Symptom-Based Disease Prediction Using Machine Learning Algorithms

Carlxen Brieyl P. Duran
College of Computing and Information Technologies
National University
Manila, Philippines
durancp@students.national-u.edu.ph

Mark Rid B. Ramirez
College of Computing and Information Technologies
National University
Manila, Philippines
ramirezmb@students.national-u.edu.ph

***Abstract*—Machine learning (ML) has become an important component in healthcare, improving how diseases are diagnosed through data-driven analysis of patient symptoms. This study evaluates and compares the performance of four algorithms: K-Nearest Neighbors (KNN), Logistic Regression (LR), Random Forest (RF), and XGBoost (XGB) for predicting diseases based on symptom data. The dataset used was obtained from Kaggle, it includes 132 symptom-related features and 42 disease categories. After data cleaning, pre-processing and feature engineering, each model was trained and tested using accuracy, precision, recall, and F1-score as performance metrics. Cross-validation and hyperparameter tuning were performed to enhance model performance. Among the models, Random Forest achieved the highest accuracy and F1-score, showing its strong capability to handle complex, symptom-based data. The findings indicate that machine learning techniques can effectively support prediction and have significant potential to assist healthcare professionals in early diagnosis and clinical decision making.**

***Index Terms*—Machine Learning, Disease Prediction, K-Nearest Neighbors, Logistic Regression, Random Forest, XG-Boost**

I. Introduction

Machine learning (ML) continues to revolutionize healthcare by enabling intelligent systems that support physicians in diagnosing diseases accurately and efficiently. Traditional diagnostic processes rely heavily on physician experience and manual analysis of patient symptoms, which can be subjective and time-consuming. With the increasing availability of structured medical datasets, ML algorithms can learn underlying relationships between symptoms and diseases, improving diagnostic accuracy and reducing human bias [1], [2]. Recent advancements in ML techniques such as ensemble learning, distance-based classifiers, and optimization-based algorithms have significantly enhanced predictive modeling in medicine. However, the performance of these algorithms often depends on factors such as dataset quality, preprocessing, and parameter tuning [3], [4].

This study aims to compare the performance of four machine learning algorithms K-Nearest Neighbors (KNN), Logistic Regression, Random Forest, and XGBoost using a symptom-based dataset that maps 132 symptoms to 42 possible diseases [4]. The study evaluates each model using accuracy, precision, recall, and F1 score, with additional validation through 5-fold cross-validation and GridSearchCV-based hyperparameter tuning. The primary goal is to identify the model that achieves the most balanced and reliable performance for symptom-based disease prediction.

II. Literature Review

Medical Practitioners in early detection and prognosis has been data-driven in disease diagnosis, therefore, Machine Learning (ML) has become significant in the medical field. Various studies [1], [2], [3], [5] have investigated different ML algorithms to enhance the accuracy of disease prediction based on patient symptoms. This section reviews existing works that have explored the algorithms K-Nearest Neighbors (KNN), Logistic Regression, Random Forest, and XGBoost, which are employed in this study.

A. Role of Machine Learning in Disease Diagnosis

The study presented in [3] emphasizes the fundamental role of machine learning algorithms in medical diagnostics, identifying approaches such as KNN, K-Means, Support Vector Machine (SVM), Naïve Bayes, Decision Tree, Random Forest, Logistic Regression, and Deep Learning. The authors conclude that algorithm performance depends heavily on dataset properties, feature selection, and preprocessing techniques. Furthermore, no single algorithm consistently outperforms others across different disease types, highlighting the importance of comparative analysis. This review establishes the groundwork for evaluating multiple algorithms for symptom-based disease prediction.

1) K-Nearest Neighbors (KNN): The KNN algorithm, discussed in [3], is a nonparametric instance-based learning model that predicts results based on the majority vote of the nearest data points. In disease diagnosis, KNN has shown reliability when symptom data is well-normalized and moderate in size. Its primary strengths include interpretability and minimal training time, while its limitations involve sensitivity to noisy data and computational inefficiency in large datasets. And within the concept of symptom-based disease prediction, KNN is an effective baseline model due to its ability to detect local pattern similarities in patient data [3].

2) Logistic Regression: Logistic Regression has been known for diagnosing diseases because of how the algorithm has solid statistical foundations. Research [6] has shown that it can work just as well as complicated machine learning approaches when comes to predicting health conditions. This suggests that a straightforward model does the same work, especially when the connections between patient factors and health outcomes follow a relatively linear pattern. The study also highlights Logistic Regression's suitability for smaller datasets and its ability to produce clinically interpretable coefficients. Therefore, Logistic Regression remains an essential algorithm for explainable symptom-based prediction models.

3) Random Forest: Random Forest is a robust ensemble method that aggregates multiple decision trees to improve predictive accuracy and reduce overfitting. According to [3] and [4], Random Forest has demonstrated excellent performance in various health-monitoring and disease-prediction applications. In the predictive health monitoring system proposed by Sony et al. [4], Random Forest effectively handled patient risk assessment tasks alongside other algorithms such as SVM and XGBoost. Its advantages include handling noisy features, generating feature importance metrics, and maintaining high accuracy even with heterogeneous medical data. However, its complexity may limit interpretability compared with linear models.

4) XGBoost Algorithm: XGBoost (Extreme Gradient Boosting) is a high-performance ensemble algorithm known for its scalability and superior classification accuracy in structured data problems. In [7], the authors applied XGBoost to multiple disease prediction tasks and achieved significant accuracy improvements compared to traditional ML models. The study in [4] further supports the algorithm's efficiency in predictive health monitoring systems, where it outperformed other classifiers in handling large and complex datasets. XGBoost's advantages lie in its gradient-boosting framework, which efficiently captures nonlinear interactions and mitigates overfitting through regularization. These findings underscore XGBoost's relevance to symptom-based disease prediction tasks.

III. Methodology

A. Data Collection

The data set was obtained from the Kaushil268's disease prediction using machine learning collection on Kaggle [2]. It has two CSV files: Training.csv and Testing.csv, that has a total of 133 columns, 132 binary symptom indicators, and a target column called prognosis, representing 42 disease classes. Each data represents a simulated mapping of symptoms and diseases generated for academic use. The training csv dataset was used to develop and tune the models, while the testing dataset was used for performance validation.

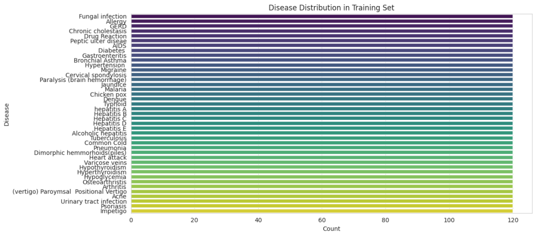

Fig. 1. Disease distribution in the training set.

Figure 1 displays the distribution of the 42 disease classes in the training csv dataset after preprocessing. The raw data contained a lot of duplicates, null values, and inconsistent columns, which were removed or corrected to ensure clean input for model training. As shown in the figure, the dataset shows varying frequencies among disease classes, with some diseases being more prevalent than others, indicating a clear sign of class imbalance. This information guided the stratified splitting of data into training and validation sets, ensuring that each subset maintained proportional representation of all disease classes, which is critical for fair and accurate model evaluation.

B. Data Pre-Processing

To ensure the quality, consistency, and reliability of the symptom-based dataset, several data cleaning and feature engineering techniques were used before training the models. These steps were made to remove redundancies, maintain feature consistency, and prepare the dataset for accurate disease classification. The complete pre-processing steps is outlined as follows:

1) Initial Inspection: The raw datasets (Training.csv and Testing.csv) were checked to assess their structure and integrity. The training dataset initially contained 4920 records and 133 columns, while the testing dataset contained 41 records. An initial check was performed to identify null, unnamed, or duplicate values that could negatively impact and confuse the models affecting their performance.

2) Removal of Unnamed Columns: Certain columns were labeled as "Unnamed: 133", which were automatically generated during CSV export. These columns had no relevant information and were removed using pattern-based filtering with the pandas library. This step improved readability and ensured only important and necessary features remained.

3) Duplicate Elimination: Redundant entries within the training data were detected using the DataFrame.duplicated() function. Duplicate records were dropped to avoid bias during training and to maintain unique symptom disease mappings. This helps preserve data integrity and prevent overfitting.

4) Feature Alignment Between Training and Testing Sets: The feature columns of the training and testing datasets were compared to ensure uniformity. Any missing features in the test set were added and filled with zeros to maintain consistent dimensionality. The columns were then reordered to exactly match the training feature arrangement, ensuring compatibility during model evaluation.

5) Removal of Overlapping Samples: Checked for possible overlaps between train and test sets to ensure clean evaluation boundaries. Each record was assigned a unique hash using `pd.util.hash_pandas_object()`. Rows with identical hashes appearing in both datasets were excluded from training. This step ensured a clear separation between training and testing samples for a fair and proper evaluation.

6) Target Label Encoding: The prognosis column, representing the disease label, was converted from categorical text format to numerical integers using Label Encoding. This transformation enabled the models to interpret the target variable effectively during classification.

7) Data Splitting: Following preprocessing, the dataset was divided into training and validation subsets using `train_test_split()` with an 80:20 ratio and stratified sampling to preserve class distribution. This internal split allowed consistent model evaluation and fine tuning while maintaining representative proportions of each disease category.

8) Final Dataset Overview: After all preprocessing steps, the datasets were free from duplicates and overlaps, aligned in structure, and encoded for model consumption. The resulting data ensured reliability, consistency, and readiness for machine learning experimentation and comparative analysis.

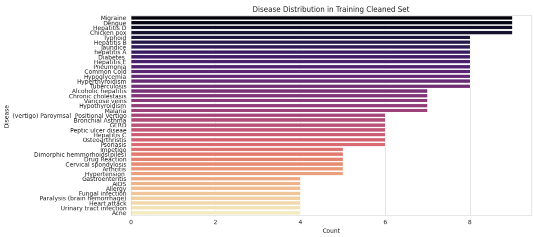

Fig. 2. Disease distribution in the training set after data cleaning and preprocessing.

The cleaned dataset, as illustrated in Figure 2, shows the distribution of disease classes after preprocessing. Removing duplicates, unnamed columns, and overlapping records ensured that each disease was accurately represented without redundancy. The figure confirms that the data set maintains class balance for most diseases, which is necessary to train machine learning models effectively. Sparse disease classes are still present but minimized, highlighting the readiness of the data set for reliable classification and evaluation.

C. Experimental Setup

The experimental framework utilized a symptom–disease dataset obtained from Kaushil268's "Disease Prediction Using Machine Learning" dataset on Kaggle. The dataset comprises 132 binary symptom attributes mapped to 42 disease labels, with 4,920 training samples and 41 testing samples after data cleaning and preprocessing.

All experiments were conducted in Python using the scikit-learn and XGBoost libraries. The environment included pandas and NumPy for data manipulation, while GridSearchCV handled hyperparameter optimization. Random seeds were fixed to ensure reproducibility. The dataset was divided into training (80%) and validation (20%) sets using stratified sampling to preserve class distribution. The testing dataset remained untouched throughout training to provide an unbiased evaluation of model generalization.

D. Algorithms

K-Nearest Neighbors (KNN)

K-Nearest Neighbors is a non-parametric, instance-based supervised learning technique that operates on a straightforward principle[8]. When presented with a new sample, the algorithm identifies the k closest training examples in feature space using Euclidean distance, then assigns the class label that appears most frequently among those neighbors. For regression, it averages (or otherwise aggregates) the target values of those neighbors. KNN has no explicit training phase and relies heavily on the structure of the data and distance measures. Because it used the entire dataset as the "model".

$$d(x,y)=\sqrt{\sum_{i=1}^{n}(x_i-y_i)^2}$$

Equation 1: K-Nearest Neighbors

Equation 1 represents the Euclidean distance, which quantifies how similar or different two feature vectors are. The term $(x_i-y_i)^2$ measures the squared difference between each feature of two samples, and the summation $\sum_{i=1}^{n}$ aggregates these across all n features. Taking the square root normalizes the measure to the same unit as the original features. Smaller $d(x,y)$ indicates higher similarity, making those samples more influential in KNN's classification or regression decision [8], [9].

Logistic Regression

Logistic Regression is a parametric supervised learning model designed for binary and multinomial classification tasks [10]. It operates by modeling the log-odds of the positive class as a linear combination of input features, then applies the logistic sigmoid function to transform this value into a probability between 0 and 1. Thus:

$$P(y=1\mid x)=\frac{1}{1+exp-(\beta_0+\beta_1x_1+\cdots+\beta_nx_n)}$$

Equation 2: Logistic Regression

In Equation 2, β_0 represents the intercept, while $\beta_1,\ldots,\beta_n$ are coefficients that weight each feature $x_1,\ldots,x_n$. The exponent term $\exp-(\beta_0+\beta_1x_1+\cdots+\beta_nx_n)$ transforms the linear combination into a nonlinear relationship. The denominator $1+\exp-(z)$ ensures the output probability $P(y=1\mid x)$ lies between 0 and 1. When $P>0.5$, the sample is typically classified as the positive class [11], [10].

Random Forest

Random Forest is an orchestra-based supervised learning algorithm by ensembling multiple decision trees to improve predictive performance and reduce overfitting [12]. Each tree in its 'forest' is trained on a sample of the dataset, and at each split, a random subset of features is examined. For classification tasks, the final prediction is determined through majority voting among the trees, whereas for regression tasks, the output represents the average of all tree predictions.

$$\hat{y} = \frac{1}{T} \sum_{t=1}^{T} f_t(x)$$

Equation 3: Random Forest

Equation 3 defines the ensemble prediction process. Each $f_t(x)$ represents the prediction from the t-th decision tree, and T is the total number of trees in the forest. The average $\frac{1}{T} \sum_{t=1}^{T} f_t(x)$ combines multiple weak lear ners to form a strong predictor. For classification, a majority vote is taken; for regression, the mean of predictions is used. This aggregation reduces variance and improves stability across datasets [5].

XGBoost Algorithm

Extreme Gradient Boosting (XGBoost) is an optimized implementation of gradient boosting designed for speed and performance. It builds additive models in a forward stage-wise fashion, allowing optimization of an arbitrary differentiable loss function. Each new tree minimizes the loss by correcting the residuals of previous predictions through gradient descent.

$$Obj = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t)$$

Equation 4: XGBoost A

Equation 4 describes the optimization objective of XGBoost. The first term $\sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i))$ is the loss function, which measures the difference between actual y_i and predicted $\hat{y}_i$. The second term $\Omega(f_t)$ is a regularization function that penalizes model complexity, controlling overfitting. Each new tree $f_t(x)$ is added to minimize the total objective by correcting previous residual errors through gradient descent [13].

E. Training Procedure

The preprocessing pipeline included removal of duplicates, handling of missing values, label encoding for the target variable, and standardization for algorithms sensitive to feature scale (e.g., KNN, Logistic Regression). Tree-based models such as Random Forest and XGBoost were trained directly on the binary-encoded data. Each algorithm, Logistic Regression, KNN, Random Forest, and XGBoost, was initially trained using default parameters to establish baseline performance. Subsequently, `GridSearchCV` with five-fold cross-validation was employed to determine the optimal hyperparameters for each model. For instance, Logistic Regression parameters included penalty type (`l1`, `l2`), regularization strength (`C`), and solver selection (`liblinear`, `saga`). Random Forest optimization explored tree depth, number of estimators, and minimum samples per split, while XGBoost tuning focused on learning rate, tree depth, and regularization terms. Once optimal parameters were obtained, each model was retrained on the entire training set and evaluated on the independent test set to compute final performance metrics.

F. Evaluation Metric

The performance of each model was evaluated using standard classification metrics, namely accuracy, precision, recall, and F1-score. Accuracy measures the overall proportion of correctly predicted instances in the dataset, while precision reflects the reliability of positive predictions across multiple disease classes. Recall, on the other hand, quantifies the model's ability to identify all relevant cases, ensuring that diseases present in the data are correctly detected. The F1-score, defined as the harmonic mean of precision and recall, serves as the primary metric in this study because it provides a balanced evaluation of performance in the presence of class imbalance. In addition to these metrics, a confusion matrix was generated for each model to assess misclassifications among specific disease categories. During hyperparameter tuning, cross-validation mean and standard deviation of the F1-score were recorded to analyze the consistency and stability of each algorithm across multiple folds.

IV. Results and Discussion

A. Baseline Comparative Models

Table I presents the performance metrics for all base models before hyperparameter tuning. Among the four models tested, Random Forest and K-Nearest Neighbors (KNN) initially achieved the highest base accuracies (97.62%), closely followed by Logistic Regression and XGBoost with similar performance.

TABLE I
Classification Report for Base Models

Model	**Accuracy**	**Precision**	**Recall**	**F1-Score**
Logistic Regression	0.9762	0.9881	0.9762	0.9762
Random Forest	0.9762	0.9878	0.9878	0.9837
KNN	0.9762	0.9878	0.9878	0.9837
XGBoost	0.9762	0.9800	0.9700	0.9700

All models demonstrate high precision, recall, and F1-scores above 0.97, suggesting that the dataset is clean, well-structured, and discriminative for disease prediction. Moreover, the minimal difference between metrics indicates that each model consistently identifies disease classes with low misclassification rates. However, Random Forest and KNN models displayed slightly more stable cross-validation results compared to XGBoost, which exhibited greater variance across folds. This implies that ensemble-based and distance-based methods adapt better to the dataset's feature space than gradient boosting under its default configuration. Overall, all baseline models can be considered highly competent, with Random Forest slightly outperforming others in terms of cross-validation stability. Logistic Regression remains valuable for its interpretability and computational efficiency, making it ideal for deployment in lightweight applications.

B. Hyperparameter Tuning of Models

Each model was parameter-optimized through `GridSearchCV`. The search space was designed to test key hyperparameters affecting model generalization, depth, and regularization.

TABLE II
PARAMETER GRID FOR GRIDSEARCHCV

Model	Parameter Values
Logistic Regression	penalty: [`l1, l2`]; C: [0.01, 0.1, 1, 10, 100]; solver: [`liblinear, saga`]
Random Forest	`n_estimators`: [100, 200, 300]; `max_depth`: [None, 10, 20, 30]; `min_samples_split`: [2, 5, 10]; `min_samples_leaf`: [1, 2, 4]; `bootstrap`: [True, False]
KNN	`n_neighbors`: [3, 5, 7, 9, 11]; `weights`: [`uniform, distance`]; `metric`: [`euclidean, manhattan`]
XGBoost	`n_estimators`: [100, 200, 300]; `max_depth`: [4, 6, 8]; `learning_rate`: [0.01, 0.05, 0.1]; `subsample`: [0.8, 1.0]; `colsample_bytree`: [0.8, 1.0]; `gamma`: [0, 0.1, 0.3]

After tuning, the models achieved their best hyperparameters summarized in Table III. The Random Forest model's best configuration produced perfect evaluation scores, while XGBoost slightly improved its recall and F1 performance, showing the effectiveness of parameter optimization.

The tuning process improved the consistency of the models performance. Random Forest achieved perfect scores in every metric, indicating its strong ability to capture complex, non linear patterns within the feature space. KNN and Logistic Regression produced similar results but showed less stability during cross-validation, while XGBoost demonstrated a clear improvement in recall and F1-score after fine-tuning. These findings indicate that tree-based models such as Random Forest are particularly effective for disease classification tasks where interactions among features significantly influence outcomes.

TABLE III
BEST HYPERPARAMETERS AND TUNED RESULTS

Model	Best Parameters	Accu	Prec	Rec	F1	Best CV
Logistic Regression	C=1, penalty=l2, solver=liblinear	0.9762	0.9881	0.9762	0.9762	1.000
Random Forest	bootstrap=True, max_depth=None, n_estimators=100	1.000	1.000	1.000	1.000	1.000
KNN	metric=manhattan, n_neighbors=7, weights=distance	0.9761	0.9880	0.9761	0.9761	0.966
XGBoost	colsample_bytree=0.8, gamma=0.1, lr=0.1, max_depth=4, n_estimators=300, subsample=0.8	0.9761	0.9900	0.9900	0.9800	0.894

C. Evaluation of Best Model

Random Forest achieved perfect classification performance with an accuracy, precision, recall, and F1-score of 1.0000. Table IV presents the performance metrics of all optimized models evaluated on the independent test set following hyperparameter tuning.

TABLE IV
PERFORMANCE METRICS OF OPTIMIZED MODELS

Model	Accuracy	Precision	Recall	F1-Score	CV Mean
Logistic Regression	0.9762	0.9881	0.9762	0.9762	1.0000
Random Forest	1.0000	1.0000	1.0000	1.0000	1.0000
K-Nearest Neighbors	0.9761	0.9880	0.9761	0.9761	0.9660
XGBoost	0.9761	0.9900	0.9900	0.9800	0.8940

This exceptional result indicates that Random Forest successfully captured the complex relationships between the 132 symptom features and 42 disease classes. The consistent cross-validation score of 1.0000 validates the model's robustness and stability across data folds. Random Forest's ensemble nature and implicit feature selection through tree aggregation effectively handled the high-dimensional symptom space and reduced prediction variance [4]. Logistic Regression maintained competitive performance with an accuracy of 0.9762 and demonstrated exceptional cross-validation stability (1.0000), making it an excellent alternative for clinically interpretable predictions [6]. KNN achieved similar test accuracy (0.9761) but exhibited lower cross-validation stability (0.9660), suggesting potential sensitivity to data partitioning and inconsistent generalization across varying distributions. XGBoost achieved competitive test accuracy (0.9761) with notably high precision and recall (0.9900), but demonstrated the lowest cross-validation mean (0.8940) among all models. This suggests that XGBoost's gradient-boosting framework may be prone to slight overfitting despite regularization measures [7]. Nevertheless, the model's performance remains clinically acceptable for disease prediction applications.

V. CONCLUSION

This study successfully compared four machine learning algorithms, K-Nearest Neighbors, Logistic Regression, Random Forest, and XGBoost for symptom-based disease prediction using a dataset comprising 132 symptom features mapped to 42 disease classes. The methodology involved comprehensive data preprocessing, hyperparameter optimization using GridSearchCV with 5-fold cross-validation, and systematic performance evaluation.

The results demonstrate that the Random Forest algorithm achieved the highest and most consistent performance, attaining perfect classification scores across all evaluation metrics with stable cross-validation results. Its ensemble-based architecture effectively mitigates overfitting and captures complex, non-linear interactions among symptom features and disease classes.

The strong performance of Logistic Regression also underscores the continued relevance of simpler and interpretable models in clinical settings, where transparency and explainability are crucial alongside predictive accuracy [6].

Future work should focus on analyzing feature importance from the Random Forest model to derive clinically meaningful insights, validating the model on real-world patient datasets, and integrating interpretability frameworks such as SHAP values to improve clinical applicability.

In conclusion, this comparative analysis demonstrates the potential of machine learning for accurate and reliable symptom-based disease prediction, providing a solid foundation for developing intelligent, ML-assisted clinical decision support systems.

References

[1] M. M. Ahsan *et al.*, "Machine-learning-based disease diagnosis," *PMC (open access) / [publisher unspecified]*, 2022. [Online]. Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC8950225/

[2] Kaushil268, "Disease prediction using machine learning," https://www.kaggle.com/datasets/kaushil268/disease-prediction-using-machine-learning, 2020, [Online].

[3] M. I. Ibrahim and A. M. Abdulazeez, "The role of machine learning algorithms for diagnosing diseases," *Journal of Applied Science and Technology Trends*, vol. 2, no. 01, pp. 10–19, 2021. [Online]. Available: https://jastt.org/index.php/jasttpath/article/view/79/23

[4] A. Sony, M. A. K. Thaile, M. A. Kiran, R. B. Pittala, B. N. Kumar, G. S. Chandu, and M. Raju, "Predictive health monitoring system for patient risk assessment using machine learning," in *Proceedings of the 2025 5th International Conference on Intelligent Technologies (CONIT)*, 2025, pp. 1–7.

[5] M. Haddouchi and A. Berrado, "A survey and taxonomy of methods interpreting random forest models," *arXiv preprint arXiv:2407.12759*, 2024. [Online]. Available: https://arxiv.org/abs/2407.12759

[6] S. Nusinovici, Y. C. Tham, M. Y. Chak Yan, D. S. W. Ting, J. Li, C. Sabanayagam, T. Y. Wong, and C.-Y. Cheng, "Logistic regression was as good as machine learning for predicting major chronic diseases," *Journal of Clinical Epidemiology*, vol. 122.

[7] U. Author(s).

[8] J. Jasmir, S. Nurmaini, and B. Tutuko, "Fine-grained algorithm for improving knn computational performance on clinical trials text classification," *Big Data and Cognitive Computing*, vol. 5, no. 4, p. 60, 2021.

[9] R. S. A. Daulay, S. Efendi, and Suherman, "Review of literature on improving the knn algorithm," *Transactions on Engineering and Computing Sciences*, vol. 11, no. 3, pp. 63–72, 2023.

[10] Anonymous, "The proper application of logistic regression model in complex survey data: A systematic review," *BMC Medical Research Methodology*, vol. 25, no. 1, p. 15, 2025.

[11] J. J. Levy, "Don't dismiss logistic regression: The case for sensible extraction of interactions in the era of machine learning," *BMC Medical Research Methodology*, vol. 20, no. 1, p. 171, 2020.

[12] M. I. Prasetiyowati, N. U. Maulidevi, and K. Surendro, "The accuracy of random forest performance can be improved by conducting a feature selection with a balancing strategy," *PeerJ Computer Science*, vol. 8, p. e1041, 2022.

[13] W. M. Jinjri, P. Keikhosrokiani, and N. L. Abdullah, "Machine learning algorithms for the classification of cardiovascular disease—a comparative study," in *2021 International Conference on Information Technology (ICIT)*, 2021, pp. 1–6.