes aiming to reduce accidents, improve drilling companies' efficiency, and make informed decisions. The challenges present in the text corpus include technical symbols, abbreviations of technical terms, misspellings, and truncated sentences. They used skip-gram a variant of Word2Vec embeddings for feature extraction and three neural networks and found that LSTM performs best for the task with an average accuracy of 82.7%.

III. PROBLEM STATEMENT

At Electrolux Professional, the customer support centers collect information on customer complaints about machines installed in 17 countries in text form. The customer support center assigns the task of addressing these complaints to a technical service agent, who visits the site, resolves the issue, and records the service job in the database. The service agent provides a text description of the resolution, including details of any faulty components and defects in the machine, selected from a pre-defined set of codes. In the subsequent phase, the individuals in the quality field department manually evaluate the job to determine whether it constitutes a genuine quality issue that requires attention at the production and design levels. This evaluation involves categorizing the service jobs into two groups: approved as a quality issue or rejected as not a quality issue. The rejected calls are further categorized into different categories that include maintenance that should be done by the customer, installation fault, customer misuse of the product, and consumables that customers should replace.

As the company expands its business to various locations, it acquires more data related to service jobs. During the past five years, the quality field department has manually classified service jobs in various languages during the machine warranty period. In our preliminary analysis of historical data, we discovered that a significant number of rejected and approved service jobs share similar characteristics. Automating the classification of technical service job text can help save time and resources while also ensuring consistency and accuracy in the classification process. Additionally, an automated approach can quickly identify quality issues and prioritize them based on their severity and impact on customer satisfaction. Therefore, in this study, we propose an ML-based solution to automate the classification process for the quality field department.

One of the challenges associated with the dataset is the installation of machines in multiple countries and the use of local service agents to resolve the issues, resulting in various reporting styles and languages. The data set mainly comprises text generated by human operators and technicians, which contains instances of misspellings, abbreviations, and inconsistencies in the representation of faulty component codes for similar tasks. We have implemented a detailed preprocessing protocol to address these challenges, which is outlined in the following section.

IV. METHODOLOGY

In the following sections, we will elaborate on the schematic workflow of the proposed methodology, as shown in 1, which outlines the classification process for service jobs. This methodology is composed of several phases, including preprocessing, addressing the issue of imbalanced class distribution, feature extraction, and classification. Finally, the performance metrics are evaluated and the analysis of the results is presented.

A. Dataset

The dataset used in this study is derived from the Electrolux Professionals service records database related to the laundry machines within the warranty period of machines. The service jobs have been manually reviewed and approved or rejected by the field quality team for the last five years. The dataset included customer complaints, technical comments in 17 languages, and other relevant metadata for each service job. Around 84% of these jobs are approved as quality calls when manually annotated, while nearly 16% of jobs are rejected, indicating a considerable imbalance between the two classes.

B. Preprocessing

The preprocessing phase is a fundamental step in Natural Language Processing (NLP) and machine learning classification, as it helps to comprehend the desired outcome. This study deals with unstructured, brief multilingual text data that lacks sufficient information for the classification task. The dataset includes categorical features like faulty component codes, defect codes, and the cost of replaced components, along with textual features technician reports, and customer complaints. Our analysis revealed that adding further categorical features and associated metadata could significantly enhance the predictive strength of the model. Therefore, we mapped faulty component codes to their corresponding names, main groups, and subgroups, considering the possibility of a single name referring to either an electrical or mechanical component. We then combined these informative categorical features with our textual features, such as customer complaints and technical comments, to create a single document for each service job. This approach aimed to improve the overall predictive strength of the model by utilizing both categorical and textual information comprehensively.

This data presents multiple challenges, including multilingual text, missing or incomplete sentences, incorrect spellings, and varying terminology and abbreviations used by different technicians in the reports. In addition, the encoding of special characters from non-English languages in older reports is flawed and has been replaced with symbols such as #, {, $, etc. To address these limitations, we conducted a manual search to identify and correct common misspellings and aimed to establish a uniform language representation of text in all languages. To achieve this, we created a dictionary that maps domain-specific technical terms and abbreviations from all languages to English and used the Google Translation API to translate the text data into English.

In summary, the preprocessing techniques involve common misspellings corrections, translation, conversion to lowercase, tokenization, lemmatization, and removal of stopwords, punctuation, and extra white spaces. Afterward, we split the data into two sets with a ratio of 80% for training and 20% for testing the model.

C. Dealing with Imbalanced Class Distributions

The dataset represents an imbalanced class distribution, with the class "approved" being overrepresented. Such an imbalance in the classes can hinder the generalization of classification models, as they might be biased toward the overrepresented class. Following the strategy of Wei et al. [12], we used random synonym replacement to augment the minority class. This technique involves randomly selecting n nouns and verbs from a given document and replacing them with their corresponding synonyms using a certain probability. Specifically, we randomly selected a subset of 60% training inputs from the minority class in the training dataset and applied the synonym replacement from WordNet [17] with a probability of 0.5 on all documents and then added these instances in the training set.

D. Feature Extraction

When dealing with NLP applications, finding an appropriate numerical representation of text data is essential to make it mathematically computable for a machine learning algorithm. There has been extensive research on various feature extraction methods for numerical representation of text [18, 19, 20, 21, 22]. Different feature extraction techniques highlight different features of the data and produce varying outcomes. Therefore, selecting a suitable representation of the text data significantly affects the text classification experiments. In this study, we have used four different feature extraction techniques to numerically vectorize the tokens of service jobs as follows:

1) Term frequency-inverse document frequency (TF-IDF) [23] is a conventional term-weighing technique to extract features from text data. TF-IDF captures the importance of a term in a document by assigning a high weight to terms that appear frequently in the document, but not so often in other documents.
2) Word2Vec [20] is a neural network-based predictive word embedding technique that can be further divided into two variants; the continuous bag of words (CBOW) and skip-gram. We utilized the skip-gram method that learns word vectors by training the network to predict the context of a word within a fixed window, given the word. In our experiments, we trained the Word2Vec model on our training data for 20 epochs, setting the context window equal to 5. Each word in the corpus was embedded into a 300-dimensional vector, and service jobs were represented as the average of their respective word vectors.
3) Doc2Vec [21] is a predictive document embedding technique that extends the concept of Word2Vec to generate embeddings for documents. The embeddings

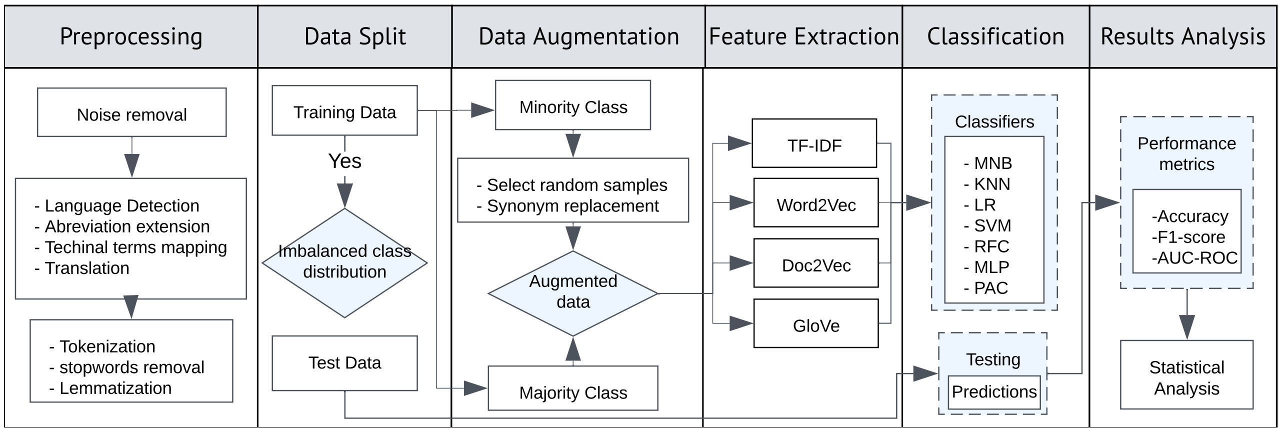

Fig. 1: The schematic workflow of the proposed methodology. First, the manually tagged historical data is preprocessed using various NLP techniques. Next, a data augmentation technique is applied to the minority class of the training data. Then feature extraction models are trained on training data and features are fed into one of the six classification models and then tested on the test data. Finally, a result analysis is presented to evaluate the performance of the model.

can be learned using either paragraph vector distributed memory (PV-DM) or paragraph vector distributed bag of words (PV-DBOW) model. In our work, we used DBOW to learn the document vectors from our training data and embed each service job into a 300-dimensional vector. The model was trained for 20 epochs with a context window of size 5.

4) GloVe (Global vectors for word representation) [22] is a count-based word embedding technique that leverages global word-to-word occurrence counts and statistical information to learn word vectors. Our study used a pre-trained GloVe word embedding with 300 dimensions. Similar to the Word2Vec approach, we obtained a vector representation of service jobs by taking the average of the word vectors for each word in a document, resulting in vectors with a dimension of 300.

E. Classification

Given the labeled service jobs data, we utilized supervised machine learning techniques for classification. This involved training a model on the labeled dataset and using it to make predictions on new data. In this study, we evaluated the performance of traditional and state-of-the-art machine learning models that are widely used in text classification, drawing inspiration from prior research that has demonstrated their effectiveness in various applications, such as sentiment analysis, spam filtering, and document categorization [24, 25, 26]. The classification models evaluated in this research are as follows:

1) Naive Bayes (NB) [27], a simple probabilistic algorithm that is computationally efficient and works well with high-dimensional data. However, NB assumes that predictor features are independent and uncorrelated, which may not hold in several real-world scenarios.
2) k-Nearest Neighbor (k-NN) [28], a non-parametric algorithm that classifies new instances based on their similarity to the k nearest neighbors in the training set. The algorithm makes no assumptions about the data distribution and can handle non-linear boundaries. however, it may be computationally inefficient and sensitive to the choice of hyperparameter k.
3) Logistic Regression (LR) [29], a statistical algorithm that utilizes the logistic function to predict the probability of each observation belonging to a certain class. LR is easier to implement and interpret and can handle large datasets. However, it assumes that data features are independent of each other and follow a linear relationship with the response variable.
4) Support Vector Machines (SVM) [30] is a non-parametric algorithm that finds a hyperplane that best separates the classes. This hyperplane is called the decision boundary. SVM can learn linear and non-linear boundaries between classes and is effective in high-dimensional data. However, it requires a careful selection of hyperparameters and takes a long training time on large datasets. Moreover, it is sensitive to the choice of the kernel function.
5) Random Forest (RF) [31] is an ensemble learning algorithm that combines multiple decision trees and its output is the mean prediction of individual trees. The RF algorithm can handle both linear and complex relationships and explicitly performs feature selection. However, RF cannot be easily interpreted and can be computationally expensive for large datasets.
6) Multi-Layer Perceptron (MLP) [32] is a feedforward neural network architecture that uses multiple layers of neurons (at least three layers including an input layer, a hidden layer of neurons, and an output layer). MLP can capture complex, non-linear patterns and relationships between data and works well on large datasets. However, MLPs are fully connected neural networks, resulting in too many hyperparameters that require careful tuning. In our experiments, we used one or two hidden layers.

7) Passive-Aggressive (PA) [33] is an online learning linear algorithm that learns incrementally and updates its weights based on a passive-aggressive strategy. The algorithm makes predictions based on the current weights when it receives a new input. If the prediction is correct, then the model acts passively, and the weights remain unchanged. If the prediction is incorrect, the algorithm acts aggressively and updates the weights to minimize the loss by adding a regularization parameter (C) that penalizes the weight vector. PA is known for its ability to adapt quickly to dynamic changes in the data. However, it cannot capture complex, nonlinear decision boundaries between classes and is sensitive to the choice of the regularization parameter.

The classification process is divided into two phases training and testing. In the training phase, 80% of the data is used to train the classification model. For this purpose, we created a pipeline that involves augmenting the minority class data to make it comparable with the majority class as mentioned in IV-C, followed by feature extraction and then classification using the classification algorithm to classify the service jobs. We optimized the model's performance by tuning the hyperparameter using an exhaustive search with 5-fold cross-validation (CV) over the hyperparameter configurations listed in Table I. This search was conducted for all combinations of feature extraction techniques and classification algorithms. The resulting optimal hyperparameters were then used to train the models. In the testing phase, the remaining 20% labeled data is vectorized using the feature extraction approach of the pipeline and then fed into the model for classification, and performance metrics are computed.

To evaluate the classification models, we used a 10-fold stratified CV method that divides the data into ten subsets of roughly equal size, with each subset imitating the class distribution in the entire dataset. We then trained and tested the model 10 times, using each subset once as the testing subset and the remaining subsets as the training subsets.

F. Evaluation Metrics & Statistical Analysis

When dealing with imbalanced class distribution in data, the accuracy score for evaluating the performance of models can be misleading. Therefore, it is necessary to select a metric that offers a more comprehensive assessment of the model and is insensitive to changes in data distribution. In this study, we have evaluated the predictive performance of classification models using the macro F1-score and the area under the Receiver Operating Characteristic (ROC) curve (AUC-ROC) scores.

The F1-score is the harmonic mean of precision and recall, where precision is the proportion of positive predictions made by the model and recall is the proportion of true positive cases correctly classified by the model. The macro F1-score is then determined by calculating the arithmetic mean of the F1-scores for all classes.

The ROC curve is a graphical representation of a classification model's performance obtained by plotting the true positive rate (TPR) on the y-axis against the false positive rate (FPR) on the x-axis. TPR is the same as recall, while FPR is the proportion of actual negative cases incorrectly classifies as positive by the model. The AUC-ROC score ranges from 0 to 1, where higher values indicate better performance.

To verify if there is a significant difference in the model's performance, we conducted Friedman test [34] at a significance level of 0.05 on the ranks of metric scores obtained from 10 CV runs of all classifiers. If the resulting p-value of the test statistic is less than 0.05, we reject the null hypothesis that all classifiers perform equally in favor of the alternative hypothesis that at least one classifier performs differently from the others. To identify which classifiers differ significantly, we performed the Nemenyi post-hoc test at a significance level of 0.05 and computed the critical difference (CD) based on the average ranks of all CV scores. If the mean ranks differ by at least CD, the two classifiers are considered significantly different.

G. Experimental Setup

In this study, we conducted 28 experiments exploring all possible combinations of feature extraction techniques and classification models described in the previous sections. The implementation was carried out in Python 3.10.6, with the aid of general machine learning and NLP tools such as scikit-learn [35], Gensim [36], spaCy [37], and NLTK [38].

TABLE I: Hyperparameters optimization setup for all classifiers

Classifier	General Setup	Hyperparameters	Grid setup
NB	multinomial	–	–
k-NN	–	no. of neighbors	$[3, 5, 7, 9, 11, 13, 15]$
LR	penalty: l_2	C	logspace$(-3, 3, 10)$
SVM	Kernel: rbf	C	$[0.1, 1, 5, 10, 100]$
		γ	$[0.001, 0.01, 0.1, 1]$
MLP	activation: $relu$ solver: $adam$	hidden layers	$[(10,), (50,), (100,), (10, 10), (50, 50), (100, 100)]$
RF	criterion: $gini$	no. of trees	$[10, 50, 100, 200, 500]$
		maximum depth	[None, $10, 50, 100]$
PA	loss: $hinge$	C	$[0.001, 0.01, 0.1, 1, 10, 100, 1000]$

V. RESULTS

In this section, we present the results obtained from our experiments as described in IV. Table II reports the average scores and 95% confidence intervals (CIs) computed over 10 runs for each combination of the feature extraction method and classifier. The width of the CI reflects the level of uncertainty of the estimate, and a narrower CI indicates a more precise estimate. To obtain a reliable performance evaluation, it is desirable to have small CIs.

Based on the results, it is evident that the TF-IDF feature extraction approach yields the highest accuracy, macro F1, and AUC-ROC scores among all evaluated classifiers. Additionally, when combined with TF-IDF, LR, SVM, RF, MLP, and PA classifiers outperform NB and k-NN classifiers.

TABLE II: Accuracy, macro F1, and AUC-ROC scores for all combinations of feature extraction method and classifier

Feature Extraction	**Classifier**	**Accuracy (%)**	**F1-Score (%)**	**AUC-ROC (%)**
Doc2Vec	NB	88 ± 0.4	78 ± 0.6	77 ± 0.7
	k-NN	91 ± 0.9	83 ± 1.6	83 ± 1.7
	LR	90 ± 0.5	82 ± 1.4	87 ± 4.3
	SVM	92 ± 0.5	85 ± 0.9	86 ± 3.7
	RF	92 ± 0.5	85 ± 0.9	86 ± 3.2
	MLP	90 ± 1.3	83 ± 1.8	82 ± 3.0
	PA	90 ± 0.7	83 ± 1.5	83 ± 1.4
GloVe	NB	87 ± 0.4	70 ± 1.0	81 ± 1.5
	k-NN	90 ± 1.1	83 ± 1.9	82 ± 1.9
	LR	90 ± 1.1	83 ± 1.8	82 ± 2.1
	SVM	93 ± 0.8	85 ± 1.8	89 ± 1.5
	RF	92 ± 0.7	82 ± 1.3	90 ± 1.2
	MLP	92 ± 0.4	85 ± 0.9	86 ± 0.9
	PA	90 ± 0.6	83 ± 0.9	82 ± 1.1
Word2Vec	NB	88 ± 0.9	78 ± 1.9	78 ± 1.5
	k-NN	92 ± 0.2	86 ± 0.6	86 ± 0.2
	LR	92 ± 0.4	85 ± 1.0	85 ± 0.6
	SVM	93 ± 0.9	87 ± 1.8	90 ± 1.4
	RF	93 ± 0.6	86 ± 1.3	92 ± 1.8
	MLP	93 ± 1.0	86 ± 0.7	89 ± 2.1
	PA	92 ± 0.7	86 ± 1.3	85 ± 1.1
TF-IDF	NB	90 ± 0.8	83 ± 1.3	81 ± 1.5
	k-NN	91 ± 0.8	85 ± 1.4	85 ± 1.4
	LR	94 ± 0.9	88 ± 1.6	90 ± 1.8
	SVM	94 ± 0.7	88 ± 1.4	91 ± 1.2
	RF	93 ± 0.8	87 ± 1.5	90 ± 1.6
	MLP	94 ± 0.5	88 ± 0.8	90 ± 1.1
	PA	94 ± 0.6	89 ± 1.0	91 ± 1.3

A. Statistical Analysis

The experimental results presented in Table II demonstrate that a direct comparison of classifiers using the best feature extraction technique, TF-IDF, can be misleading, as there is minimal difference in accuracy, macro F1-scores, and AUC-ROC scores of the models. Therefore, we applied the Friedman test on the ranks of 10 CV runs of all classifiers when combined with TF-IDF. The results of the Friedman test are shown in Table III. All p-values are less than the significance level of 0.05, leading us to reject the null hypothesis and conclude that at least one classifier performs differently from the others.

TABLE III: Results of the Friedman Test on CV scores of all classifiers in combination with TF-IDF vectorization

Score	**Friedman's statistics**	**p-value**
Accuracy	34.66	10^{-6}
Macro F1	31.02	10^{-5}
AUC-ROC	36.43	10^{-6}

According to the Nemenyi post-hoc test, the CD is 3.404. The pairwise comparison of the average ranks of all classifiers, using TF-IDF vectorization, is presented in Figure 2. Groups of classifiers that are not significantly different (at the significance level of 0.05) are connected with a horizontal line, and the length of the line between the two classifiers indicates the difference between them. Furthermore, the lower the rank, the better the classification model in terms of the corresponding performance measure.

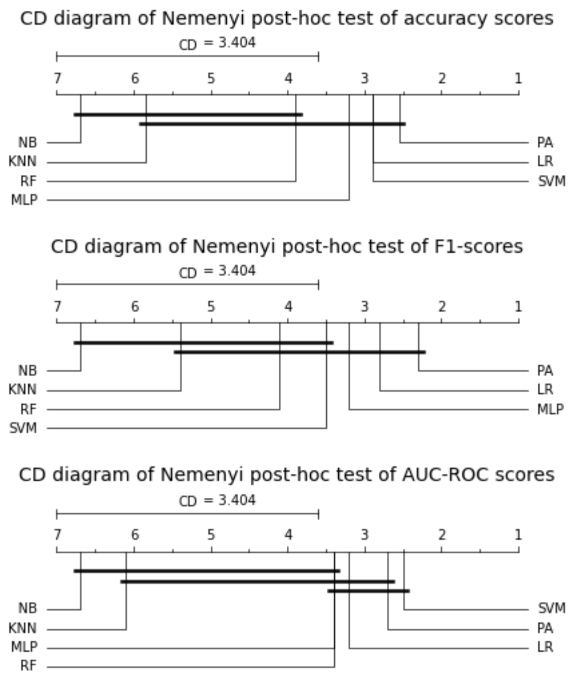

Fig. 2: Nemenyi test results on pairwise comparison on mean ranks of accuracy, F1 and AUC-ROC scores of all classifiers

VI. DISCUSSION

In this section, we will provide a conclusive analysis of the methodologies and results used to achieve the goals of this research. Finally, We will discuss the limitations of the methodology and datasets used.

A. Comparison of Feature Extraction Methods

The results in Table II show that, regardless of the paired classification model, Doc2Vec embeddings exhibited weaker predictive strength in terms of accuracy, F1-score, and AUC-ROC score than the other feature extraction techniques. The only exception was the NB classifier, which performed better when trained on Doc2Vec features than GloVe. SVM and RF classifiers achieved the highest accuracy and F1-scores when trained on Doc2Vec embeddings, while LR classifiers achieved the highest AUC-ROC score. One possible reason for this could be that Word2Vec and Doc2Vec are unsupervised feature extraction techniques that require a large corpus of text to learn a meaningful representation of the text, which is not available in our small, domain-specific corpus.

Furthermore, the GloVe feature extraction method performed slightly better than Doc2Vec, achieving the best accuracy score of 93% using the SVM classifier. The highest F1-score of 85% was achieved by SVM and MLP, closely followed by the k -NN, LR, and PA classifiers. However, Word2Vec performed better than GloVe and Doc2Vec, achieving the highest accuracy score of 93% using SVM, RF, and MLP classifiers, followed closely by k-NN, LR, and PA classifiers with an accuracy score of 92%.

The TF-IDF feature extraction approach achieves the highest accuracy, F1-score, and AUC-ROC score across all evaluated classifiers. MLP, PA, and SVM classifiers achieved the highest accuracy score of 94% when using the TF-IDF feature extraction method. Furthermore, the PA classifier achieved the best macro F1-score of 89%, closely followed by the MLP, SVM, and LR classifiers. This could be attributed to the fact that the dataset contains domain-specific terms and abbreviations that are less frequent in general language usage. Furthermore, our documents have a more limited vocabulary, with a greater emphasis on specialized terms and jargon, causing TF-IDF to perform better as it assigns more weight to the terms that are specific to a certain domain.

In conclusion, the TF-IDF feature extraction method is the most effective method, irrespective of the choice of classifier in this task. However, Word2Vec and GloVe can also be used as effective alternatives to TF-IDF, particularly when using MLP and SVM classifiers.

B. *Comparison of Classifiers*

Based on the Nemenyi post-hoc analysis of the results of all classifiers when combined with the TF-IDF feature extraction approach (see figure 2), it can be inferred that PA, LR, SVM, and MLP achieved significantly higher accuracy scores than NB. Additionally, NB, k-NN, and RF classifiers did not show any significant differences in their accuracy scores. The Nemenyi post-hoc test on F1 scores showed that PA performed significantly better than NB, but no significant difference was found between PA, LR, MLP, and SVM in terms of F1-scores. Moreover, the Nemenyi post-hoc test on AUC-ROC scores revealed that SVM performed significantly better than NB and k-NN. At the same time, no significant differences were found between the SVM, PA, RF, and MLP classifiers.

The results in Table II suggest that the LR and SVM classifiers exhibit greater consistency across all feature extraction methods. However, the PA classifier achieves the highest accuracy and F1-score among all classifiers when trained on TF-IDF vectors. It is important to note that these findings may not generalize to all datasets and classification tasks, since the performance of different classifiers can depend on the unique characteristics of the data.

C. *Potential Applications*

Electrolux Professional's quality field department aims to consistently improve the quality of their machines by monitoring the KPI Service Call Rate (SCR). This study contributes to more accurate SCR calculations by identifying legitimate quality issues in service jobs, resulting in time and resource savings, as well as ensuring consistency and accuracy in the classification process

D. *Limitations*

During our experiments, we discovered several limitations that hinder the learning process of a machine learning model. Firstly, we observed inconsistencies in the labeling of service jobs, which could be attributed to the manual reading and labeling of the dataset by different quality team members. Secondly, customer complaints do not always provide sufficient information to accurately diagnose the machine issue. Furthermore, some technicians provide an insufficient job description, leading to a very short text and a lack of understanding of the problem's nature.

Another significant limitation that we encountered is the insufficient number of labeled data to train the model. The dataset has a skewed distribution, with the minority class divided into several subgroups, making it challenging to accurately classify. However, this limitation can be overcome by generating model predictions and manually reviewing instances with low confidence in the predictions. In summary, our experiments revealed that inconsistent labeling, insufficient information in customer complaints and technician descriptions, and a scarcity of labeled data are the primary factors that need to be addressed to improve the model's performance.

In addition to the techniques explored in this research, another potential solution for this problem could be found in transfer learning. Transfer learning involves training a model on a large, general dataset and then fine-tuning it on a smaller, more specific dataset. This approach is useful in real-world scenarios where we have limited labeled data, as it allows the model to leverage knowledge learned from a larger, more diverse dataset.

VII. CONCLUSION

The study aimed to automate the classification of technical service repair job data into legitimate quality issues or non-issues. In this concluding section, we present a comprehensive analysis of the answers to the research questions detailed in Section I.

To effectively address the challenges associated with the dataset, we proposed and implemented a comprehensive preprocessing protocol. One of the challenges was the predominance of very brief, multilingual, and domain-specific text data. To address this, we manually created a dictionary of technical terms and jargon used in the technicians' reports. We then used the Google Translation API in our preprocessing pipeline to translate the data into English. In addition, to tackle the imbalanced class distribution problem, we implemented a data augmentation technique that randomly replaced some terms in a document with their synonyms at a certain probability to make the minority class comparable to the majority class. This preprocessing protocol has been effective in improving the overall accuracy of the classifiers.

We evaluated several feature extraction methods, including TF-IDF, Word2Vec, Doc2Vev, and GloVe, and found that TF-IDF outperformed the other methods for this classification task across all evaluated classifiers. TF-IDF's superior performance can be attributed to its ability to weigh technical terms and jargon effectively in domain-specific text. However, it is important to note that TF-IDF cannot capture the syntactic and semantic relationships between words and may suffer from sparsity when dealing with very short text documents.

We used seven machine learning algorithms in our study and found that the passive-aggressive classifier achieved the highest accuracy of 94% and an F1-score of 89% when trained on TF-IDF vectors. Therefore, we incorporated the PA classifier into our proposed framework, knowing that it can quickly adapt to rapid changes in the data and learns incrementally. However, PA assumes that classes are linearly separable and may not perform as well when dealing with complex data relationships.

In conclusion, the ML-based solution proposed in this study can effectively automate the classification of technical service job data, improve the efficiency of the quality field department, and save time. The techniques used in this approach can be extended to similar problems in other industries. Future work will focus on developing a solution for technicians and the customer care department to identify possible problem resolutions before visiting the site using only customer complaint data and historical records.

References

[1] A. Ittoo, A. van den Bosch *et al.*, "Text analytics in industry: Challenges, desiderata and trends," *Computers in Industry*, vol. 78, pp. 96–107, 2016.

[2] M. Anandarajan, C. Hill, T. Nolan, M. Anandarajan, C. Hill, and T. Nolan, "Text preprocessing," *Practical text analytics: Maximizing the value of text data*, pp. 45–59, 2019.

[3] M. K. Dalal and M. A. Zaveri, "Automatic text classification: a technical review," *International Journal of Computer Applications*, vol. 28, no. 2, pp. 37–40, 2011.

[4] M. O. Aftab, U. Ahmad, S. Khalid, A. Saud, A. Hassan, and M. S. Farooq, "Sentiment analysis of customer for ecommerce by applying ai," in *2021 International Conference on Innovative Computing (ICIC)*, 2021, pp. 1–7.

[5] X. Wang, M. Tao, R. Wang, and L. Zhang, "Reduce the medical burden: An automatic medical triage system using text classification bert based on transformer structure," in *2021 2nd International Conference on Big Data & Artificial Intelligence & Software Engineering (ICBASE).* IEEE, 2021, pp. 679–685.

[6] S. Delshad, V. S. Dontaraju, and V. Chengat, "Artificial intelligence-based application provides accurate medical triage advice when compared to consensus decisions of healthcare providers," *Cureus*, vol. 13, no. 8, 2021.

[7] S. Magdy, Y. Abouelseoud, and M. Mikhail, "Efficient spam and phishing emails filtering based on deep learning," *Computer Networks*, vol. 206, p. 108826, 2022.

[8] T. Peng, I. Harris, and Y. Sawa, "Detecting phishing attacks using natural language processing and machine learning," in *2018 ieee 12th international conference on semantic computing (icsc).* IEEE, 2018, pp. 300–301.

[9] A. Gupta, V. Dengre, H. A. Kheruwala, and M. Shah, "Comprehensive review of text-mining applications in finance," *Financial Innovation*, vol. 6, pp. 1–25, 2020.

[10] K. Nair, A. Pawle, A. Trisal, and S. Krishnan, "Bitcoin price prediction using sentimental analysis - a comparative study of neural network model for price prediction," in *2022 2nd Asian Conference on Innovation in Technology (ASIANCON)*, 2022, pp. 1–4.

[11] Q. Li, H. Peng, J. Li, C. Xia, R. Yang, L. Sun, P. S. Yu, and L. He, "A survey on text classification: From shallow to deep learning," *arXiv preprint arXiv:2008.00364*, 2020.

[12] J. Wei and K. Zou, "Eda: Easy data augmentation techniques for boosting performance on text classification tasks," *arXiv preprint arXiv:1901.11196*, 2019.

[13] L. Fromme, J. Bogojeska, and J. Kuhn, "Contextgen: Targeted data generation for low resource domain specific text classification," in *International Conference on Artificial Intelligence and Statistics.* PMLR, 2022, pp. 3016–3027.

[14] P. S. Parmar, P. Biju, M. Shankar, and N. Kadiresan, "Multiclass text classification and analytics for improving customer support response through different classifiers," in *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI).* IEEE, 2018, pp. 538–542.

[15] E. F. Ohata, C. L. C. Mattos, S. L. Gomes, E. D. S. Rebouças, and P. A. L. Rego, "A text classification methodology to assist a large technical support system," *IEEE Access*, vol. 10, pp. 108 413–108 421, 2022.

[16] J. Hoffimann, Y. Mao, A. Wesley, and A. Taylor, "Sequence mining and pattern analysis in drilling reports with deep natural language processing," in *SPE Annual Technical Conference and Exhibition.* OnePetro, 2018.

[17] G. Miller, "&quot; wordnet: An on-line lexical database, &quot," *International Journal of Lexicography*, vol. 3, no. 4, pp. 235–244, 1990.

[18] V. John, "A survey of neural network techniques for feature extraction from text," *arXiv preprint arXiv:1704.08531*, 2017.

[19] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint*

arXiv:1810.04805, 2018.

[20] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *Advances in neural information processing systems*, vol. 26, 2013.

[21] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International conference on machine learning*. PMLR, 2014, pp. 1188–1196.

[22] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[23] J. Ramos *et al.*, "Using tf-idf to determine word relevance in document queries," in *Proceedings of the first instructional conference on machine learning*, vol. 242, no. 1. Citeseer, 2003, pp. 29–48.

[24] R. Li, "A review of machine learning algorithms for text classification," *Cyber Security*, p. 226, 2022.

[25] X. Deng, Y. Li, J. Weng, and J. Zhang, "Feature selection for text classification: A review," *Multimedia Tools and Applications*, vol. 78, pp. 3797–3816, 2019.

[26] K. Nagashri and J. Sangeetha, "Fake news detection using passive-aggressive classifier and other machine learning algorithms," in *Advances in Computing and Network Communications: Proceedings of CoCoNet 2020, Volume 2*. Springer, 2021, pp. 221–233.

[27] I. Rish *et al.*, "An empirical study of the naive bayes classifier," in *IJCAI 2001 workshop on empirical methods in artificial intelligence*, vol. 3, no. 22, 2001, pp. 41–46.

[28] A. Mucherino, P. J. Papajorgji, P. M. Pardalos, A. Mucherino, P. J. Papajorgji, and P. M. Pardalos, "K-nearest neighbor classification," *Data mining in agriculture*, pp. 83–106, 2009.

[29] D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant, *Applied logistic regression*. John Wiley & Sons, 2013.

[30] M. Awad, R. Khanna, M. Awad, and R. Khanna, "Support vector machines for classification," *Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers*, pp. 39–66, 2015.

[31] L. Breiman, "Random forests," *Machine learning*, vol. 45, pp. 5–32, 2001.

[32] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA: MIT Press, 2016. [Online]. Available: https://www.deeplearningbook.org/

[33] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer, "Online passive aggressive algorithms," 2006.

[34] K. Stąpor, "Evaluating and comparing classifiers: Review, some recommendations and limitations," in *Proceedings of the 10th International Conference on Computer Recognition Systems CORES 2017 10*. Springer, 2018, pp. 12–21.

[35] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

[36] R. Rehurek and P. Sojka, "Gensim–python framework for vector space modelling," *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, vol. 3, no. 2, 2011.

[37] M. Honnibal, I. Montani, S. Van Landeghem, and A. Boyd, "spaCy: Industrial-strength Natural Language Processing in Python," 2020.

[38] S. Bird, E. Klein, and E. Loper, *Natural language processing with Python: analyzing text with the natural language toolkit*. O'Reilly Media, Inc., 2009.