Utilizing k-Nearest Neighbors and Support Vector Machine for Ticket Support System

Faqihah Binti Zakir [1], Su-Cheng Haw [1], Tong-Ern Tai [1], Kok-Why Ng [1], Maw Maw [1]

[1] *Faculty of Computing and Informatics, Multimedia University, Jalan Multimedia, 63100 Cyberjaya, Malaysia*

Abstract **– Support tickets are vital for providing high-quality customer service. Ensuring very customer has a positive experience, regardless of their issue, is essential. This paper explores optimizing support ticket systems identifying common algorithms and predicting ticket resolution time. Leveraging Machine Learning, specifically k-Nearest Neighbours (KNN) and Support Vector Machines (SVM), the study aims to predict resolution times using historical data. These algorithms will be evaluated using Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) metrics. The goal is to assess the effectiveness of KNN and SVM in improving support processes. Accurate time predictions can help businesses allocate resources proactively, minimize delays, and streamline ticket assignments. By applying these models, organizations can enhance response efficiency and reduce the impact of unresolved issues.**

Keywords **– Ticket system, machine learning, prediction, customer support, resolution time.**

DOI: 10.18421/TEM153-07
https://doi.org/10.18421/TEM153-07

Corresponding author: Su-Cheng Haw,
Faculty of Computing and Informatics, Multimedia University, Jalan Multimedia 63100 Cyberjaya, Malaysia.
Email: sucheng@mmu.edu.my

Received: 07 August 2025.
Revised: 13 February 2026.
Accepted: 25 March 2026.
Published: 27 August 2026.

1. Introduction

The rise in Customer Service needs results from business growth and growing emphasis on customer happiness. Organizations realized how crucial good customer service is to attract and retain customers as marketplaces expanded and competition increased. Technical support is an essential component of most businesses [1], [2]. Ticket support systems have been developed in response to an increasing number of customer inquiries, issues, and requests. This has led to the need for an organized method to handle these interactions. These systems offer a centralized platform for managing client complaints, expediting correspondence, and guaranteeing prompt resolution [3]. Therefore, a good ticketing system must include features and functionalities to help organizations manage and improve their resolution times [4]. Large firms frequently use temporary or less-skilled employees to distribute support tickets, or they employ third-party assistance because it may occupy a huge amount of time for their workers. The manual procedure for dispersing emerging support tickets by human staff, who are frequently less skilled, is both costly and ineffective [5]. However, it lengthens the time it takes to resolve tickets, which may indirectly cause customer dissatisfaction [2], [6], [7].

With the widespread use of machine learning (ML) and Artificial Intelligence (AI) methods, the automation of support ticket systems has become increasingly compelling. Automated technologies can potentially replace first-level support staff by automatically classifying problems and resolving tickets using ML [8], [9]. To speed up ticket assignment and completion, implementing a predictive model that uses ML techniques and the underlying pattern of historical data would provide more effectiveness and productivity [10]. The trained ML model automatically determines how long it will take to answer a query for any new incoming ticket. With the advancement of ML, tickets can now be categorized, and the time needed for resolution is automatically estimated [11], [12], [13], [14].

Currently, most Information Technology (IT) firms use Ticket Support Systems to deal with customers' problems and issues [15]. The main objectives of a ticket support system are to streamline communication, prioritize tasks, and ensure the timely resolution of client issues. They are essential for raising customer satisfaction, streamlining support operations, and offering an organized method for handling and resolving client complaints. A troubled ticket is a digital log that documents the details of an outage, including the workflow and actions performed. Users are unable to use their equipment during outages, and they frequently do not know when the problem will be resolved. Because of this unpredictability, resources are wasted on temporary fixes for short outages or inadequate planning for extended outages. Enabling more informed mitigation decisions during issue-ticket submission is the ability to predict resolution time [16].

Ticket support systems have several limitations. These include high ticket volumes, which enable incorrect classifications and inconsistent response quality, limited predictability for ticket resolution times, excessive knowledge bases, and a lack of automation in routine tasks. These factors proceed from general factors, such as rapidly growing customer expectations for prompt issue resolution. Moreover, the tool is necessary for allocating and organizing resources, tracking, and analyzing customer data. Furthermore, ticket support addresses the need to support knowledge management and to maintain consistent communication. In addition, a high incoming ticket count can be a problem. Large entities must handle high volumes of incoming data. It is challenging to achieve this in the most efficient manner, which results in a response time. Therefore, the ticketing system needs to be improved.

One alternative for optimizing the system is to determine which incident tickets are more urgent and to find a way to reduce the ticket resolution time. In a general approach, every incident ticket in a defined IT environment is first split into different sets to clarify the aspects of the problems associated with a system. Moreover, they assist in defining requests, events, and alerts. Therefore, effective ticket management should address the most severe or frequent issues. However, incident ticket clustering and categorization issues can arise for several reasons. First, in a large IT ecosystem, the number of tickets would increase to 1000 years, making classification impossible [17]. ML can optimize a ticket support system in several ways, making its experience more efficient and smoother. First, ML techniques can automatically assign priority levels to tickets and even guess when a ticket with this particular issue will be resolved. When you need a ticket in a support system, it is crucial to make professionals aware of how much time to expect is dedicated to your issues.

An ML model can calculate the amount of time tickets of a particular type have been solved before assuming a newly arrived ticket. The estimation can be further refined by considering the complexity of the issue, the type of ticket, and the average resolution time. ML has opened a way for automated ticket classification. This enables customer support to see the time it takes to settle a case. Most of the time, ML algorithms are designed to create a model based on data, without the need for human interference. Some of the possible automated features include face recognition, speech recognition, prediction, and forecasting.

Typically, in a prediction resolution time, ML models can examine historical data to resolve new incoming ticket prediction times. Categorizes them based on their urgency and is used for training. A few ML algorithms typically used in ticket system classification and Prediction Time Resolution include k-nearest neighbors (KNN), Support Vector Machines (SVM), and Naïve Bayes.

Supervised learning, a subset of ML, involves training algorithms to accurately predict outcomes or classify data using labelled datasets. The model modifies its weights as it analyzes the input data during the cross-validation procedure until the data properly fits the model. Companies utilize supervised learning to address various real-world challenges on a large scale, such as data categorization. Some of the ML techniques in this group are Random Forest (RF), SVM, Kernelized SVM, and KNN.

Unsupervised ML algorithms were crafted to analyze and classify unlabelled datasets autonomously. They unveil underlying patterns or groupings within data, devoid of human intervention. These models were employed for three primary tasks: dimensionality reduction, association, and clustering. Some common examples used in unsupervised learning are k-means clustering, association-rule mining, and the Apriori algorithm.

All of these algorithms are essential for revealing relevant relationships in datasets that serve various purposes in industries such as retail and healthcare.

Research Significance: Reducing the time, it takes to resolve tickets is essential to increasing the efficacy and efficiency of customer support systems. While the majority of previous research uses conventional techniques like linear regression, this study investigates machine learning strategies that are less frequently used in this setting, namely k-Nearest Neighbours (KNN) and Support Vector Machines (SVM). The novelty of this study lies in demonstrating the potential of these algorithms to enhance resource allocation and minimize customer wait times. By providing a comparative analysis of KNN and SVM, the research opens up new possibilities for optimizing support systems with advanced machine learning techniques.

Additionally, the development of an interactive prediction interface using Streamlit highlights the practical applications of the findings, offering valuable tools for real-world implementation.

2. Related Works

A predictive approach for understanding event data and forecasting expected closing time was proposed to identify bottlenecks in incident closure, the ML was used, and prior to that, different learning models such as Gradient Boosting Decision Trees, Logistic Regression, and Naive Bayesian classifier were considered and tested with F1 as a performance measurement. Given the F1-score, the gradient boosting decision tree model revealed the best results and was considered for further examination. This model was tested using the real service desk incident record, which means that the method could be applied to problem-solving forecasts for a wide range of issues. Implementation of ITSM also has a positive effect on consumer satisfaction, service expenses, and staff burden [18].

A ML based system that forecast the trouble ticket resolution time was proposed. The advantage of the system is the proposed model can predict the accurate resolution time even when the network problems are inevitable. Several ML techniques, such as boosted regression trees, logistic regression, Artificial Neural Networks (ANNs), and Linear Discriminant Analysis, have been applied. An evaluation based on Mean Absolute Error (MAE) was conducted. Logistic regression and LDA achieved accuracies of 72.7% and 71.3%, respectively. The boosted regression tree model achieved the highest classification accuracy of 74.5%, followed by the ANN with an accuracy of 73.9%. Notably, the ANN attained the lowest MAE of 24.8 hours when predicting the resolution time, demonstrating its efficacy in regression modelling. The ML techniques in this study offer high accuracy and predictive capabilities [16].

A ML-based help desk that embodies additional algorithms to achieve the user's score resolution was proposed. The system allowed the sharing of files and comments concerning an incident, and sending an email in response to each notification from the ticket. Additionally, an ML model was built, which automatically classified the tickets from the very beginning, saved the time of incident resolution, and allowed each issue to achieve resolution and the user to be satisfied. Satisfying other ITSM capabilities, the system also allowed the workflow of IT business processes, the evaluation of KPIs, and the management of users' roles and tickets. The help desk system employs diverse algorithms, such as SVM, Naive Bayes, Rule-based, Decision Tree, J48 (tree-based), Decision Table, and SMO (SVM-based), to categorize tickets and incidents automatically.

This approach managed to reduce the resolution time by considering information such as the ticket title, description, and comments. This approach increased the accuracy of the classification model; that is, the prediction accuracy increased from 53.8% to 81.4%. The ticket comment exchange feature of the system helped validate the accuracy significantly. Nevertheless, the weakness of this system is that more resources are required to generate the ticket generation model. Furthermore, certain knowledge and resources may be needed, which raises the overall cost of the system [8].

A surgical time duration prediction approach which applied ML algorithms to maximize operating room (OR) efficiency and enhance surgery scheduling chores was proposed. A study conducted at a university hospital in Colombia utilized past data to train four ML algorithms: linear regression, SVM, regression trees, and bagged trees. The results indicated that the bagged tree algorithm demonstrated the lowest Root Mean Square Error (RMSE) and computational cost. Consequently, it was identified as the most efficient algorithm for predicting surgical duration. The study also noted a 16% reduction in the error rate of surgical time prediction and a 35% decrease in the minimum programming block when employing the bagged tree algorithm. Additionally, lowering the space of attributes might decrease the impact on computational time, but could likewise lower the discriminant power of the algorithms. Whether information is lost and services decline, some future research studies might include making information recording equipment in the hospital more available to carry around. Other possible research studies might focus on solving the problem of applying categorical attributes in ML and evaluating the impacts of various spaces on the algorithm's behavior. Through these possibilities, it would certainly be easier to enhance the application of ML prediction in terms of surgery length and productivity [19].

On the other hand, [20] suggested using Automated ML (AutoML) to develop data-oriented models for predicting lead times in manufacturing firms. They assessed AutoML solutions (TPOT, H2O.ai, and Auto-Keras) using data from two small and medium enterprises (SMEs). They assessed how their model performed compared to the manual forecast. In addition, they compared AutoML models with linear regression, RF, SVM, and ANN for lead-time prediction. Evaluation metrics such as MAE, RMSE, R^2 score, mean squared log error (MSLE), and training time (in minutes) were used. The results showed that all data-driven models, including the basic mean model, performed better than manual predictions. The TPOT consistently produced the highest-scoring models, followed by H2O.ai and Auto-Keras.

The paper also firmly stated that the benchmarked AutoML systems did not consider the laborious activities of data interpretation, transformation, filtering, pre-processing, and feature engineering. Only a part of the model construction is considered by the systems. Therefore, the authors suggested that future research could be conducted with comprehensive AutoML solutions that actively support users with all the tasks within the modeling process and integrate such solutions more coherently into the diversity of current small and medium-sized enterprise (SME) environments.

A method to determine the complexity of an incident was introduced in 2018. To solve this problem, this study implemented the possibility of ticket triage by identifying tickets associated with the complexity of their resolution, which could allow for additional checks and error-proofing. The experiment attempted to predict the complexity of ticket resolution using a Graph Convolutional Network as one of the features of the incident node. In addition to the high accuracy of the predictions, the model also aimed to evaluate these predictions using a specific loss function, such as the loss of cross-entropy. This experiment attempted to determine the complexity of the actions described above by tracking the number of tickets that required reallocation. Accordingly, this approach depends on the flexibility of ticket resolution [21]. The benefits of using the graph model for ticket resolution in ad hoc inquiry aided the innovation of a Relational Graph Convolutional Network with the aid of ML to forecast ticket reassignments. In addition to accurately predicting ticket reassignment, this model has additional benefits. This helped the embeddings learn the fundamentals of help desk organizations and the roles that their users play. Confidential information was not available in the dataset provided for this study because of privacy concerns. The approach presented in this paper and a roadmap to connect neural network models with a more detailed dataset comprising all ticket actions helps to learn more about ticket resolution.

Additionally, a study that employed supervised ML to forecast the time required to complete IT support tickets was proposed. The authors underscored the significance of data pre-processing and feature selection methods, highlighting their importance alongside the choice of ML algorithms. The ML techniques and algorithms utilized in the research encompassed SVM, Classification and Regression Trees (CART), Multinomial Naive Bayes (MNB), KNN, Linear Regression, and Regression Trees. The evaluation metrics included Mean Squared Error (MSE), Mean Absolute Percentage Error (MAPE), and MAE. The findings revealed that Support Vector Regression (SVR) showed the lowest Mean Average Error (MAE) and MAPE compared to the other algorithms, indicating its superior predictive performance [4].

The authors combined natural language processing techniques with ML techniques to automate the incident response time in an IT ticket management system. They developed automated text categorization ML models by categorizing the incidents. The ML techniques employed are gradient boosting machines, Naive Bayes, SVMs, and maximum entropy. The NLP methods adopted are stop-word removal, lemmatization, 3-gram, and latent Dirichlet allocation with Term Frequency-Inverse Document Frequency. Accuracy and similarity are two of the evaluation metrics used in their study, using a ten-fold cross-validation method. The evaluation findings demonstrate the accuracy of several classifiers and feature-extraction methods [22].

AutoML, an approach to automatically build and evaluate ML models to forecast the factors that resulted in massive delays in a business customer support system was proposed. AutoML automatically builds ML pipelines, including data normalization, model fitting, and parameter tuning. Several ML algorithms, including MLR, ANN, KNN, RF, and SVM, have been implemented. Evaluation metrics such as the Mean Error (ME), RMSE, and MAE were used in the experimental evaluations. The performance was the best in the Gradient Boosting Machine, with the lowest RMSE and MAE values [23]. The advantages of the proposed approach were that it performed better than a typical model selection, a chance to save the costs of manual interference, and the chores of the optimization process. However, attention was limited to one step in that study, and additional research on further applications of AutoML was proposed. Similarly, the study provided little knowledge about the actual implementation, quality, and availability of data-relevant issues regarding how well AutoML models performed [24], [23].

An ML classification system that recognized the subject of the inbound tickets directed by the firm was also implemented. The goals were to improve user happiness and decrease the time required to complete the ticket. Moreover, they delivered the model trained on multilingual ticket classification and urban prototypes for IT analysts to fit it into their day-to-day business. They included 2229 categorized tickets in an open, anonymized dataset released to the public. A fully connected neural network with 500 input neurons and four hidden layers, using the ReLu activation function, along with a 30% dropout, was employed for ticket classification. The output layer consists of seven neurons with a softmax activation function. Furthermore, naive Bayes, decision table, J48, and SMO (SVM-based) algorithms were utilized. Evaluation metrics such as accuracy, precision, and recall achieved an average of 75% for the seven categories within the 2229 support tickets used for training and testing.

In summary, this study achieved a precision of 75% and provided a web prototype for IT analysts to integrate the system into their daily workflow [25].

In addition, [26] proposed employing Graph Neural Networks (GNN) to forecast the remaining cycle time (RCT) in various processes. They suggested that the interdependencies among activities within a process could be depicted as graphs, and that GNNs could effectively utilize this information for prediction. Two distinct GNN models were assessed and compared to the Long Short-Term Memory (LSTM) model using real-world manufacturing company and public datasets. The findings illustrate the superior performance of the GNN model in prediction. The MAE served as the primary evaluation metric to gauge the performance of the models in RCT prediction. This study also underscored the potential of MAPE as an alternative metric for training and assessing prediction models in the future. Figure 1 describes the related works.

References (Year)	ML Techniques								
	SVM	LR	RF	K-NN	Naive Bayes	Linear Discriminant	Neural Networks	Gradient Boosting	Decision Trees
[16] (2018)		X			X			X	X
[14] (2018)		X				X	X	X	
[7] (2021)	X	X			X				X
[17] (2021)	X	X							
[18] (2022)	X	X	X				X		
[3] (2022)	X	X		X	X				
[21] (2022)	X	X	X	X			X		

Figure 1. Comparison of related works

In reviewing related previous studies, it can be seen that a prevalent trend emerged, which is the predominant use of linear regression models for similar task analyses. Although linear regression models have their advantages, they also have certain limitations, particularly when dealing with nonlinear and complex data patterns. This observation led to the decision to explore alternatives. In this context, the selection of k-nearest neighbors (kNN) [11], [27], [28] and Support Vector Machines (SVM) [29], [30] are the primary techniques for several reasons.

1. Versatility and Effectiveness: Both Support Vector Machines and k-nearest neighbors have been widely used and validated across various domains and datasets, demonstrating their versatility and effectiveness. This is in line with studies on[11], [27], [28] kNN and [29], [31] SVM.
2. Handling nonlinearities: SVM and kNN approaches can handle nonlinear problems. This can be advantageous in cases where the data exhibit variations in which the linear models are missing.

This study aims to obtain more accurate and in-depth analysis outcomes using k-nearest neighbors and support vector machines, which could overcome the limitations observed with other models.

3. Methodology

This section discusses the theoretical framework of the proposed approach and how each step is conducted in an elaborated way.

3.1. Theoretical Framework

1) *KNN:* KNN is well known for its simplicity, flexibility, and user-friendliness. It does not require assumptions about data distribution, making it suitable for various datasets. It is commonly used for classification and regression analyses. It can be used for both numerical and categorical data. In addition, the KNN is less affected by outliers. During training, the KNN method employed the entire training dataset as a reference. Next, to make a prediction, a chosen distance metric (e.g., Euclidean distance) is utilized to calculate the distance between each training example and the input data point. The system identifies the K-nearest neighbors of the input data point by assessing the distances.

The KNN offers simplicity and versatility, making it easy to understand and implement for classification and regression tasks without a formal training phase. Its ability to handle nonlinear relationships and robustness to outliers is a notable advantage [32]. However, KNN has disadvantages, including computational complexity, memory requirements, sensitivity to noise and irrelevant features, and the challenge of selecting an optimal k value. It may not perform well in high-dimensional spaces or with imbalanced datasets, and scalability issues may arise with large datasets.

2) *SVM:* Although SVM is applicable for regression and classification assignments, it typically excels in classification scenarios. They gained significant popularity in the 1990s and remain the preferred method for high-performance algorithms with minimal adjustments. SVMs are widely used in various tasks, such as text and image classification, handwriting identification, face detection, spam detection, handwriting identification, and anomaly detection [33]. Their primary objective is to maximize the margin - the distance between the hyperplane and the closest data points from each class in order to identify the optimal hyperplane that divides the classes of the training set.

New data can be categorized according to which side of the hyperplane it falls on once it has been located. SVMs are particularly beneficial for data with many features and distinct separation margins. They can handle both linear and nonlinear relationships through the use of kernel functions, making them versatile for modelling. SVMs are recognized for their robustness, effectiveness in high-dimensional spaces, and ability to generalize well to new, unseen data.

A kernelized SVM provides several benefits. They perform well on various datasets and demonstrate adaptability by allowing the specification of various kernel functions or the definition of bespoke kernels suited to particular data types. They work well with high- and low-dimensional datasets; thus, they can be used in various settings. However, there are significant disadvantages to be considered. Kernelized SVMs efficiency declines with training set size, affecting the operating time and memory consumption. Normalizing the input data and fine-tuning parameters properly can be difficult; however, these are necessary for successful implementation. Additionally, deciphering the rationale behind a certain prediction might be difficult with Kernelized SVMs, because they lack a straightforward probability estimator. Despite these drawbacks, they are invaluable tools for machine learning applications because of their strong performance in various contexts [34].

3.2. Implementation

Figure 2 shows the implementation of the prototype. The prototype consists of three tabs: (1) Model evaluation, (2) Prediction based on KNN approach, and (3) Prediction based on SVM approach. Only one dataset was cleaned and pre-processed in this phase to ensure the output accuracy. The model evaluation tab shows the RMSE and MAE of the model. The prediction tabs display the ticket time resolution based on user input.

Remaining values have been replaced with none. Next, the data type of a few columns consisting of datetime information was converted to a datetime format.

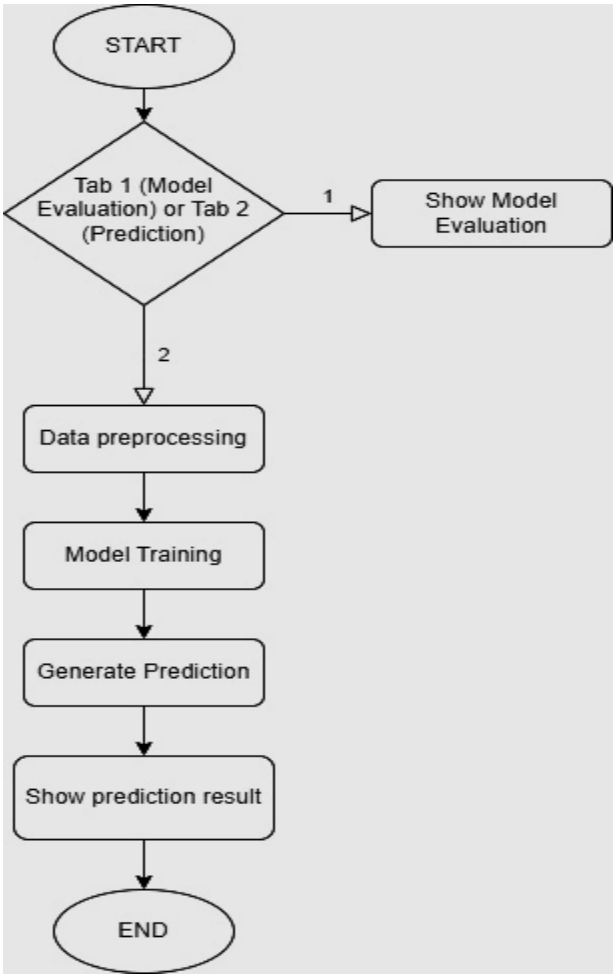

Figure 2. Implementation of prototype flow

The dataset employed is the "Incident Management Process Enriched Event Log" obtained from the UCI ML Repository. This sequential multivariate dataset contains 141,712 rows of data organized into 36 columns. Detailed attribute information is shown in Figure 3. Each attribute plays a role in tracking, analysing and resolving incidents within an organization.

The data cleaning stage starts by filtering only closed-state incidents. Then, the dataset is cleaned by removing columns containing many '?' values.

#	Column Name	Description
	NUMBER	Incident id (24,918 unique values)
1	INCIDENT_STATE	Eight stages to track the changes in the incident manage...
2	ACTIVE	The Boolean attribute which indicates the incident is activ...
3	REASSIGNMENT_COUNT	The reassignment of group or support analyst responsibili...
4	REOPEN_COUNT	The rejection count of incident resolution from the caller
5	SYS_MOD_COUNT	The update count of this incident
6	MADE_SLA	A boolean attribute that signifies whether the event excee...
7	CALLER_ID	Identifier of the impacted user
8	OPENED_BY	Identifier of incident reporter
9	OPENED_AT	Time which incident user open
10	SYS_CREATED_BY	Identifier of incident register
11	SYS_CREATED_AT	Date and time of the incident system creation
12	SYS_UPDATED_BY	Identifier of the user which updates this incident
13	SYS_UPDATED_AT	Time which this incident updates
14	CONTACT_TYPE	Categorical attribute that indicates how this incident is re...
15	LOCATION	Identifier of the location
16	CATEGORY	1st level description of the impacted service
17	SUBCATEGORY	2nd level description of the impacted service
18	U_SYMPTOM	How users describe accessibility of services
19	CMDB_CI	Confirmation item identifier
20	IMPACT	Impact level of incident (values: 1=High; 2=Medium; 3=L...
21	URGENCY	Urgency level of incident (values: 1=High; 2=Medium; 3=...
22	PRIORITY	Priority level of incident (values: 1=High; 2=Medium; 3=L...
23	ASSIGNMENT_GROUP	Identifier of the support group
24	ASSIGNED_TO	Identifier of the person in charge
25	KNOWLEDGE	Boolean attribute that indicates the necessity of involving...
26	U_PRIORITY_CONFIRMA...	Boolean attribute that indicates has the priority field dou...
27	NOTIFY	Categorical attribute that indicates has the notifications w...
28	PROBLEM_ID	Identifier of the problem
29	RFC	(request for change) Identifier of the changing incident re...
30	VENDOR	Identifier of the vendor in charge
31	CAUSED_BY	Identifier of the RFC of this incident
32	CLOSED_CODE	Identifier of the incident resolution
33	RESOLVED_BY	Identifier of the person who resolved the incident
34	RESOLVED_AT	Time which this incident resolved (dependent variable)
35	CLOSED_AT	Time which this incident status changes to closed (depen...

Figure 3. Description of the dataset's attributes

The time required to resolve the ticket was calculated and stored in a new column. Columns containing mixed features are converted to capture only the numerical part. Columns "incident_state" and "active" were removed because all the values in these columns were the same.

Any missing values present in column "resolved_at," "sys_created_by," "sys_created_at," and "u_symptom" are removed. Figure 4 shows a boxplot of the resolution time column. Any outliers in the resolution-time column were removed. Figure 5 shows a correlation heatmap of these features. Highly correlated features were also removed, such as 'sys_mod_count,' 'resolved_by,' and 'made_sla. Next, the dataset is split into test and training sets before continuing with the rest of the data pre-processing to avoid data leakage. Then, one-hot encoding was performed on the categorical data. "impact," "urgency," and "priority" columns were label encoded to leave only scale values, and numerical features were then normalized. The feature selection process had a significant impact on model performance. Removing irrelevant and highly correlated features reduced noise in the data, improving the model's ability to focus on the most influential factors affecting resolution time. For instance, removing redundant features like 'made_sla' reduced the likelihood of overfitting, which would have otherwise compromised the generalization capability of the model.

Moreover, handling outliers and ensuring proper encoding of categorical data contributed to a more robust model that could make better predictions. For example, without handling outliers in the "resolution time" column, the KNN and SVM models would have produced skewed results, as these algorithms are sensitive to extreme values.

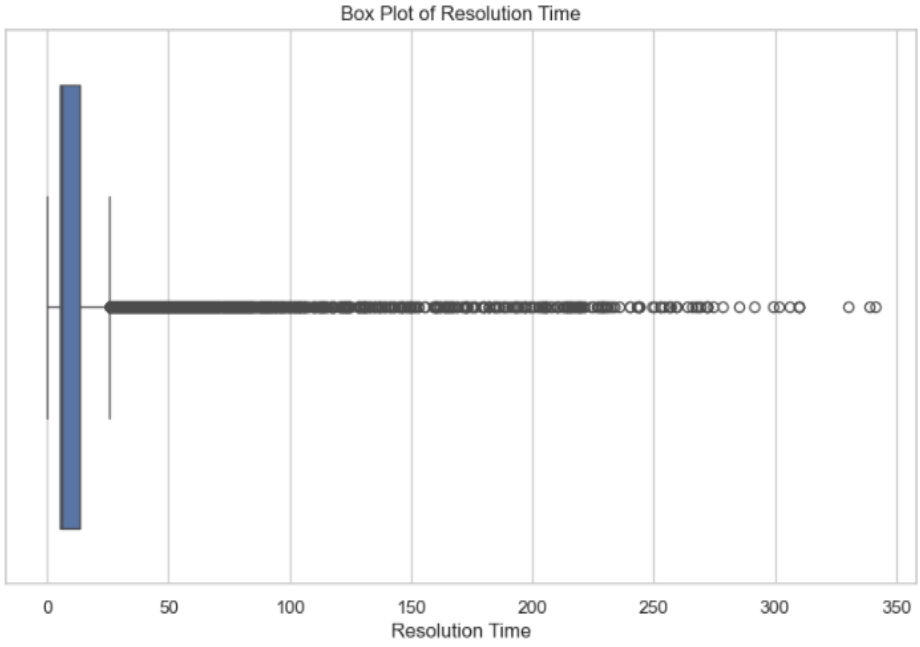

Figure 4. Box plot of resolution time

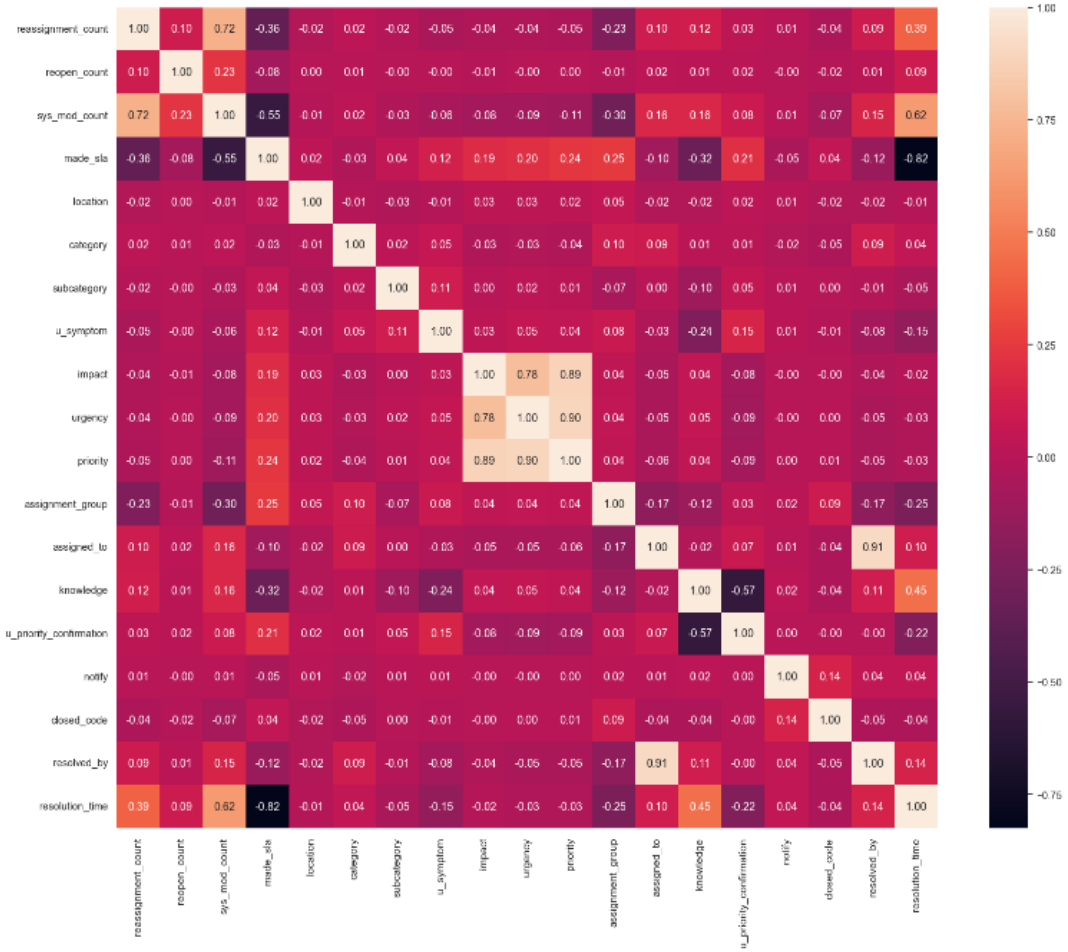

Figure 5. Correlation heatmap

After completing the data pre-processing, the dataset was ready for model training. The Scikit-learn library offers quick and effective tools for performing data prediction in Python. The library also provides built-in features for the train-test split function and the evaluation metrics.

In this phase, 70% of the dataset was allocated to fit the ML model, and 30% for the evaluation. The 70:30 split is a common practice in ML, and many algorithms and techniques are well-suited to this ratio. It achieves this without overly prioritizing one over the other by providing adequate data for training and evaluation. The trained model was prepared to forecast the ticket time resolution using the test data. The prediction is then evaluated based on measures such as the RMSE and MAE.

4. Results and Discussion

The significant result of the proposed approach is shown and discussed in this section.

4.1. Implementation

The GUI was developed using Streamlit. It features three tabs that users can access from the left side of the page. The first tab displays the evaluation metrics for both models. The second tab presents the ticket resolution time prediction based on the KNN model using user input, whereas the third tab shows the prediction based on the SVM model. Figure 6 illustrates the graphical user interface in the first tab.

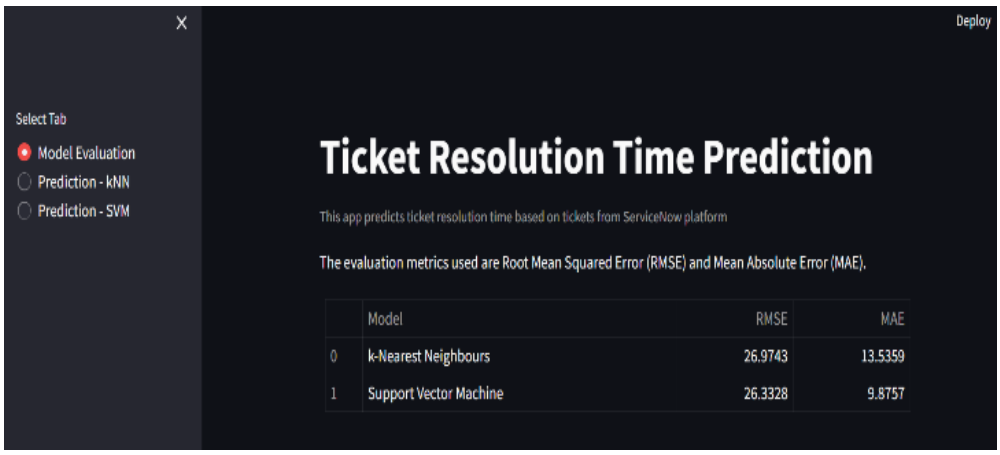

Figure 6. Model evaluation tab

The second tab shows the resolution time prediction of a ticket based on user input. The prediction in the second tab was based on the kNN model.

Figure 7 shows the graphical user interface for the second tab.

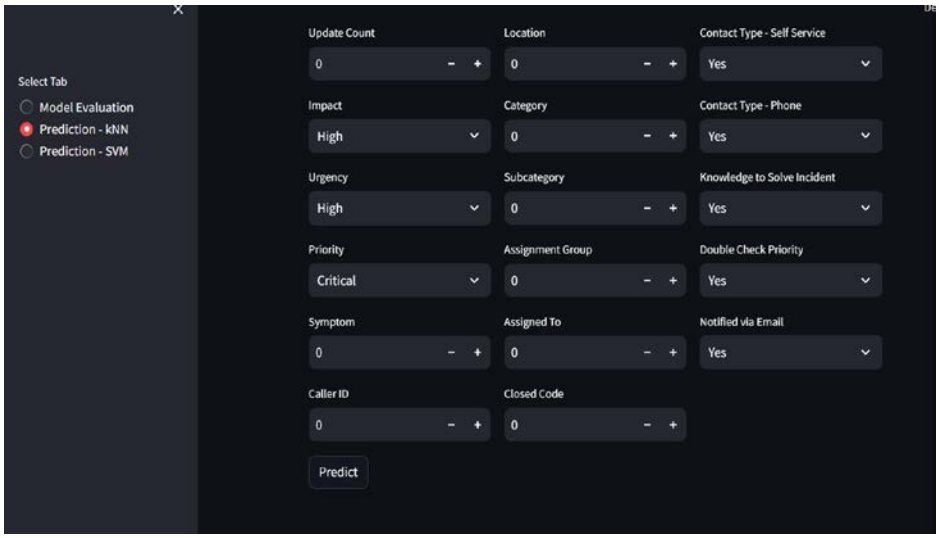

Figure 7. Prediction- kNN tab

The third tab shows the resolution time prediction of a ticket based on the user input.

The prediction in the third tab was based on the SVM model. Figure 8 shows the graphical user interface for the third tab.

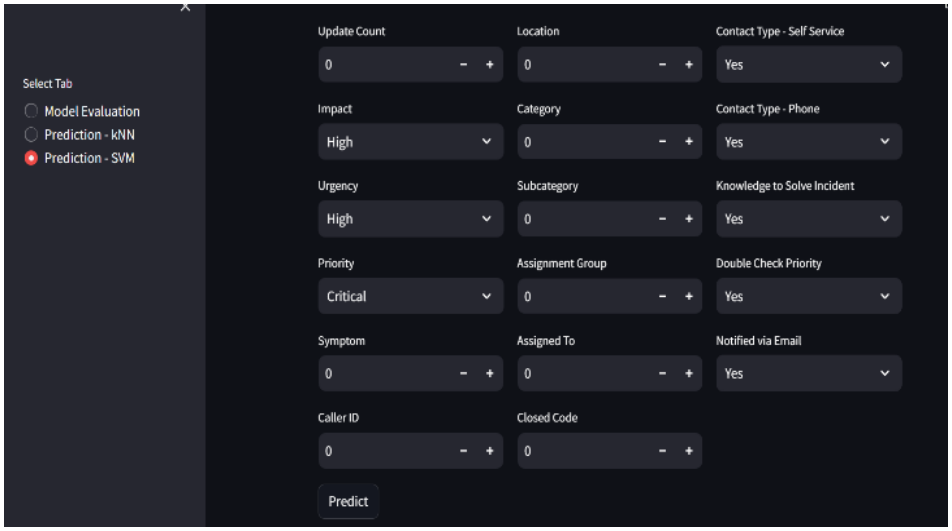

Figure 8. Prediction- SVM tab

4.2. Pseudocode

The pseudocode in Table 1 (Algorithm 1) outlines the application for predicting the ticket resolution time using the two ML models. It contains three main functions: Predicting the resolution time using the KNN model, predicting the resolution time using the SVM model, and the main function displaying the user interface.

Table 1. Algorithm 1: ticket resolution time prediction

Algorithm 1 Ticket Resolution Time Prediction

```
function predict resolution time (input data)
    LOAD scaler
    CREATE input_df from input data
    numerical features = ['reassignment count','reopen count','sys mod count']
    SCALE input_df's numerical features using the loaded scaler
    LOAD knn_regressor
    predictions = PREDICT input_df using knn regressor
    return predictions
end function
function predict resolution time2(input data)
    LOAD
    CREATE input_df from input data
    numerical features = ['reassignment count','reopen count','sys mod count']
    SCALE input_df's numerical features using the loaded scaler
    LOAD svm_regressor
    predictions1 = PREDICT input_df using svm_regressor
    return predictions1
end function
function main
    DISPLAY title "Ticket Resolution Time Prediction"
    selected tab = GET user's selected tab from sidebar with options "Model Evaluation", "Prediction − kNN", "Prediction − SVM"
    if selected tab is "Prediction - kNN" then
        DISPLAY input fields for user input
        if user clicks "Predict" button then
            CONVERT input data to numerical values using mappings
            PERFORM prediction using predict resolution time and DISPLAY the predicted resolution time
        end if
    else if selected tab is "Prediction - SVM" then
        DISPLAY input fields for user input
        if user clicks "Predict" button then
            CONVERT input data to numerical values using mappings
            PERFORM prediction using predict resolution time2 and DISPLAY the predicted resolution time
        end if
    else if selected tab is "Model Evaluation" then
        DISPLAY "The evaluation metrics used are (RMSE) (MAE)."
        LOAD metrics for Model 1 (KNN)
        LOAD metrics for Model 2 (SVM)
        COMBINE metrics for both models into a single DataFrame
        DISPLAY the combined metrics table
    end if
end function
```

4.3. Evaluation Metrics

The evaluation metrics employed in the evaluations were the RMSE, MSE, MAE, and Coefficient of Determination, R^2 Score. RMSE, MSE, and MAE are common evaluation metrics used to check for accuracy in a recommender system. The RMSE measures the square root of the average squared differences between the predicted and actual ratings, the MAE measures the average absolute differences, and the MSE measures the average squared differences. Generally, RMSE is a better evaluation metric in most cases because it is sensitive to large errors. The lower the value, the lower the error.

The R^2 score ranged from 0 to 1, with 1 indicating perfect prediction and 0 indicating that the model did not explain any variance in the target variable. The R^2 score can also be negative if the model performs worse than a horizontal line representing the mean of the actual values.

4.4. Evaluation Results and Discussions

The evaluation score for KNN and SVM approaches are recorded in Table 2.

Based on these results, the best model with the highest R^2 score was the KNN model, followed by the SVM model. Specifically:

- KNN Model: Achieved an R^2 score of -0.08, an RMSE of 26.33, an MSE of 693.41, and an MAE of 9.87.
- SVM Model: Achieved an R^2 score of -0.13, an RMSE of 26.97, an MSE of 727.61, and an MAE of 13.53.

The KNN model demonstrates the best performance in the R^2 score, indicating that it explains the variance in the data better than the SVM. In addition, KNN has lower MSE, MAE, and RMSE, suggesting more accurate predictions, as it makes smaller errors on average compared to the other models. There may be factors why the KNN model outperformed the SVM model. Firstly, KNN relies on finding the closest neighbors in feature space, which has been advantageous for this dataset, especially since there are well-defined clusters or groups within the data. Next, SVM is highly sensitive to the selection of hyperparameters like the kernel type and regularization term. While SVM generally performs well for high-dimensional data or when there is a clear boundary separating classes, in regression tasks, its performance may deteriorate if the dataset is noisy or non-linearly separable.

Table 2. Evaluation results

ML Approach	MSE	RMSE	MAE	R^2
SVM	727.61	26.97	13.54	-0.14
KNN	693.41	26.33	9.88	-0.08

Furthermore, to analyze the results based on the dataset utilized:

- The Incident Management Process dataset is sequential. Thus, the dataset's row or event may carry crucial information. From this observation, KNN naturally handle the dataset well since it operates based on proximity in feature space and the implementation is straightforward. If neighboring rows in this dataset share similar outcomes or processes, KNN can pick up on these trends [35], [36].
- Since KNN can be scaled and normalized, KNN can balance the contributions of different features to the proximity calculation when appropriate scaling and feature engineering are applied. Contrast to this, SVM struggles with large feature sets unless the features are highly informative [37], [38].
- As a non-parametric technique, KNN can better accommodate noise by averaging among neighboring points, lessening the effect of outliners. Since it may allow for class distribution by varying the weight of neighbors (e.g., distance-weighted KNN), it is also more resilient to class imbalance. On the other hand, the support vectors, which are susceptible to outliers or noisy data points, significantly impact the decision border of SVM. SVM's performance may suffer if the dataset contains outliers or incorrect classifications because of improper boundary placements [39], [40].

5. Conclusion

With an emphasis on automating the ticket assignment and forecasting resolution times, this paper reviewed some existing ML techniques to optimize the Ticket Support System. By applying KNN and SVM to the problem of ticket resolution time prediction, it explores less commonly used techniques compared to prevalent linear regression models. This study examined the development and application of these machine learning models, considering best practices and optimization through hyperparameter tuning.

The limitations to be considered of this study include data availability, model limitations and evaluation metrics. Firstly, since the data only relied on historical data and from only one platform, the model's predictions on future trends that are new may not be good enough.

Also, both KNN and SVM models have their own limitations which contribute to the models' predictions results. While metrics such as MSE, RMSE, MAE, and R^2 were used to evaluate model performance, other important metrics like MAPE (Mean Absolute Percentage Error) or F1 score could offer more comprehensive insights, especially in understanding model performance for different types of errors.

Future work will include implementing additional ML techniques for evaluation with other datasets. To improve prediction accuracy, future work will also involve the exploration of additional machine learning models, including but not limited to ensemble methods, AutoML, and Advanced Neural Networks. In addition, an enhanced GUI with additional analysis features can be implemented.

References:

[1]. İşcen, E. S., & Gürbüz, M. Z. (2019). A comparison of text classifiers on it incidents using weka. *2019 4th International Conference on Computer Science and Engineering (UBMK)*, 510-515. Doi: 10.1109/UBMK.2019.8907063

[2]. Gupta, M., Asadullah, A., Padmanabhuni, S., & Serebrenik, A. (2018). Reducing user input requests to improve IT support ticket resolution process. *Empirical Software Engineering*, *23*(3), 1664-1703. Doi: 10.1007/s10664-017-9532-2

[3]. Taha, A., et al. (2025). AI Web-Based Smart Ticketing System. *Advanced Sciences and Technology Journal*, *2*(1), 1-22. Doi: 10.21608/astj.2025.345106.1027

[4]. Yıldız, M., et al. (2022). *IT support ticket completion time prediction*. Proceedings of the 7th International Conference on Computer Science and Engineering (UBMK 2022), 198–203. Doi: 10.1109/UBMK55850.2022.9919591

[5]. Qamili, R., Shabani, S., & Schneider, J. (2018). An intelligent framework for issue ticketing system based on machine learning. *2018 IEEE 22nd international enterprise distributed object computing workshop (EDOCW)* 79-86. Doi: 10.1109/EDOCW.2018.00022

[6]. Montgomery, L., et al. (2018). Customer support ticket escalation prediction using feature engineering. *Requirements Engineering*, *23*(3), 333-355. Doi: 10.1007/s00766-018-0292-3

[7]. Nayebi, M., Dicke, L., Ittyipe, R., Carlson, C., & Ruhe, G. (2018). Essmart way to manage user requests. *arXiv preprint arXiv:1808.03796*. Doi: 10.48550/arXiv.1808.03796.

[8]. Al-Hawari, F., & Barham, H. (2021). A machine learning based help desk system for IT service management. *Journal of king saud university-computer and information sciences*, *33*(6), 702-718. Doi: 10.1016/j.jksuci.2019.04.001

[9]. Mandal, A., et al. (2019). Improving it support by enhancing incident management process with multi-modal analysis. *International Conference on Service-Oriented Computing*, 431-446. Springer International Publishing.

[10]. Konda, R. (2025). AI-driven customer support: Transforming user experience and operational efficiency. *IJSAT-International Journal on Science and Technology*, *16*(1). Doi: 10.71097/IJSAT.v16.i1.2600

[11]. Shanardi Wijaya, A., & Oktavia, T. (2024). Machine learning approaches for helpdesk ticketing system: A systematic literature review. *Journal of Theoretical and Applied Information Technology*, *102*(5).

[12]. Haw, S. C., et al. (2022). Improving the prediction resolution time for customer support ticket system. *Journal of System and Management Sciences*, *12*(6), 1-16. Doi: 10.33168/JSMS.2022.0601

[13]. Kannan, R., et al. (2023). Real-time prediction of free lime in cement clinker using support vector machine algorithm. *2023 4th International Conference on Big Data Analytics and Practices (IBDAP)*, 1-4. Doi: 10.1109/IBDAP58581.2023.10271990

[14]. Yujiao, Z., et al. (2023). Dropout prediction model for college students in moocs based on weighted multi-feature and svm. *Journal of Informatics and Web Engineering*, *2*(2), 29-42.

[15]. Kent, K. D., et al. (2023). Ticketing system: A descriptive research on the use of ticketing system for project management and issue tracking in IT companies. *International Journal of Computing Sciences Research*, *7*, 1066–1075. Doi: 10.25147/ijcsr.2017.001.1.90

[16]. Sample, K. R., et al. (2022). Predicting trouble ticket resolution. *Proceedings of the AAAI Conference*,

[17]. Meena, M. J., & Chandran, K. R. (2009). Naïve Bayes text classification with positive features selected by statistical method. *2009 First International Conference on Advanced Computing*, 28-33. Doi: 10.1109/ICADVC.2009.5378273

[18]. Zuev, D., Kalistratov, A., & Zuev, A. (2018). Machine learning in IT service management. *Procedia computer science*, *145*, 675-679. Doi: 10.1016/j.procs.2018.11.063

[19]. Martinez, O., et al. (2021). Machine learning for surgical time prediction. *Computer Methods and Programs in Biomedicine*, *208*, 106220. Doi: 10.1016/j.cmpb.2021.106220

[20]. Bender, J., Trat, M., & Ovtcharova, J. (2022). Benchmarking AutoML-supported lead time prediction. *Procedia Computer Science*, *200*, 482-494. Doi: 10.1016/j.procs.2022.01.246

[21]. Schad, J., Sambasivan, R., & Woodward, C. (2022). Predicting help desk ticket reassignments with graph convolutional networks. *Machine Learning with Applications*, *7*, 100237. Doi: 10.1016/j.mlwa.2021.100237

[22]. Subbarao, M. V., et al. (2022). Automation of incident response and IT ticket management by ML and NLP mechanisms. *Journal of Theoretical and Applied Information Technology*, *100*(12), 3945-3955.

[23]. Gerunov, A. A. (2022). Forecasting customer support resolution times through automated machine learning. *Economics and Management, 19*(2), 1–11. Doi: 10.37708/em.swu.v19i2.1

[24]. Amaral, C. A., et al. (2018). Enhancing completion time prediction through attribute selection. *Conference on Advanced Information Technologies for Management*, 3-23. Doi: 10.1007/978-3-030-15154-61

[25]. Pereira, L. S. B., et al. (2023). Machine learning for classification of it support tickets. In *2023 International Conference On Cyber Management And Engineering (CyMaEn)*, 210-213. Doi: 10.1109/CyMaEn57228.2023.10051041

[26]. Duong, L. T., et al. (2023). Remaining cycle time prediction with graph neural networks for predictive process monitoring. *Proceedings of the 2023 8th international conference on machine learning technologies*, 95-101. Doi: 10.1145/3589883.3589897

[27]. Tai, T. E., et al. (2023). Performance evaluation on resolution time prediction using decision tree, random forest and extreme gradient boosting. *2023 International Conference on Computer Applications Technology (CCAT)*, 74-79. Doi: 10.1109/CCAT59108.2023.00021

[28]. Khan, T. A., et al. (2024). Sentiment analysis using support vector machine and random forest. *Journal of Informatics and Web Engineering*, *3*(1), 67-75.

[29]. Liao, W., Cade, & Behdad, S. (2022). Forecasting repair and maintenance services of medical devices using support vector machine. *Journal of Manufacturing Science and Engineering, 144*(3). Doi: 10.1115/1.4051886

[30]. Harun, N. A., Huspi, S. H., & Iahad, N. A. (2023). Question classification for helpdesk support forum using support vector machine and Naïve Bayes algorithm. *International Journal of Innovative Computing*, *13*(1), 37-45. Doi: 10.11113/ijic.v13n1.388

[31]. Zhou, W., et al. (2016). Resolution recommendation for event tickets in service management. *IEEE Transactions on Network and Service Management*, *13*(4), 954-967. Doi: 10.1109/TNSM.2016.2587807

[32]. Bahrani, P., et al. (2024). A new improved KNN-based recommender system. *Journal of Supercomputing*, *80*(1). Doi: 10.1007/s11227-023-05447-1

[33]. Lim, S. T., et al. (2023). Predicting travel insurance purchases in an insurance firm through machine learning methods after COVID-19. *Journal of Informatics and Web Engineering, 2*(2). Doi: 10.33093/jiwe.2023.2.2.4

[34]. bin Harunasir, M. F., et al. (2023). Sentiment analysis of amazon product reviews by supervised machine learning models. *Journal of Advances in Information Technology*, *14*(4), 857-862. Doi: 10.12720/jait.14.4.857-862.

[35]. Xing, Z., Pei, J., & Keogh, E. (2010). A brief survey on sequence classification. *ACM Sigkdd Explorations Newsletter*, *12*(1), 40-48. Doi: 10.1145/1882471.1882478

[36]. Bao, Z., et al. (2023). Sequence Classification Prediction Based on SVM-KNN. *2023 IEEE International Conference on Sensors, Electronics and Computer Engineering (ICSECE)*, 1472-1476. Doi: 10.1109/ICSECE58870.2023.10263416

[37]. Lu, Z. Q. J. (2010). The elements of statistical learning: Data mining, inference, and prediction. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, *173*(3). Doi: 10.1111/j.1467-985x.2010.00646_6.x

[38]. Hastie, T., et al. (2005). The elements of statistical learning: data mining, inference and prediction. *The Mathematical Intelligencer*, *27*(2), 83-85.

[39]. Palli, A. S., et al. (2024). Online machine learning from non-stationary data streams in the presence of concept drift and class imbalance: A systematic review. *Journal of Information and Communication Technology*, *23*(1), 105-139. Doi: 10.32890/jict2024.23.1.5.

[40]. Japkowicz, N., & Stephen, S. (2002). The class imbalance problem: A systematic study. *Intelligent data analysis*, *6*(5), 429-449. Doi: 10.3233/ida-2002-6504