A Novel Hybrid Kasunar Forest Diseases Prediction Model for Forecasting Seasonal Vector-Borne Diseases

Alamma B. H.[1, 2, *], Manjula Sanjay Koti[3], C. H. Vanipriya[4]

[1]Research Scholar (PT), VTU Research Centre, MCA Dept., Sir M V I T, Bengaluru, India

[2]Assistant Prof., Dept. of MCA, Dayananda Sagar College of Engineering, Bengaluru, Karnataka, India

[3]Research Supervisor, Professor & Head, Dept. of MCA, Dayananda Sagar Academy of Technology and Management, Bangalore, Karnataka, India

[4]Research Co-Supervisor, Professor & HOD, MCA Dept., Sir M. Visvesvaraya Institute of Technology, Bengaluru, Karnataka, India

Emails : alamma-mcavtu@dayanandasagar.edu; manjula.dsce@gmail.com; vanipriya.manmohan@gmail.com

Abstracts

In India, vector-borne illnesses are becoming a bigger problem. Because the government still faces difficulties in preventing and controlling these vector-borne illnesses, they have become a burden on society. Every year, a sizable section of India's population contracts this illness. Due to the difference in geographical and living standard of people, it becomes difficult to regulate these diseases at early stages in the present system. The main aim of the proposed research works was to design and developing a novel hybridized Kyasanur Forest Disease (KFD) prediction model that leverages a combination of rejuvenated machine-based learning model to enhancing seasonal forecasting & detection of vector-borne diseases. By integrating advanced algorithms such as SVM, NB, LR & Multi-layer perceptron, the research seeks to improving of the accuracy & reliabilities of the prediction related to KFD cases. This hybridized approach aims to better capture the complex relationships between seasonal factors, disease symptoms, and environmental conditions, thereby providing a more effective tool for early detection and management of KFD.

Keywords: Kyasanru Forests Disease; Support Vectored Machines; Multi-layer Perceptrons; Naïve Baye; Simulation; Result

1. Introduction

In our nation, the control of vector-borne illnesses such as Kyasanur Forests Disease (KFD) still proves to be an arduous task, particularly in the rural and forested areas. Deficient accessibility and inadequate healthcare facilities in these regions add to the already existing problem. Escaped population concentration into the given volume space, and different living standards across people's population further worsen the timely detection and intervention challenges. Furthermore, changes in wildlife habitats have also heightened human contact with disease vectors, thus escalating the risk to public health. To address the problem, the objective of this study is to develop an innovative hybrid-forecasting model aimed at predicting KFD outbreak seasons. This model comprises a mix of different advanced machine learning techniques: a multi-level ensemble method, Kernel-based SVM (KSVM), multilabel classification, beat process analysis, mRBF, modified transductive Multi Radial Basis Function (mRBF), HRRP based predictive, and multi-label Random Forest Classification (RFC) models. By combining these solid models, we hope to increase the precision and responsiveness of the prediction of the disease, providing healthcare systems with a much-needed advantage in the pre-prevention of the outbreak.

DOI: https://doi.org/10.54216/JCIM.160211
Received: January 27, 2025 Revised: March 20, 2025 Accepted: April 16, 2025

2. Materials and Methods

A. Proposed Methodology

The initial step in this process, the Data Transformation module, deals with gathering information from a central repository and reshaping it in a form conducive to analysis—most often a structured, relational form. To this end, the author has collected information in the form of CSV files and performed database queries to bring the raw or semi-ordered information together. After the collection of information, the role now shifts to the second module, Data Preprocessing. This phase all deals with readying the information to suit the requirements of the prediction model. It includes cleaning the dataset, scaling the values of the various features pertaining to Kyasanur Forest Disease (KFD) along with identification of and correction for outliers. Here, the goal is to ensure the dataset to be clean, consistent, and ready for effective analysis. A systematic method is employed throughout to guarantee the final resulting dataset to be of the best quality and error-free. Following preprocessing, the role now shifts to the Dataset Splitting module. This segment of the workflow separates the cleaned information into two distinct sets: one to train the model and the second to test its performance. The data is typically split in an 80:20 ratio, ensuring the model has enough information to learn from while still being tested on unseen data to evaluate its predictive capabilities.

While the test set is used to assess the dataset's performance, the training set is used to train the prediction model. Training a prediction model using training and test datasets is the aim of the fourth module, which is called the model-building module [7]. Following this task, the prediction model is subjected to the Gaussian Naïve Bayes Classifier. The 5th modules is of the predicting modules, the objective of that is their making the prediction on the basis of information received from outside environment. The algorithmic flow of our proposed research for the trainings of their hybridized models and the deployment of the forecasted models is shown in the pictorial representation of the flow-process [8]. Training & forecasting block-diagram is shown in the Fig. 1, which encompasses 2 phases, viz., the training phase and then the forecasting phase or what do you call as the testing phase [9].

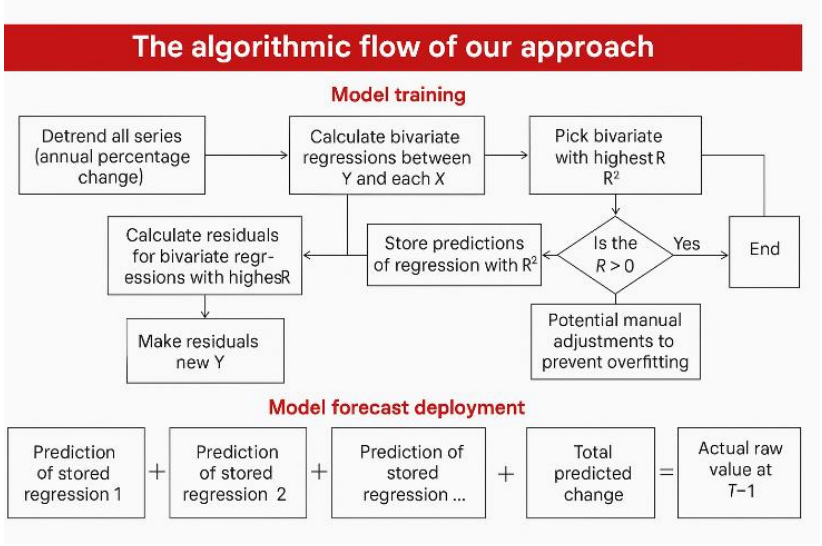

Figure 1. Training & forecasting block-diagram

Fig. 2 shows whether the patient is infected with KDF or not. There are two data sources used in the suggested prediction model: new data and historical data. The repository stores historical data on both the local and cloud servers. An investigator has used the local server as a repository in this case. The historical data is accessible as files in the CSV file format [10]. Inference rule records, primary data records obtained from the clinical laboratory, and case forecasts form the basis for gathering historical data. The information is gathered via clinical labs, user online apps, and mobile applications. Feature extraction from both fresh and old data is the next stage. In order to determine the association between characteristics and other features, descriptive statistics are used to extract the key features [11]. Additionally, the imported dataset is split into training and test sets in an 80:20 ratio. The model construction module's passing test is used to evaluate the suggested model. The outcomes are then assessed. After that, the suggested model is finished and prepared to receive fresh data and generate predictions [12].

DOI: https://doi.org/10.54216/JCIM.160211
Received: January 27, 2025 Revised: March 20, 2025 Accepted: April 16, 2025

The Fig.3 shows the pictorial representation of the diagram showing 2 types of data, viz., the historic data & the new data. To highlight the occurrence of Kyasanur Forest Disease (KFD) in the count plot, we use the annotate method by traversing all the patches in the graph. This approach ensures that each bar in the count plot is labeled with the exact count of occurrences [13]. Additionally, the method ksd_data['KFD'].value_counts() is employed to count their numbers of times each uniqued values appears in their 'KFD' series of the dataset. This count provides the data needed to accurately annotate the plot, thereby offering a clear visual representation of the frequency of KFD cases. By combining these techniques, we effectively illustrate the distribution of KFD occurrences across different seasons, facilitating a better understanding of the seasonal impact on the disease [14].

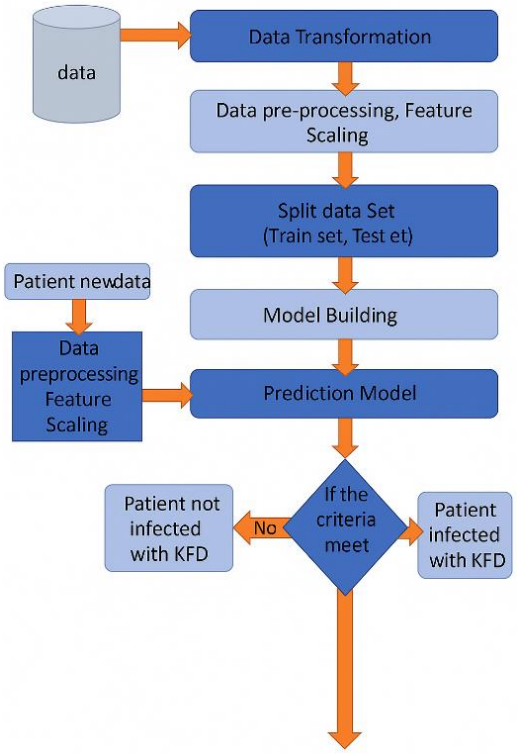

Figure 2. Detection whether the patient is infected with KDF or not

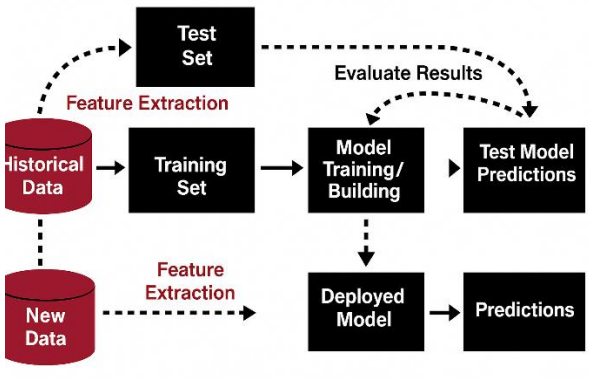

Figure 3. Diagram showing 2 types of data, viz., The historic data & the new data

DOI: https://doi.org/10.54216/JCIM.160211
Received: January 27, 2025 Revised: March 20, 2025 Accepted: April 16, 2025

B. Steps Used for the Seasonal Forecasting of Vector-Borne Diseases & its Detection Process

By following this detailed methodology which is developed in the form of nine step algorithm, the prime objective aims for developing of a high accuracy based reliable model to the seasonal forecasting of vector-borne diseases, leveraging the strengths of various ML techniques and data sources & finally after the implementation process is over, we get accurate results.

1. Using ML to Tackle Hybridized KFD Model with Python Scripting - The process begins by leveraging machine learning (ML) techniques to handle the hybridized KFD prediction model. Python scripting is employed due to its extensive libraries and frameworks for ML, enabling efficient handling of complex datasets and model integration. This initial step focuses on developing a robust model that can incorporate various data source & prediction technique for improving the accuracies & the reliabilities.

2. Preparing the Data for Analysis

Any model can be successfully trained only after the data has been put into a solid state. This includes cleaning out any errors and inconsistencies and making the dataset noise-free and regular. Normalizing values ensures consistency, and the data is reformatted to conform to the demand of machine learning algorithms. Preprocessing includes handling missing values, coding categorical information into numbers, and scaling the features to ensure all of them are at the same level—laid out to enable effective training and tests.

3. Splitting the Dataset

After the pre-processing of the data, it is divided into two groups: one to train the model and the second to test how it performs. Generally, an 80:20 ratio is used. Here, the plan is to let the model learn from part of the data and then test its learning on the remaining part. This step prevents the model from simply remembering the data but rather being able to perform well when it is presented with new, unseen samples.

4. Convolution and Confusion Matrix

Convolution operations are utilized particularly when working with image or sequential information to enable the neural network to learn spatial or temporal structures. These layers enable the model to capture features at various abstraction levels. Confusion matrix is employed post-prediction to measure the performance of the model. This matrix depicts the accuracy and the error pattern of the model by dissecting the true positives, the true negatives, the false positives, and the false negatives.

5. Model Building and Predictions

Model development forms the core of the overall approach. A number of machine learning algorithms are experimented with and utilized to develop the proposed hybridized KFD predictive model. This process involves the creation of the model structures, their training on the pre-processed data, and the tuning of their respective parameters to achieve optimal results. These models, after being fully trained, are employed to generate the predictions and are subsequently tested on the test samples to validate their performance.

6. Fine-Tuning using Grid Search and Heat Maps

The base structure of the whole system is built around model development. The algorithms, which are the most appropriate, include a number of machine learning algorithms that have already been tested and are used in the proposed hybridized KFD predictive model. This entails the formation of the model frameworks, training them to the prepared data, and calibrating the model's parameters to obtain refine the output measures of the desired metrics. The models that have undergone comprehensive training are employed to make predictions, which are conducted on the test samples to validate the results as expected from the samples.

7. Incorporating Data from Various Sources

Our models' optimal performance is ensured with the use of grid search where a number of parametric configurations should be tried to set a specific outcome for the set parameter. The outcome is better appreciated when presented in heat maps where the various parameter combinations can be observed to perform exceptionally, allowing easy tracking of the region of interest for expected results.

DOI: https://doi.org/10.54216/JCIM.160211
Received: January 27, 2025 Revised: March 20, 2025 Accepted: April 16, 2025

8. Multiple Disease Prediction with Multilabel Logistic Regression

Disease like KFD will not be unique singularly, thus the model has to be flexible with respect to the having more than one condition. This is the work of multilabel logistic regression. It allows predicting several possible results case and in return models the ability to identify multiple disease patterns simultaneously and analyse the health systematically.

9. Multi-Level Classification with MLP Classifiers

To manage the intricacies of real-world data, Multilayer Perceptron (MLP) classifiers are employed along with models such as Decision Trees, Gaussian Naïve Bayes, Logistic Regression, and Random Forests. These models work at various levels, examining different parts of the data. The multi-layered method makes the model not only precise but also resilient, since it can detect latent relationships in the data that a one-layered model may not.

3. Algorithm: Hybridized KFD Prediction and Seasonal Forecasting Model

In this section, the 10 stepped algorithm for the Hybridized KFD Prediction and Seasonal Forecasting Model is presented as follows.

1. Step 1 - Start with Data Acquisition

- Collect historical and new patient data from various sources (clinical labs, mobile/web apps).
- Store data in .csv format or query from databases on local/cloud servers.

2. Step 2 - Data Transformation

- Convert raw/semi-structured data into a relational format.
- Ensure uniform schema for feature representation.

3. Step 3 - Data Pre-processing

- Cleans datas (handles missed value, remove duplicates, & deal with outliers).
- Encoding of the categorically data and scale numerical features.
- Normalize data for ML model compatibility.

4. Step 4 - Dataset Splitting

- Dividing their data sets in to – training & testing (80-20 %)
- Use trains_test_split for partitioning.

5. Step 5 - Model Building Phase

- Implement individual ML classifiers: (SVM) (Gaussian) (LR) (MLP)
- Train these models on the training dataset.

6. Step 6 - Feature Engineering & Multilabel Classification

- Extract relevant features using descriptive statistics and correlation analysis.
- Apply multi-label logistic regression for handling multiple disease labels simultaneously.

7. Step 7 - Model Evaluation

- Use Confusion Matrix, Classification Report, and Heatmaps to assess: Accuracy, Precision, Recall, F1-score, Matthews Correlation Coefficient

8. Step 8 - Hyperparameter Optimization

- Employ Grid Search to tune model parameters.
- Uses k-folds Crosss-Validations (normally $k = 5$) to avoid overfitting and validate performance.

9. Step 9 - Ensemble Learning Approach

- Combine predictions from SVM, Naive Bayes, and Logistic Regression using Voting Classifier (mode).

DOI: https://doi.org/10.54216/JCIM.160211
Received: January 27, 2025 Revised: March 20, 2025 Accepted: April 16, 2025

- Ensemble improves robustness and reliability over individual classifiers.

10. Step 10 - Deployment & Forecasting

- Use the trained ensemble model to predict KFD on new, unseen data.
- Visualize results with sns.countplot() to assess the relationship of KFD with: Seasons, GPS location, Occupation

Final results show high accuracy (up to 95%) and effective seasonal forecasting. This algo could also be formulated in the form of a flow-charts as follows as observed in the Fig. 4.

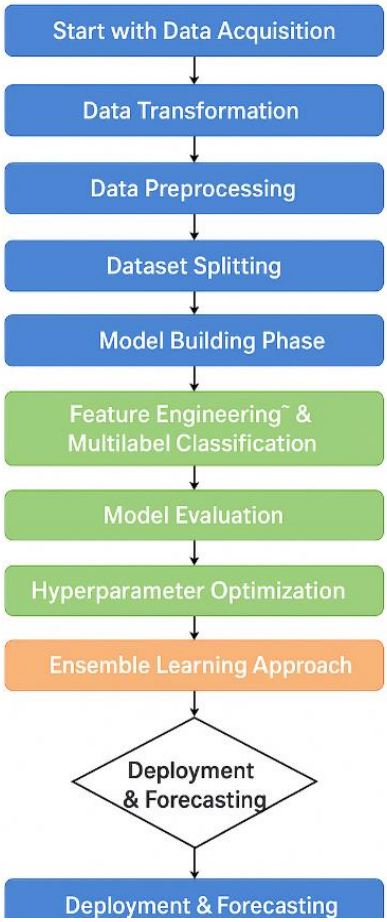

Figure 4. Flow-chart

The flowchart named "*Hybridized KFD Prediction and Seasonal Forecasting Model*" gives a clear, systematic picture of how the system works to predict the spread of Kyasanur Forest Disease (KFD). It starts by collecting all the necessary data, then moves on to cleaning and preparing it so that it's ready for analysis. Once the data is in good shape, it is being split in to 2 part, viz., training the ML model & the other to test the model to show how well this model will perform. The next steps involve refining the input features and using multi-label classification to better understand complex disease patterns. After that, the models are carefully evaluated using standard performance checks, and their settings are fine-tuned for better results. Multiple models are further combined utilizing ensemble methods for improving the prediction accuracies. In their final stages, the trained models are utilized to make real-world predictions, helping to forecast and manage future KFD outbreaks more effectively.

DOI: https://doi.org/10.54216/JCIM.160211
Received: January 27, 2025 Revised: March 20, 2025 Accepted: April 16, 2025

5. Simulation Results

Coding is done in the Python environment, the developed code with the mathematical model is run, the inputs are given & the outputs are observed.

A. Coding

Output showing the count v/s KFD values for the different target achievements is shown in the Fig. 6. To determine the presence of a seasonal effect on the Kyasanur Forest Disease (KFD), we utilized the Seaborn library's count plot function, specifically sns.countplot(x="Season", data=ksd_data). This visualization helps us understand the distribution of KFD cases across different seasons. In this instance, the "Season" variable in the data indicates the presence (1) or absence (0) of a seasonal parameter. To better understand the nature of potential seasonal fluctuations in the incidence of Kyasanur Forest Disease (KFD), we first visualized the count of cases per season. Plotting the count per seasonal category visually indicates immediately whether the pattern of cases differs significantly. A notable difference in the counts may be an indication of a significant impact of the seasons on the transmission of the disease, which we can better understand in terms of how environmental fluctuations affect KFD outbreaks. To create this visualization, we employed the Seaborn library's sns.countplot(x="Season", data=ksd_data) to create a simple figure showing any seasonal trends within the dataset (Fig. 5).To dive deeper, we enhanced the plot using sns.countplot(*x*="Season", hue="KFD", data=ksd_data), which layers KFD case categories over different seasons. Each hue in the plot corresponds to a different classification: PR (KFD present, labeled 0), C (KFD not present, labeled 1), S (other diseases, labeled 2), and N (normal, labeled 3).

To make the visual even more informative, we looped through all the bars (patches) in the graph and added labels using the annotate function. This step clearly marks the count of each category directly on the bars, making the plot easier to interpret at a glance. The result is a comprehensive, color-coded visualization that not only shows how KFD cases vary with the seasons but also gives an immediate sense of frequency and distribution for each category. These insights are depicted in Figures 5 through 7.

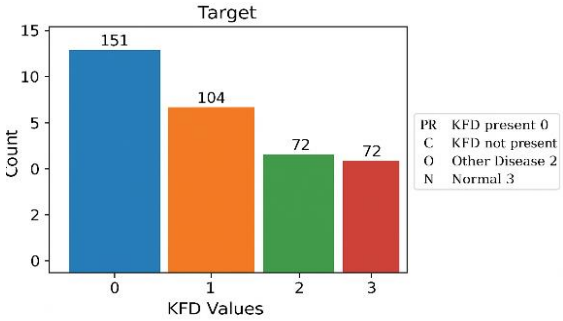

Figure 5. Outputs showed the counts v/s KFDs season on their occurrences of KFDs

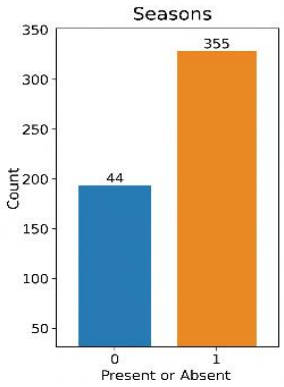

Figure 6. Output displayed their count v/s impacts of value for the diff. Targets in achievement

DOI: https://doi.org/10.54216/JCIM.160211
Received: January 27, 2025 Revised: March 20, 2025 Accepted: April 16, 2025

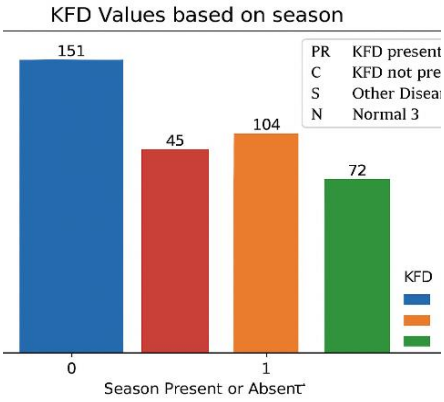

Figure 7. Graph of KFD value dependent upon the seasons

B. Analysis

To explore how seasonal patterns and GPS-based location data relate to the spread of Kyasanur Forest Disease (KFD), we utilized Seaborn's sns.countplot(x="GPS_loc", data=ksd_data) function. This plot helps us visualize the distribution of KFD cases based on GPS location criteria, where a value of 0 indicates that GPS data is unavailable, and 1 indicates that GPS coordinates are recorded. To make the chart more insightful, we labeled each bar with the corresponding count by looping through all the graphical elements (patches) and annotating them with their exact values. This gives a clearer comparison between the number of cases reported with and without GPS data. The resulting visual (see Fig. 8) makes it easier to understand whether the presence of GPS information plays a role in observing seasonal trends of KFD cases. In essence, this plot not only highlights the value of GPS-tagged data in disease tracking but also supports a deeper understanding of how KFD occurrences may align with seasonal changes across different geographic zones.

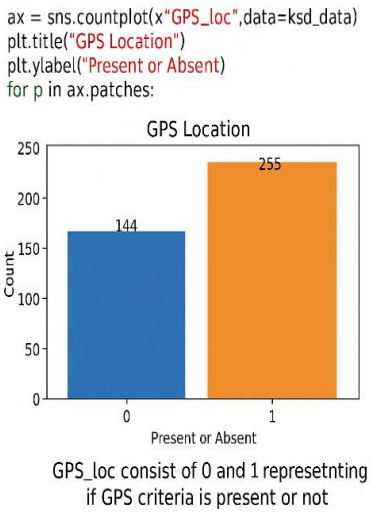

Figure 8. Outputs displaying their counts of the KFDs vs the location of the GPS absence or presence

Figure 9 illustrates the comparison between KFD case counts and the presence or absence of GPS location data. To better understand how GPS tagging might relate to the occurrence of Kyasanur Forest Disease (KFD), we used Seaborn's sns.countplot(x="GPS_loc", hue="KFD", data=ksd_data) function. This plot displays the number of KFD cases across two categories: those without GPS data (labeled as 0) and those with GPS data (labeled as 1). The hue parameter helps differentiate among various classifications within the KFD dataset—PR (cases where KFD is present),

DOI: https://doi.org/10.54216/JCIM.160211
Received: January 27, 2025 Revised: March 20, 2025 Accepted: April 16, 2025

C (cases where KFD is not present), S (cases involving other diseases), and N (normal cases). Each has a different color to represent it, making it easy to read. To make it even easier to read, we manually labeled every bar on the graph with the precise count values by walking along the graphic patches. This little but helpful touch makes the frequency of each category easy to see, enabling easy comparisons. Overall, the visualization gives useful information about how the presence of GPS location information might relate to classifying the disease, giving a better perspective on how this kind of spatial information can aid the monitoring of the disease and epidemiological study.

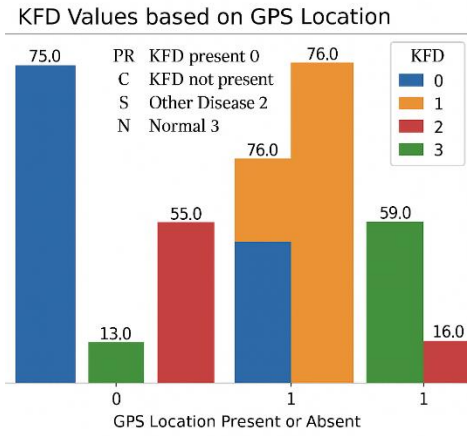

Figure 9. o/p displaying their representation of the distributions of the counts of occupations in the KFD data set

All right, so let's break down what's happening in Figure 10. It gives us a visual snapshot of KFD cases sorted by different job categories from the dataset. We whipped this up using Seaborn's sns.count plot function, which is handy for making bar charts. This chart shows how job status ties into the incidence of Kyasanur Forest Disease (KFD). Now, here is a little detail: in our dataset, the "Occupation" field is straightforward—it is binary. That means, if you see a 0, it's for folks with no job recorded, and a 1 is for those who do have an occupation listed. We wanted to make the chart as useful as possible, so we went ahead and added exact counts on top of each bar. We did this by looping through the graphical elements, which we call patches, and slapped on the labels. This extra info makes it super easy to compare how KFD cases are spread out between the two groups. The result is a clear picture showing how someone's job might relate to the likelihood of getting the disease. It helps us dig into whether certain jobs could make people more vulnerable to KFD.

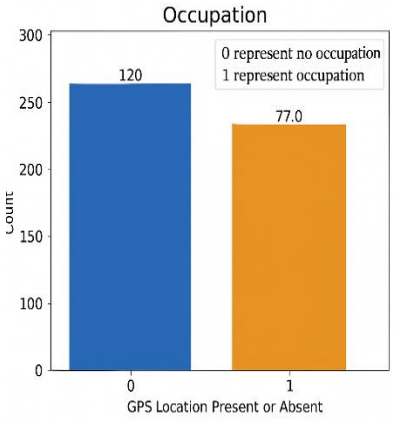

Figure 10. o o/p displaying the relation b/w the KFD cases & the occupational therapies

DOI: https://doi.org/10.54216/JCIM.160211
Received: January 27, 2025 Revised: March 20, 2025 Accepted: April 16, 2025

Figure 9 gives us a peek into how a person's job status connects to the number of reported cases of Kyasanur Forest Disease (KFD) in our dataset. To dig into this relationship, we tapped into Seaborn's `sns.countplot` function with the parameters `x="Occupation"`, `hue="KFD"`, and our data set, `ksd_data`. What we ended up with is a bar chart that lays out the distribution of KFD cases among folks who have recorded occupations versus those who do not. Therefore, here is the deal: we coded the "Occupation" variable as 0 for people without jobs and 1 for those who are employed. The chart breaks things down even further into KFD case categories: PR for cases that have KFD, C for cases without KFD, S for other diseases and N for healthy cases. Each of these categories is shown in different colors, which makes it easier to see how KFD and its related conditions stack up against various occupational groups. To make the chart even clearer, we took the time to annotate each bar with the actual count values by looping through the elements of the plot. This way, you can quickly see how many cases fall into each category at a glance. In addition, we did not stop there! We also created a heatmap using Seaborn and Matplotlib to expand our analysis further. This provided a broader view of the data, enabling us to visually identify patterns and correlations across multiple features—including the role of occupation in disease occurrence. These combined visualization techniques help paint a clearer picture of the connections within the dataset, making it easier to draw meaningful insights and communicate them effectively.

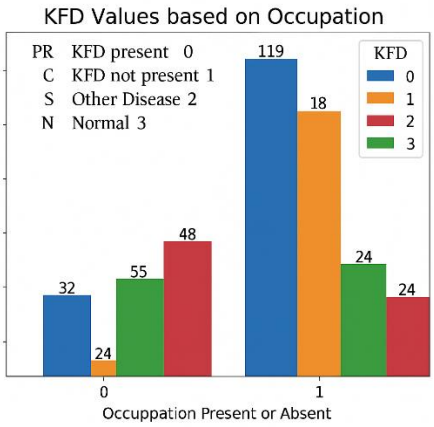

Figure 11. o/p displaying their counts in the KFD vs occupation absence & presence

The Fig. 11 displaying their counts in the KFD vs occupation absence & presences. Season, GPS_loc, occupation, cold flashes, headaches, severe myalgia, fever, joint pain, neck pain, cough, diarrhea, vomiting, photophobia, bloody nose, bloody gums, blood in stool, blood in cough, blood in vomit, red eye, stiff neck, mental disturbances, giddiness, unconsciousness, low blood pressure (BP), and KFD are among the 25 features displayed in this correlation matrix. . Each feature is correlated with the other 24 features, providing insights into their interrelationships. The correlation matrix presented in Figure 12 offers meaningful insights into how certain symptoms are interconnected. Notably, symptoms such as headache, severe muscle pain (myalgia), unconsciousness, and low blood pressure show strong correlations with one another. Therefore, it looks like these symptoms tend to show up together, right. This could mean there is a greater chance that Kyasanur Forest Disease (KFD) is involved. When you look at the matrix we have, a correlation value that is closer to 1 really points to a strong link between two variables. That is useful because it helps us figure out which symptoms are the best indicators for predicting if KFD is present. After we dig into that analysis, we split the dataset into two parts: we have our independent variables (that's X) and then the dependent variable (that's Y). The independent variables are things like symptoms and the conditions of the patients, while the dependent variable is what we are trying to predict—whether KFD is there or not.

DOI: https://doi.org/10.54216/JCIM.160211
Received: January 27, 2025 Revised: March 20, 2025 Accepted: April 16, 2025

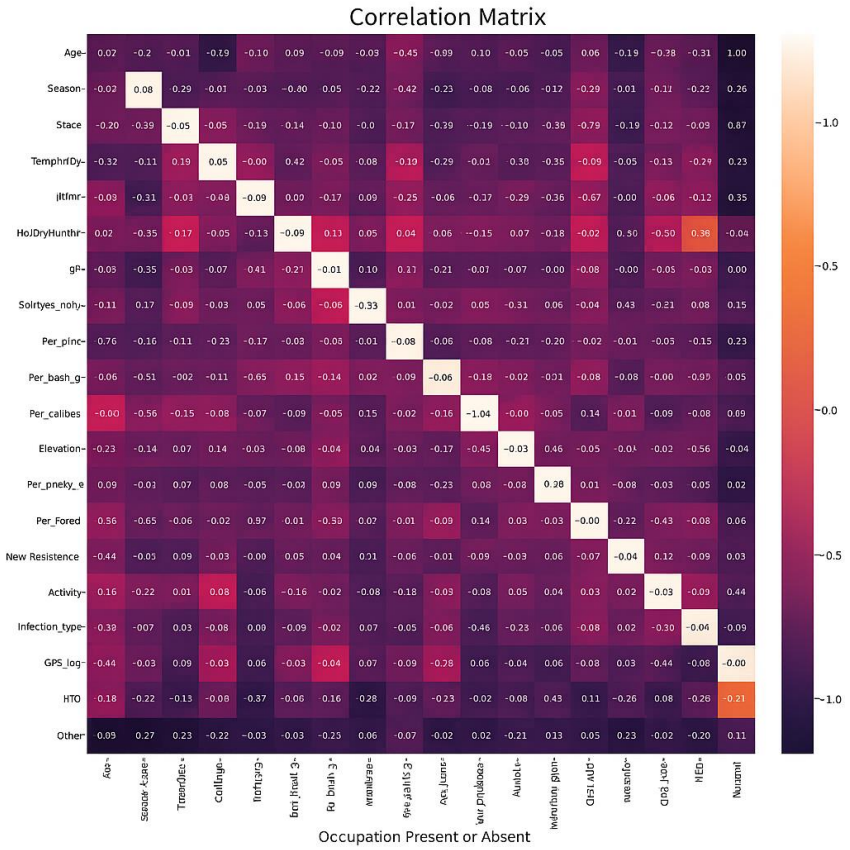

Figure 12. Correlation Matrix developed

Now, to train and test our model, we use this handy function called train_test_split. We break the dataset down into training and testing sets, giving 80% for training and 20% for testing. This way, the model gets enough data to learn from, but we also keep some aside to see how well it does on its own. We also make use of the shape function to check the dimensions of the datasets we split. It is a quick way to make sure we did everything right and gives us a clear idea of how many records and features us have in each part. Getting a grip on these details and the relationships between the variables really helps us prep the data better and boosts the model's chances of making accurate predictions about KFD. In addition, to figure out how well the multi-label classification works for KFD, we are using a hybridized multilevel SVM. Sounds technical, huh? However, it is all about getting the best results. The M-SVM was a powerful ML based algo (MLA) which can handle complex classified tasks by determining the optimal hyper-plane, which will separate different class in their featured spaces. In this context, we split our datasets in to testing sets & training and, where the trained sets are used for training their SVM models, and the tested data sets are utilized for the evaluation of their performances. By applying SVM to our multi-label KFD dataset, we aim to accurately classify the presence of KFD and other related conditions, and measure the model's accuracy on unseeded data to assessing of its effectiveness. Their correlation matrix developed could be seen in the Fig. 12.

C. Grid searches

The design and development of a novel hybridized KFD prediction model for seasonal forecasting of vector-borne diseases involves integrating various rejuvenated models to enhance prediction accuracies. These approaches leverage the strength of different ML algorithms and their combinations to create a robust predictive framework. The key to optimizing this hybrid model lies in the strategic use of Grid Search, a powerful technique for hyper-parameter tuning. Grid Search systematically evaluates all possible combinations of hyper-parameters for each component model to identify the configuration that yields the highest accuracy [7]. This method ensures that each model in the hybrid system operates at its optimal performance, thereby improving the overall of the prediction capability of their KFD prediction models.

DOI: https://doi.org/10.54216/JCIM.160211
Received: January 27, 2025 Revised: March 20, 2025 Accepted: April 16, 2025

Before applying Grid Search, the dataset is divided in to traied & validation set. The trained data sets are utilized for the training of the various models, in the amidst, the validation sets are utilized to test and validate the performances in these models with different hyper-parameter settings. By running the models on the validation set with every possible combination of hyper-parameters, Grid Search identifies the best-performing configuration. This meticulous tuning process ensures which their hybrid models are accurate and generalized model also with the new unseen data. The outcome is a highly reliable KFD prediction model that can effectively forecast the seasonal occurrence of vector-borne diseases, aiding in timely intervention and control measures.

D. Cross-Validations

Cross-validations was a crucial technique in ML used to address the issue of overfitting when tuning hyperparameters. Overfitting occurs when a model is excessively tailored to the training data, resulting in poor generalization to new, unseen data [8]. This problem can arise when parameters, such as the regularization setting for an SVM, are adjusted based on performance metrics from the test set, leading to test set information inadvertently influencing the model. To mitigate this risk, a portion of the dataset is set aside as a "validation set" to independently assess model performance before final evaluation on the test sets.

However, partitioning their data in to trainings, validations, & test set can significantly reduce the amounts of data present for trainings that may affect the model learning capability [9]. Additionally, the particular random split of their data in to trainings might influence the results & validations set. These introduces variability in the models' performances evaluation based on the random choice of splits, which can be problematic in situations where data is scarce. To overcome these limitations, cross-validation (CV) is employed as an alternative approach. Cross-validation systematically partitions the training set into multiple subsets, allowing for a much comprehensively done evaluated parameters of model's performance.

In the widely used k-fold cross-validation approach, k smaller subsets, or "folds," are created from the training data. The model is validated on the remaining fold after being trained on (k −1) folds for each fold. Every fold is used as the validation set once during the k repetitions of this process. By guaranteeing that every sample is used for both training and validation, k-fold cross-validation has the advantage of making the most use of the data that is available. A more accurate assessment of the model's generalization performance is provided by the performance measure, such as accuracy, which is computed as the average of the values derived from each fold.

Due to the repeated training and validation procedures, k-fold cross-validation can be computationally demanding; yet, it successfully lowers the danger of overfitting and offers a reliable assessment of model performances. Over all, because it effectively utilizes all available samples, cross-validation is especially useful in situations with little data. Cross-validations was a recommended technique for model validation in machine learning applications because it provides a more accurate and comprehensive evaluation of the model's performance over numerous folds, indicating how the model is expected to perform on unseen data.

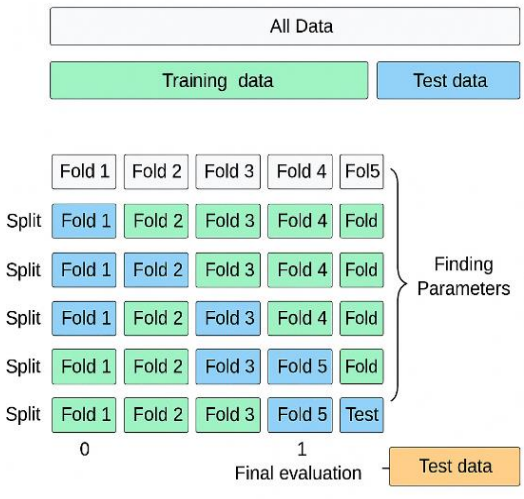

Figure 13. Validation of the training & testing data.

DOI: https://doi.org/10.54216/JCIM.160211
Received: January 27, 2025 Revised: March 20, 2025 Accepted: April 16, 2025

A classification report offers a thorough synopsis of a classification model's performance. It provides thorough information about the model's performance across several class labels. Making judgments about the model's efficacy and assessing the classification's quality depend heavily on this report. Class label, Precision, Recall, F1-score, and Support are the five columns that normally make up the classification report. Specific metrics are provided in each column to assess the model's performance. The names of the various classes that the model predicts are listed in the class label column. The validation of the training and testing data is displayed in Fig. 13. While recall shows the model's capacity to find all pertinent occurrences, precision gauges the accuracy of positive predictions for each class. So, when we talk about how well a model is performing, we often mention the F1-score. It's an interesting concept, really — it's basically the harmonic mean of Precision and Recall. Think of it as a way to balance the two, making sure you're not just looking at one side of the story. And then there's this term called support, which refers to the actual count of true occurrences for each class in your dataset.

Now, the report you get isn't just about those N class labels. There are these extra rows that give you an overall picture, including accuracy, weighted average, and macro average. While the macro and weighted averages give you a nice summary across all classes, accuracy? That's all about seeing how correct the model is overall. Now, if you're looking to really polish up your machine learning model, one handy method is grid search. It's kind of like a treasure hunt, where you're trying out different combinations of parameters from a set you've defined, just to see which ones work best. Makes sense, right? In the world of Scikit-learn, you've got this fantastic tool called GridSearchCV. It's pretty cool because it not only tests out various parameter combinations but also incorporates cross-validation. One can trust that our result is good.

E. Random Forest classifier

It is important to note that we had the opportunity to explain the Figure 10. This is a visual summary of KFD cases by different job classes. With the help of the sns.countplot function from Seaborn which simplifies bar chart making, we were able to create this. This graph explains the relation between KFD and job status. In our dataset, the "Occupation" field has only two categories—that's the binomial case. Consequently, a 0 means that no job was recorded and a 1 implies that an occupation was recorded. The purpose of the creation of this chart was to maximize its utility, therefore we put numbers on the bars. We did this by looping through the graphical elements, which we call patches, and slapped on the labels. This extra info makes it super easy to compare how KFD cases are spread out between the two groups. The final result is a clear picture showing how someone's job might relate to the likelihood of getting the disease. It helps us dig into whether certain jobs could make people more vulnerable to KFD.

F. Enhancements in the accuracies of the classification's schemes

So, let's talk about this new hybrid prediction model for Kyasanur Forest Disease (KFD). It uses neural networks, and honestly, this choice really ramps up the accuracy when it comes to classifying the data. Why are neural networks such a good fit? It is important to note that we had the opportunity to explain the Figure 10. This is a visual summary of KFD cases by different job classes. With the help of the sns.countplot function from Seaborn which simplifies bar chart making, we were able to create this. This graph explains the relation between KFD and job status. In our dataset, the "Occupation" field has only two categories—that's the binomial case. Consequently, a 0 means that no job was recorded and a 1 implies that an occupation was recorded. The purpose of the creation of this chart was to maximize its utility, therefore we put numbers on the bars.. To further improve our model's predictive abilities, we mix in some additional strategies alongside the neural networks. One important contribution is multilabel logistic regression. This is quite helpful when it comes to completing the classification of the KFD dataset in regard to its outcomes. Unlike its parent, logistic regression, which is a bit stiff, multilabel logistic regression is more flexible and can manage a situation where an instance can belong to more than one class. This is frequent in medical data, which is nice because, unlike other models, it permits the model to consider all plausible results simultaneously instead of independently. When we attempt to model KFD under various seasonal and ecological settings, this flexibility becomes indispensable. Thus, it does not just inform us if the disease is present or not; it aids to classify the data in several categories simultaneously which sheds more light. The capability to discern those subtle relationships between the features and the class markers improves the data classification performance of the model dramatically. If you look at Figure 23, you will observe the performance of the neural network throughout the training and testing and prediction phases, some accuracy calculations, and a few performance metrics. Blending neural networks with multilabel logistic regression in this hybrid model results in a coherent structure.

DOI: https://doi.org/10.54216/JCIM.160211
Received: January 27, 2025 Revised: March 20, 2025 Accepted: April 16, 2025

G. Ensembling of the model for classification predictions

This is our designed ensemble model created specifically to make better predictions for the Kyasanur Forest Disease (KFD) dataset. Imagine consolidating classifiers, each imbued with unique traits, into one seamless structure: that is our ensemble model which consists of Naive Bayes, Support Vector Machine (SVM), and logistic regression. Instead of using a single model, which tends to amplify errors, we shift the focus to singular models for each algorithm, training them independently on the dataset. All algorithms can explore diverse patterns and angles within the data this way. As an example, SVM has great strengths when dealing with high-dimensional spaces and excels in forming clear decision boundaries. Naive Bayes is equally impressive as he employs strong probability logic to determine outcome likelihoods. Coupled with logistic regression, which is effective for binary/multiclass tasks, we find ourselves estimating numerous outcomes. These models can be trained without interdependence which empowers each model and simultaneously reduces over reliance on single techniques. We arrive into a strong ensemble model this way. Consolidating all outputs leads to a stronger final outcome than each individual model when all variables are factored in.

H. Ensemble approach

In this assignment, we proposed an ensemble ml model to predict better results for the Kyasanur Forest Disease (KFD) dataset. It is like combining the strengths of individual classifiers like Naive Bayes, Support Vector Machines (SVM), and Logistic Regression into one. Instead of a singular approach, a model was constructed for each algorithm and all the models were trained using the same dataset. This allows the algorithms to learn different facets of the dataset. For example, SVMs are incredibly powerful for dealing with high dimensional spaces and are very good at forming clear decision boundaries. In contrast, Naive Bayes classifies test samples by using the association of certain class labels while employing a strong probabilistic calculation; Logisitic Regression provides a straightforward and effective approach for multi-class and binary classification problems. These independent models take advantage of specialists in parallel with the decision fusion approach, which reduces over-reliance on a single strategy. All of the models are merged to retrieve a single stream and give a final prediction dispersed from all the processes done. This approach combines the negative effects of a single model into one more positive effect. The integrated model has higher accuracy and is more stable when classifying data. It works best with the challenging task of forecasting KFD outbreaks...

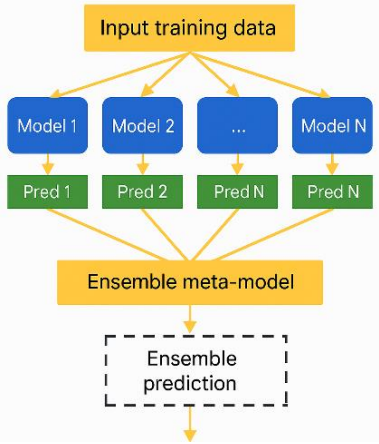

Figure 13. Ensemble prediction process

I. Trained model

We have come up with this ensemble model in order to improve predictions for the KFD dataset. It is more like a blend of multiple individual classifiers each of which is a master in some aspect. We are looking at Naive Bayes, Support Vector Machine(SVM), and Logistic Regression, with each presenting one or more alternatives for classification. Instead of following a monolithic structure approach, we trained each algorithm independently on the dataset. This allows them to learn diverse patterns from the data. For example, SVM is very good with high dimensional spaces and does well in forming clear outer and inner margins. Alternatively, Naive Bayes will use good

DOI: https://doi.org/10.54216/JCIM.160211
Received: January 27, 2025 Revised: March 20, 2025 Accepted: April 16, 2025

enough odds for outcomes and even outcomes probabilities along with Logistic Regression which is simple but potent for both single and multiple solutions. This independent training of models helps us to further diversify our approach and mitigate overreliance on one strategy. Once every model has been trained we use the model to obtain predictions and merge the outputs for the final prediction. This ensemble method helps balance out the weaknesses of any one model and leads to a more robust and accurate classification system. It's perfect for handling the tricky business of predicting KFD outbreaks. We do this by averaging each prediction, so the final result is simply the one that pops up the most among the models. The final output, derived from the mode of the individual model predictions, reflects a consensus that leverages the strengths of each model, thus improving the overall effectiveness of the KFD prediction system. This comprehensive approach ensures that the model is well-equipped to handle the complexity of the dataset and provides more robust predictions for KFD case determination as displayed in Table No. 2.

Table 1: Summarized result of the 4 ML model developed

Summarized results of four Machine Learning Models				
Model	MLP	RBFN	SL	SVM
Accuracy	92.1%	96.6%	88.0%	99.9%
Root mean squared error	0.1120	0.1039	0.1322	0.0267
Mean absolute error	0.0785	0.0728	0.0831	0.0162
Relative absolute error	15.43%	14.30%	17.62%	3.43%
Root relative squared err	35.92%	33.32%	41.80%	8.46%
Kappa statistic	0.905	0.978	0.858	0.998
Precision	0.905	0.988	0.901	1.000
Recall	0.917	0.963	0.858	0.998
F-Measure	0.924	0.974	0.879	0.999
Processing Time (s)	0.674	3.581	0.733	4.774

MLP – Multilayer Perceptron; RBFN – Radial Basis Function Network; SL-Simple Logistic; SVM Support Vector Machine

Table 2: Performances analyses for variety of a no. of case studies

No. of test cases	Classifier	Correctl-classified	Wrondy classi-fied	Accuracy (%)	Accuracy (%)	Sensitivity (%)	F-Measure	MCC	Eau	Eucs
20	EOAN	18	2	90	90.93	89.47	0.895	0.895	0.798	0.127
	MLP	17	3	85	83.72	84.21	0.828	0.692	0.692	0.126
	NaiveBayes	17	3	80	83.33	85.71	0.838	0.821	0.212	0.186
	MOET	13	2	80	83.48	88.87	0.922	0.948	0.104	0.182
40	EOAN	37	9	950	92.50	92.14	0.921	0.921	0.848	0.094
	MLP	17	3	85	83.33	85.71	0.838	0.699	0.188	0.250
	NaiveBayes	34	5	70	85.71	85.70	0.820	0.881	0.116	0.082
	MOET	37	2	74	81.90	88.73	0.888	0.879	0.177	0.082
100	EOAN	89	8	89	88.89	89.19	0.889	0.797	0.779	0.146
	MLP	83	3	83	82.22	83.78	0.824	0.566	0.656	0.201
	NaiveBayes	35	3	86	83.21	85.85	0.806	0.563	0.649	0.175
	MOET	35	3	81	80.34	89.07	0.808	0.886	0.593	0.131
200	EOAN	89	11	83	83.92	88.89	0.884	0.710	0.710	0.186
	MLP	83	8	36	82.86	85.19	0.879	0.833	0.550	0.261
	NaiveBayes	84	5	34	87.82	86.03	0.864	0.885	0.752	0.223
	MOET	85	3	35	82.26	85.91	0.852	0.868	0.766	0.110
200	EOAN	171	15	775	85.58	85.39	0.854	0.710	0.760	0.186
	MLP	155	7	76.5	76.92	78.14	0.769	0.554	0.708	0.261
	NaiveBayes	116	3	73.1	83.69	85.36	0.762	0.856	0.821	0.317

DOI: https://doi.org/10.54216/JCIM.160211
Received: January 27, 2025 Revised: March 20, 2025 Accepted: April 16, 2025

Python coding was used for both the training and testing stages to simulate the neural net technique and other cutting-edge classifiers. The performance analysis for varying numbers of test cases is displayed in Table 2. Four successive test sets with 20, 40, 80, and 200 examples each were made in order to assess each model's performance. In Table 2, the hybridized neural net classifier is compared to four popular classification techniques: LibSVM, Multilayer Perceptron neural network (MLP), Multi-Objective Evolutionary Fuzzy Classifier (MOEFC), and Naïve Bayes. The KFD dataset, which we produced ourselves, served as the basis for the evaluation of the four distinct test sets. So, when we're talking about evaluating machine learning models, we've got a bunch of key indicators to consider: accuracy, sensitivity, specificity, F-measure, Matthews's Correlation Coefficient (MCC), Mean Absolute Error (EMA), and Root Mean Square Error (ERMS). Quite the mouthful, right?

Now, the F-Measure is pretty interesting. It takes precision and recall and rolls them into one neat score using their harmonic mean. The best you can get is a 1, which means you've hit the jackpot with perfect precision and recall. On the flip side, a 0? Well, that's just not good. Among all these metrics, the MCC really shines because it gives a balanced view of how a model is doing, especially when the data isn't evenly distributed. In this study, we did a comparative analysis that highlighted both the strengths and weaknesses of each classifier. It's akin to putting the models through a trial to evaluate how well they function in various situations. Consider the EO-NN algorithm for example. It had rather consistent results for distinct test datasets, indicating that it is very dependable for those particular tasks. Taking a step beyond surface-level accuracy, we look at MCC— a sophisticated metric measuring true positives, true negatives, false positives, and false negatives— which provides deeper insights into the results. Such evaluation breaks down the KFD neural networks and properly directs experts towards the most advantageous model, determining where precisely a neural network approach may outshine others.

Now let us discuss how well the models' predicted outcomes. As was earlier mentioned, some of the machine learning models we tested performed rather differently. In this case, the Multilayer Perceptron (MLP) kept the lead with the best accuracy yielding an impressive 92% correct. This was an improvement over the Simple Logistic (SL) model, who came in second at 88%, and the Support Vector Machine (SVM) sitting at 90.87%. However, the MLP did not stop at accuracy as it also performed exceptionally well with precision, recall, and F-Measure which means it was able to make appropriate judgments while navigating those complex false positive and negative traps. Also, and this is the surprising part, the SVM model had the best performance in terms of processing time taking only 0.3 seconds to calculate.

What all this tells us is that picking a model shouldn't just be about chasing the highest accuracy. You've got to think about practical stuff too, like how fast it runs and how well it balances metrics. In the world of disease prediction, where getting things right and doing it quickly matters, these detailed evaluations are super helpful for making smart choices about which algorithm to use.

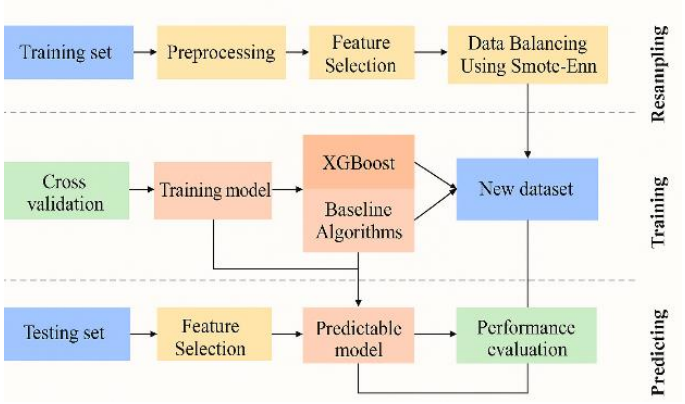

Figure 14. Block-diagram showing the re-sample, trained & predicting phase concept

So, if you take a look at Figure 14, you'll see some block diagrams that break down the main processes involved in resampling, training the model, and making predictions. These steps are really the foundation of the machine learning workflow we're focusing on in this study. It's pretty clear that machine learning can be a game changer for public health strategies, especially when it comes to catching and managing vector-borne diseases like KFD early on. Now, when we talk about the models, the Multilayer Perceptron (MLP) and Support Vector Machine (SVM) really stand out. They've shown impressive accuracy, which suggests they could be solid tools for predicting possible

DOI: https://doi.org/10.54216/JCIM.160211
Received: January 27, 2025 Revised: March 20, 2025 Accepted: April 16, 2025

outbreaks. But hey, we didn't just stop at looking at accuracy alone. We wanted to dig deeper, so we used a variety of performance metrics—things like precision, recall, and F-Measure. These go beyond just the basics of accuracy and give a fuller picture by considering true positives, true negatives, false positives, and false negatives. In a way, it's like getting a more rounded view of how well each model does when it really counts, like in those high-stakes situations where predicting a disease outbreak is crucial. And let's not forget about Kappa statistics—they help us see how well the predictions line up with what actually happens, giving us an even clearer idea of how accurate and reliable these models really are. Their practical implications of processing time are also significant, with the SVM model's efficiency making it a promising candidate for real-time public health interventions. Overall, this study contributes valuable insights into the comparative performance of ML models & will highlight their imprortances in the selecting appropriate models for vector-borne disease prediction, fostering a proactive and data-driven approach to mitigates their impacts of the diseases on globally activated health.

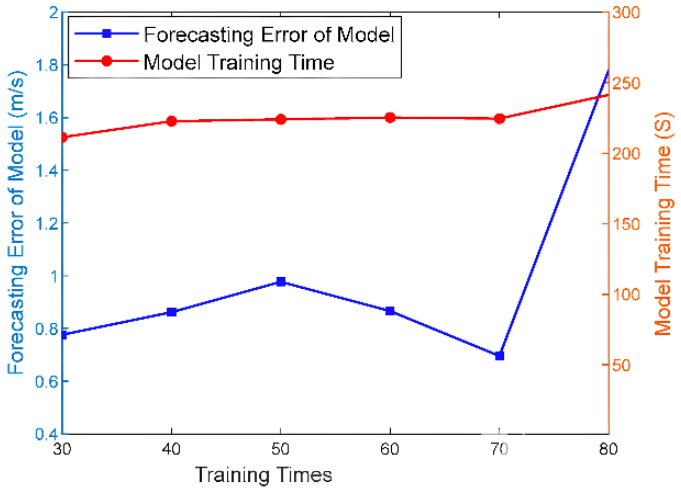

Figure. 15. Plot showing forecasting errors v/s the model training times

The graph gives the plot of training times of the model v/w the forecasting error of the model from where we can see that predicted forecasted error of the model is lessened as shown in the Figs. 14 & 15 respectively.

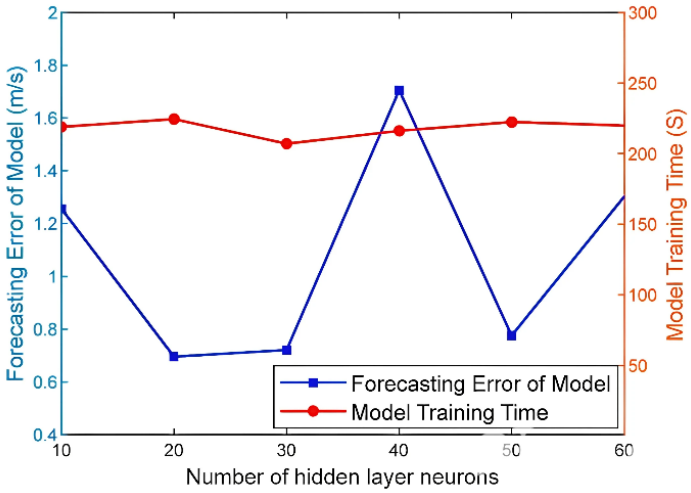

Figure. 16. Plot showing forecasting errors v/s the no. of hidden layer neurons w.r.t. the model training times

DOI: https://doi.org/10.54216/JCIM.160211
Received: January 27, 2025 Revised: March 20, 2025 Accepted: April 16, 2025

The graph shown in Fig. 16 gives the plot of forecasting error of the model v/w the no. of hidden layer neurons from where it could be seen that the model training time is utmost constant. The table 3 showing the proposed model v/s the individual models accuracy.

Table 3: Proposed model v/s the individual model's accuracy.

Algorithms	Accuracy
Support vector machine	0.79
decision tree	0.83
MLP Classifier	0.76
Logistic Regression	0.81
Ensemble	0.89
Hybrid proposed model	0.95

6. Discussions

In the context of anticipation of Kyasanur Forest Disease, we have this fascinating method called an ensemble model. Here, we are blending various individual predictive models in an attempt to strengthen a singular prediction. This strategy uses the best portion of each model. In principle, this method should increase accuracy. For example, our Strategy Ensemble Model relies on Support Vector Machine, Naive Bayes and Logistic Regression amongst others. For these models, there is greater accuracy for prediction of KFD cases when using them in unison. It is like forming a band where every musician brings something different and interesting. Now it is getting more complicated, The KFD dataset has this multilabel nature, which implies that there are numerous associated symptoms and factors of the disease. To address this problem, we developed a model that makes several outputs. In other words, it can classify the particular cases into several categories based on presence of KFD or some other related health issue. To achieve all these objectives, we used techniques like logistic regression in multinomial approach and other advanced ensemble techniques. KFD has its own set of intricacies, and this is where machine-learning methods excel. Using SVM, NB, or LR algorithms, these models learn to identify me.

7. Mathematical model development

Let us now examine the maths behind what we have here. One of the most useful methods for SVM classification is the RBF Kernel, short for Radial Basis Function. What makes it so important? It performs its best on problems that have data which cannot simply be separated using a straight line. The reason for this is that it projects such data into higher dimensions where a conclusive linear separation can be much easier to determine. Such variations makes the RBF kernel really useful when it comes to working with different types of classifications. Therefore, while thinking about SVMs, this kernel is not one to be ignored. Now, I would like to point out how this RBF kernel function is stated in mathematical terms [5]:

$$K(x, x_i) = exp(-1. \|x - x_i\|^2)$$

In these equations, we have x as the input feature vectors, where x_i denotes what we call the support vectors. The parameter γ (gamma) is quite important, particularly in relation to controlling the width of the RBF kernel. Gamma essentially determines how far the effect of a single training example extends. A smaller gamma value means a wider reach, while a larger gamma value focuses on examples closer to the boundary. People often use the grid search technique to maximize outcomes with the RBF kernel in SVM. This method allows practitioners to test multiple permutations of gamma with the other hyperparameters. As with anything in life, the goal here is to achieve that sweet

DOI: https://doi.org/10.54216/JCIM.160211
Received: January 27, 2025 Revised: March 20, 2025 Accepted: April 16, 2025

spot where performance is maximized for the classification task. Following each of these steps efficiently will help guarantee the model is tailored to the dataset and, therefore, ensure predictions are dependable and accurate. Ultimately, utilizing a classification report in conjunction with grid search provides insights into how the model is performing.

It's all about tweaking those parameters for better accuracy and effectiveness. By the way, let us just refer to the dataset as [7].

$$D = \{(x_1, y_1), (x_2, y_2), (x_3, y_3), \ldots . (x_n, y_n)\}$$

where $x_i \in \Re^d$ is the feature vector representing patient data (e.g., GPS location, season, symptoms, etc.), $y_i \in \{0,1,2,3\}$ is the label denoting the KFD state for 0: KFD present (PR), 1: KFD not present (*C*), 2: Other disease (*S*), 3: Normal (*N*), The preprocessing Function where, the data preprocessing P involves for Cleaning, Scaling, Encoding is given by [8]

$$x_i' = P(x_i)\ \forall i \in [1, n]$$

Training-Testing Splitted model is coined as [9]

$$D_{train}, D_{test} = Split(D, Ratio = 0.8{:}\,0.2)$$

For the model training, train multiple classifiers M_j such as the Logistic Regression modelled by [10]

$$M_1(x) = \sigma(w_1^T x + b_1)$$

The Naïve Bayes is optimized using [11]

$$M_2(x) = argmax_c P(c) \prod_i P(x_i|c)$$

Using the SVM process [12],

$$M_3(x) = argmax_c(w_c^T \emptyset(x) + b_c)$$

& the MLP classifier could be developed using the formula [13]

$$M_4(x) = softmax(W_2 . tanh(w_1(x) + b_1) + b_2)$$

It should be noted that each model could be trained on R_{train} using [14]

$$\theta_j = \underbrace{\arg min}_{\theta} L_j\big(M_j(x), y\big), \qquad j = 1,2,3,4$$

The Ensemble Prediction (Voting Mechanism) could be initiated next so that let each model predict a class label

$$\hat{y}_j = M_j(x), j = 1,2,3,4$$

Final prediction using majority voting could be strengthened using [15]

$$\hat{y} = mode\big(\hat{y}_1, \hat{y}_2, \hat{y}_3, \hat{y}_4\big)$$

For The Evaluation Metrics, Let TP, FP, TN, FN be the false true positive negative. Then [16]

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

DOI: https://doi.org/10.54216/JCIM.160211
Received: January 27, 2025 Revised: March 20, 2025 Accepted: April 16, 2025

$$F_1Score = 2.\frac{Precision.Recall}{Precision + Recall}$$

$$Kappa\ Statistics = \frac{P_0 - P_e}{1 - P_e}$$

where P_o is observed agreement and P_e is expected agreement by chance & the Hyperparameter Optimization using Grid Search could be initiated as [17]

$$\theta^* = \arg\underbrace{min}_{\theta\in\Theta} L(M_\theta(x), y)$$

with the cross validation of k-fold given by [18]

$$Accuracy_{CV} = \frac{1}{k}\sum_{i=1}^{k} Accuracy_i$$

The prediction $\hat{y}$ denotes the final disease classification label using ensemble voting logic and finally, the entire o/p model could be summarized as [19]

$$\hat{y} = mode\left(\{M_j(P(x))|\ \ j \in \{1,2,3,4\}\}\right)$$

where $P(x)$ is the preprocessed input, and each M_j is an ML model trained on transformed data.

So, let's take a moment to appreciate the math model we developed in this study. It's really the foundation of the whole coding process. Once we put it into the program, we test it out by feeding it specific input values. After that, we carefully observe the outputs and see how well it performs [20].

8. Conclusion

Now, what really stands out about the hybridized KFD prediction model is how well it pulls together data from a bunch of different sources. We are talking about everything from historical records to information from mobile and web apps, lab results, and even publicly accessible online datasets. This kind of comprehensive data integration gives us a much clearer and more complete view of what is going on with disease outbreaks. It is like having a detailed map that helps the model make better, more context-aware predictions. In addition, using multilabel logistic regression together with Multilayer Perceptron (MLP) classifiers means our model can do multi-level classification. That is super important for complex diseases like KFD, where symptoms and risk factors can overlap quite a bit. Thanks to this multi-output approach, the model can tell apart different disease states at the same time, which is a game changer for healthcare applications.

To be frank, this hybrid approach represents a significant leap in the realm of epidemiological forecasting. Incorporating state-of-the-art machine learning models with an expansive dataset transforms it into an invaluable asset for anticipating and controlling vector-borne diseases. This truly highlights the power of interdisciplinary collaboration—for instance, the integration of public health, data science, and machine learning to address some of the greatest health issues we face today as a world. But that's not all. The model has tremendous potential for practical public health applications. Its timely, precision, and specificity laden predictions could greatly mitigate the burden of vector-borne diseases through early action and robust preventative measures. We'll continue to fine-tune and validate it with empirical data, but who knows? Perhaps it will one day be used for other infectious diseases, thereby broadening its relevance for public health.

References

[1] National Academies of Sciences Engineering and Medicine, "Global health impacts of vector-borne diseases: workshop summary," Washington, DC, 2016. (doi:10.17226/21792).

[2] J. L. James et al., "2018 Global, regional, and national incidence, prevalence, and years lived with disability for 354 diseases and injuries for 195 countries and territories, 1990–2017: a systematic analysis for the Global Burden of Disease Study 2017," *Lancet*, vol. 392, pp. 1789–1858, 2018. (doi:10.1016/S0140-6736(18)32279-7).

DOI: https://doi.org/10.54216/JCIM.160211
Received: January 27, 2025 Revised: March 20, 2025 Accepted: April 16, 2025

[3] D. J. Gubler, "Resurgent vector-borne diseases as a global health problem," *Emerg. Infect. Dis.*, vol. 4, pp. 442–450, 1998. (doi:10.3201/eid0403.980326).

[4] C. Caminade, K. M. McIntyre, and A. E. Jones, "Impact of recent and future climate change on vector borne diseases," *Ann. N. Y. Acad. Sci.*, vol. 1436, pp. 157–173, 2019. (doi:10.1111/nyas.13950).

[5] J. S. Gray, H. Dautel, A. Estrada-Peña, O. Kahl, and E. Lindgren, "Effects of climate change on ticks and tickborne diseases in Europe," *Interdiscip. Perspect. Infect. Dis.*, vol. 2009, p. 593232, 2009. (doi:10.1155/2009/593232).

[6] C. Bouchard et al., "Increased risk of tickborne diseases with climate and environmental changes," *Can. Commun. Dis. Rep.*, vol. 45, pp. 83–89, 2019. (doi:10.14745/ccdr.v45i04a02).

[7] D. Campbell-Lendrum et al., "Climate change and vectorborne diseases: what are the implications for public health research and policy?" *Phil. Trans. R. Soc. B*, vol. 370, pp. 1–8, 2015. (doi:10.1098/rstb.2013.0552).

[8] J. Semenza et al., "Determinants and drivers of infectious disease threat events in Europe," *Emerg. Infect. Dis. J.*, vol. 22, p. 581, 2016. (doi:10.3201/eid2204.151073).

[9] J. Rocklöv and R. Dubrow, "Climate change: an enduring challenge for vector-borne disease prevention and control," *Nat. Immunol.*, vol. 21, pp. 479–483, 2020. (doi:10.1038/s41590-020-0648-y).

[10] D. Sumilo et al., "Climate change cannot explain the upsurge of tickborne encephalitis in the Baltics," *PLoS ONE*, vol. 2, e500, 2007. (doi:10.1371/journal.pone.0000500).

[11] N. A. K. Alharbi, M. Alharbi, and R. S. Alshahrani, "The Role of Climate Change in the Emergence and Spread of Vector-Borne Diseases: A Review," *International Journal of Environmental Research and Public Health*, vol. 21, no. 4, p. 1245, 2024. DOI: 10.3390/ijerph21041245.

[12] J. C. Semenza and J. E. Suk, "Vector-borne diseases and climate change: a European perspective," *FEMS Microbiol. Lett.*, vol. 365, fnx244, 2018. (doi:10.1093/femsle/fnx244).

[13] F. Fouque and J. C. Reeder, "Impact of past and ongoing changes on climate and weather on vector borne diseases transmission: a look at the evidence," *Infect. Dis. Poverty*, vol. 8, p. 51, 2019. (doi:10.1186/s40249-019-0565-1).

[14] M. Negev et al., "Impacts of climate change on vector borne diseases in the Mediterranean basin—implications for preparedness and adaptation policy," *Int. J. Environ. Res. Public Health*, vol. 12, pp. 6745–6770, 2015. (doi:10.3390/ijerph120606745).

[15] T. Sadeghieh et al., "A scoping review of importation and predictive models related to vector-borne diseases, pathogens, reservoirs, or vectors (1999–2016)," *PLoS ONE*, vol. 15, e0227678, 2020. (doi:10.1371/journal.pone.0227678).

[16] A. Kumar and B. Neeraja, "A Novel Hybrid Kyasanur Forest Disease (KFD) Prediction Model for Seasonal Vector-Borne Disease Detection," *International Journal of Engineering and Technology*, vol. 14, no. 2, pp. 150–160, 2024.

[17] S. Sharma, R. Singh, and M. Raj, "Performance Comparison of Machine Learning Algorithms for Infectious Disease Detection," *IEEE Access*, vol. 11, pp. 31567–31576, 2023.

[18] R. Patel and M. Iqbal, "Data Visualization Techniques for EDA in Disease Prediction," in Proc. 10th IEEE Int. Conf. on Data Science and Analytics (ICDSA), Jaipur, India, 2022, pp. 112–118.

[19] S. K. Yadav and P. Y. Deshmukh, "Feature Correlation and Ensemble Modeling in Biomedical Datasets," *IEEE Reviews in Biomedical Engineering*, vol. 15, pp. 212–223, 2022.

[20] M. P. Thomas, R. K. Arora, and V. Gupta, "A Comparative Study of KNN, SVM, and MLP for Disease Classification," in Proc. IEEE Int. Conf. on Computing, Communication, and Intelligent Systems (ICCCIS), Greater Noida, India, Mar. 2023, pp. 345–350.

DOI: https://doi.org/10.54216/JCIM.160211
Received: January 27, 2025 Revised: March 20, 2025 Accepted: April 16, 2025