Vector-Borne Disease Prediction

Rajat Kumar
IIITD
rajat21185@iiitd.ac.in

Rajat Vatwani
IIITD
rajat21186@iiitd.ac.in

Ramit Nag
IIITD
ramit21188@iiitd.ac.in

Yusuf Jamal
IIITD
yusuf21220@iiitd.ac.in

Abstract

This project aims to leverage machine learning for the early diagnosis of vector-borne diseases based on patient symptoms. We will employ a range of algorithms, including Random Forest, Logistic Regression, Decision Trees, Support Vector Machines, and Multi-Layer Perceptron Classifiers, to analyze the data and make predictions.

According to the World Health Organization's 2020 estimates, these diseases contribute to an alarming annual mortality rate of 700,000 individuals, posing a significant global health threat. We believe that machine learning holds the key to addressing this pressing issue, and this belief is what inspired us to pursue this project.

1. Motivation and Introduction

Vector-borne diseases represent a critical global health challenge, causing significant morbidity and mortality worldwide. These diseases, transmitted through vectors like mosquitoes, ticks, and fleas, include malaria, dengue, Zika virus, and Lyme disease, among others. They often affect populations in tropical and subtropical regions, where healthcare infrastructure may be limited, making early detection and intervention particularly challenging.

In this context, timely and accurate diagnosis is crucial in reducing the spread of these diseases and improving patient outcomes. Traditional diagnostic methods can be time-consuming, costly, and often rely on advanced laboratory equipment that is not readily accessible in affected areas. Therefore, there is a pressing need for innovative approaches to improve early detection, enabling quicker response and treatment to prevent severe outcomes.

This project aims to address the challenge of early diagnosis by leveraging machine learning techniques to construct predictive models. Using patient symptomatology as input features, we aim to develop a robust machine-learning model that can assist in the early detection of vector-borne diseases. By identifying patterns in symptoms and correlating them with disease prognosis, we hope to contribute to a more effective and accessible solution to mitigate the global impact of vector-borne diseases.

2. Literature Survey

1. **Disease Prediction using Machine Learning**[1]
The paper presents a system for predicting diseases using machine learning. The authors employed Decision Tree, Random Forest, and Naïve Bayes classifiers to identify diseases based on patient symptoms. They used a dataset of 4,920 patient records, encompassing 41 diseases and 132 symptoms, and selected 95 key symptoms to optimize accuracy and avoid overfitting. It provides a detailed description of each algorithm and its implementation.

2. **Vector Borne Disease Outbreak Prediction by Machine Learning.** [2]

The paper predicts outbreaks of Chikungunya, Malaria, and Dengue in India using a CNN-based approach, leveraging demographic (positive cases) and meteorological data (temperature, humidity, rainfall) from 2013 to 2017. The CNN-MDOP algorithm classifies regions into low, moderate, or high risk, achieving 88% accuracy by using CNN for feature extraction and a Softmax classifier for prediction. Combining weather and case data improved outbreak prediction accuracy.

3. **Application of Machine Learning in Disease Prediction** [3]
The paper applies various machine learning classification algorithms to predict diseases such as heart disease, breast cancer, and diabetes. They utilize the Heart Disease Dataset, Wisconsin Breast Cancer Dataset, and Pima Indians Diabetes Dataset from the UCI repository for their anal-

ysis. Some of the key models and techniques used are Logistic Regression, Decision Trees, Random Forest, Support Vector Machine (SVM), and AdaBoost.

3. Dataset

3.1. Description

In this section, we provide an overview of the dataset used in this study and the preprocessing steps applied to prepare the data for modeling. The dataset contains 1,010 records with 66 columns, and it consists of records that capture the symptoms and medical information of patients. It includes the following key attributes:

- **ID**: A unique identifier for each record.
- **Symptoms**: A set of 64 binary features indicating the presence (1) or absence (0) of specific symptoms for each patient. These symptoms serve as input features for the predictive model, and they include various indicators like fever, headache, vomiting, muscle pain, etc.
- **Prognosis**: The target variable that represents the medical outcome or disease diagnosis for each patient. This column has categorical values such as "Lyme disease," "Zika," "Rift Valley fever," and others.
- **Dataset Dimension**: The dataset contains 64 features, leading to a high-dimensional dataset. This highlights the need for dimensionality reduction techniques like PCA to manage and extract essential patterns efficiently.

3.1.1 Initial Observations

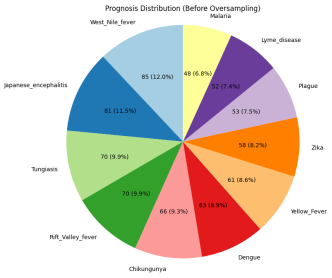

Figure 1. Sample Class Distribution

Upon an initial examination of our dataset, several key observations emerged. The dataset is free of missing values, ensuring data integrity, and it maintains a clean structure without any duplicate records. There are 11 distinct prognosis categories, reflecting the complexity of the conditions we aim to predict. All features are one-hot encoded, indicating the presence or absence of specific symptoms as binary values.

3.1.2 Correlation Matrix

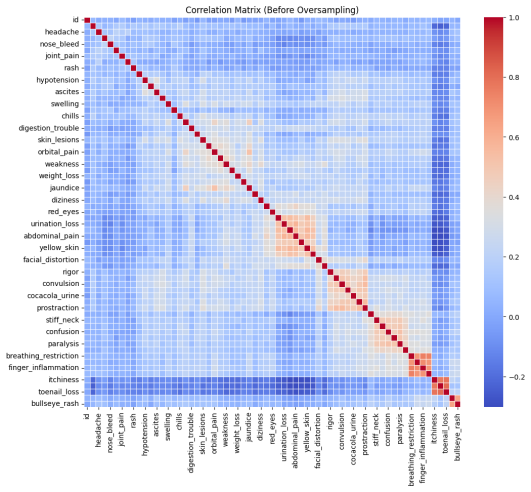

Figure 2. Sample Class Distribution

To better understand relationships between features, we computed a correlation matrix for the binary symptom features. It revealed moderate feature collinearity, with some symptoms frequently co-occurring, indicating relationships between specific symptoms. Strong correlations between certain features suggest interdependencies and potential redundancy. These insights highlight the need for dimensionality reduction techniques like PCA to address feature redundancy and optimize the dataset for modeling.

3.2. Data Preprocessing

3.2.1 Label Encoding (Also Handling Missing Values)

The target variable, *prognosis*, which is categorical, was label-encoded into numerical values to facilitate processing by machine learning models. Missing values in the feature columns were handled by imputing the mean of each column, ensuring that all entries in the dataset are complete and ready for training.

3.2.2 Dataset Splitting

The processed dataset was split into *training* and *testing* sets using an 80:20 ratio. This ensures the model is trained on a majority of the data and evaluated on unseen data, providing an accurate measure of performance on future predictions.

3.2.3 Principal Component Analysis (PCA)

PCA was applied for dimensionality reduction while retaining 95% of the dataset's variance. This step helps in speeding up the training process by reducing the number of features, while also minimizing the risk of overfitting by eliminating irrelevant information.

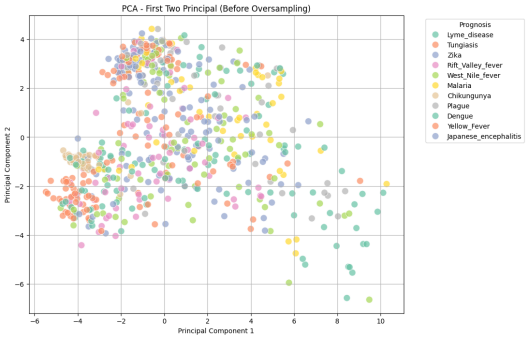

Figure 3. PCA for Dimensionality Reduction

3.2.4 Data Balancing and Using Oversampling (SMOTE)

To address the class imbalance in the dataset, *SMOTE* (Synthetic Minority Over-sampling Technique) was used. This technique generates synthetic samples for minority classes, balancing the dataset and ensuring that models don't bias toward the majority class.

3.2.5 Data Standardization

All feature columns were standardized using *StandardScaler* to ensure they are on a similar scale. This is particularly important for models that rely on distance calculations, as standardization improves overall performance by preventing features with larger scales from dominating the learning process.

4. Methodology

We employed a systematic machine learning approach to develop a disease prediction model using patient symptoms. After data loading and preprocessing, missing values were imputed with mean values, and the categorical target variable (prognosis) was label-encoded. To handle class imbalance, SMOTE was applied, generating a balanced dataset. Features were standardized, and PCA reduced dimensionality while retaining 95% of variance. The dataset was split into training and testing sets (80:20). During training, a custom bootstrapping function trained models on multiple bootstrapped samples, with predictions aggregated via majority voting. We used models such as Naive Bayes, Decision Tree, Random Forest, Logistic Regression, and MLP, evaluating their performance on the test set using classification metrics and accuracy.

4.1. Model Details

4.1.1 Naive Bayes (GaussianNB):

A probabilistic model based on Bayes' theorem, which assumes that all features are independent. Since Naive Bayes is a low-variance model and inherently stable, using bootstrapping isn't particularly useful, and in our analysis, we found that there is only a difference of 1% in accuracy with and without using bootstrapping, which is expected.

4.1.2 Decision Tree Classifier:

It is a robust tree-based model that recursively splits the dataset based on feature values, creating a tree-like structure. Each decision node represents a condition related to a feature, while the leaves show the final class predictions. While decision trees can be effective, they often struggle with overfitting, performing well on training data but poorly on new data. To address this, we used ensemble methods like bootstrapping, which led to an impressive 8% increase in accuracy compared to not using it.

4.1.3 Random Forest Classifier:

An ensemble model that builds multiple decision trees using bootstrapped samples and aggregates their predictions to achieve higher accuracy. In a Random Forest, each decision tree is trained on a bootstrapped sample of the original dataset. Bootstrapping is essential here as it involves creating multiple datasets by sampling with replacement, so each dataset is slightly different from the original one. This introduces diversity among the trees, which is crucial for the Random Forest to achieve better generalization.

4.1.4 Logistic Regression:

Logistic regression is a linear model for binary classification that estimates the probability of a target class using a logistic function. It supports L1 regularization (Lasso), which helps select important features by setting some coefficients to zero, and L2 regularization (Ridge), which reduces overfitting by shrinking coefficients without eliminating them. In our experiments, L2 regularization yielded results similar to standard logistic regression, while L1 regularization enhanced accuracy by 2%. The regularization strength is controlled by the parameter C, where larger C values imply weaker regularization and smaller C values indicate stronger regularization. The best performance was achieved around $C = 1$.

4.1.5 XGBoost Classifier:

XGBoost (Extreme Gradient Boosting) is a powerful machine learning algorithm based on the gradient boosting framework. It is widely used for classification, regression, and ranking tasks due to its high performance and speed. XGBoost builds an ensemble of decision trees in a sequential manner, where each tree tries to correct the errors of the previous trees.

5. Results and Analysis

Model	Precision	Recall	F1-score	ROC-AUC
Decision Tree	0.71	0.71	0.70	0.76
Random Forest	0.78	0.78	0.77	0.96
Logistic Reg	0.49	0.50	0.48	0.85
XGBoost	0.78	0.78	0.78	0.97
Naive Bayes	0.42	0.43	0.40	0.81

Table 1. Performance comparison of different models.

- **Precision**: XGBoost and Random Forest have the highest precision (0.78), meaning they are more reliable in predicting true positives. Naive Bayes has the lowest precision (0.42), making more false positive predictions.
- **Recall**: Random Forest and XGBoost also excel in recall (0.78), identifying most true positive cases. Naive Bayes performs poorly (0.43), missing many true positives.
- **F1-score**: XGBoost has the best F1-score (0.78), showing a great balance between precision and recall. Naive Bayes has the lowest F1-score (0.40), reflecting a poor balance between precision and recall.
- **ROC-AUC**: XGBoost (0.97) and Random Forest (0.96) have the highest ROC-AUC, indicating strong class separation. Decision Tree lags behind with a lower ROC-AUC (0.76), showing weaker class separation.

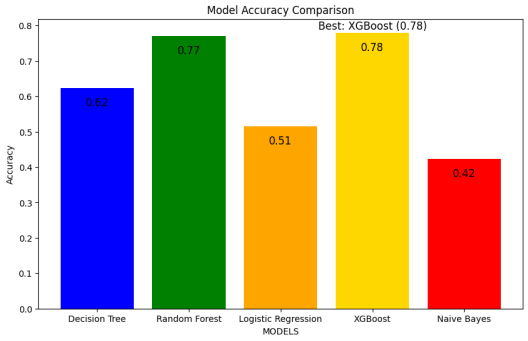

Figure 4. Accuracy for different models

The bar chart compares the accuracy of five different models: Decision Tree, Random Forest, Logistic Regression, XGBoost, and Naive Bayes. Among these models, XGBoost achieves the highest accuracy of 0.78, as highlighted in the chart. The Random Forest model also performs well, slightly lower than XGBoost. Logistic Regression and Decision Tree show moderate accuracy, while Naive Bayes has the lowest accuracy among all the models. This suggests that ensemble-based methods (Random Forest and XGBoost) are more effective for the dataset compared to simpler algorithms like Naive Bayes or Logistic Regression.

6. Conclusion

- **Best Precision**: The Random Forest model achieved the highest precision score of 0.78, making it the best choice when minimizing false positives (i.e., avoiding misclassification of non-disease cases as vector-borne diseases) is essential.
- **Best Recall**: Both Decision Tree and XGBoost models had the highest recall scores of 0.78, indicating their effectiveness in capturing the majority of true positives (i.e., correctly predicting the occurrence of vector-borne diseases). These models are ideal when accurately identifying disease cases is crucial.
- **Best Overall Performance**: The XGBoost model showed the highest ROC-AUC score of 0.97, reflecting its superior ability to distinguish between vector-borne diseases and non-disease cases. Therefore, XGBoost is the most suitable model for achieving balanced and reliable performance in vector-borne disease prediction.

In Summary, while Random Forest offers unmatched precision, Decision Tree and XGBoost provide a slightly better balance between precision and recall. XGBoost stands out with the highest ROC-AUC score, making it the optimal choice for accurate and reliable predictions in vector-borne disease cases.

7. Future Work

We plan to explore additional boosting methods such as AdaBoost and conduct hyperparameter tuning for both AdaBoost and XGBoost. Additionally, we will assess the performance of Support Vector Machines (SVM) and Multilayer Perceptron (MLP) for this task. We also aim to integrate various fairness measures into the classification process and evaluate the models accordingly.

8. Contributions

- Yusuf Jamal: Literature review, Data preprocessing, and visualization, XGboost, Report, PPT
- Rajat Kumar: Decision Trees, Random Forest, Naive Bayes, Code, Data preprocessing
- Rajat Vatwani: Logistic Regression, XGBoost, Literature Review, Report, PPT
- Ramit Nag: Naive Bayes, Literature review , Logistic Regression, PPT

References

[1] S. Grampurohit and C. Sagarnal, "Disease prediction using machine learning algorithms," in *2020 International Conference for Emerging Technology (INCET)*, 2020, pp. 1–7. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9154130

[2] S. Raizada, S. Mala, and A. Shankar, "Vector borne disease outbreak prediction by machine learning," in *2020 International Conference on Smart Technologies in Computing, Electrical and Electronics (ICSTCEE)*, 2020, pp. 213–218. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9277286

[3] P. S. Kohli and S. Arora, "Application of machine learning in disease prediction," in *2018 4th International Conference on Computing Communication and Automation (ICCCA)*, 2018, pp. 1–4. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8777449