Precision Learning for Fever Medication Prediction Using Machine Learning

Miltone Awiti
Department of Health Informatics
Michigan Technological University
Houghton, MI, USA
moawiti@mtu.edu

Syllas Otutey
Department of Health Informatics
Michigan Technological University
Houghton, MI, USA
sotutey@mtu.edu

Gideon Owusu
Department of Health Informatics
Michigan Technological University
Houghton, MI, USA
gideono@mtu.edu

Frank Ofosu
Department of Health Informatics
Michigan Technological University
Houghton, MI, USA
fyofosu@mtu.edu

***Abstract*—Fever is a common clinical symptom traditionally managed through standardized treatment protocols. While these guidelines provide a general framework, they often fail to consider the variability in individual patient responses. Recent advancements in machine learning offer new opportunities for enhancing fever management through precision medicine. By analyzing diverse patient-specific data such as age, medical history, genetic markers, and real-time clinical metrics, machine learning models can identify patterns and predict the most effective treatment strategies for everyone. This approach not only improves therapeutic outcomes but also minimizes the risks associated with one-size-fits-all treatments. This work explores the potential of integrating machine learning into clinical decision-making to personalize fever treatment, aiming to bridge the gap between generalized care and patient-specific needs.**

I. Introduction

A. Clinical Context

Fever is a common clinical symptom, often indicative of underlying infections or inflammatory conditions. Traditionally, its management has adhered to standardized protocols, which, while effective for general patient populations, may not account for individual variations in response to treatment. This gap underscores the importance of personalized medicine, where treatment strategies are tailored to the individual characteristics of each patient.

The integration of machine learning (ML) into healthcare presents a promising avenue for enhancing personalized treatment approaches. ML, a subset of artificial intelligence, encompasses various algorithms capable of identifying patterns within complex datasets. Techniques such as neural networks, linear regression, and support vector machines have been employed to analyze patient data, facilitating more accurate predictions and informed decision-making.

In the realm of fever management, ML has been utilized to predict fever onset and classify its types. For instance, a study by Singh et al. (2020) demonstrated that ML algorithms could predict fever onset in critically ill patients up to four hours in advance by analyzing continuous physiological data. Similarly, research by Zhang et al. (2020) employed ML to classify fever types, aiding in the differentiation between bacterial and viral infections.

These studies highlight the potential of ML in refining fever management. However, challenges remain, including the need for large, diverse datasets to train robust models and the integration of ML tools into existing clinical workflows. Additionally, the interpretability of ML models is crucial to ensure clinician trust and adoption.

This work aims to explore the application of ML in fever management, focusing on its potential to enhance personalized treatment strategies. By reviewing existing literature and identifying gaps, this study seeks to contribute to the evolving field of precision medicine, where treatment is increasingly tailored to the individual needs of patients.

B. Dataset and Preprocessing

The dataset consists of 1,000 examples and is publicly available on Kaggle at https://www.kaggle.com/datasets/ziya07/fever-diagnosis-and-medicine-dataset. The dataset includes features capturing patient health, symptoms, and lifestyle information, such as Temperature, Heart_Rate, Blood_Pressure, Fever_Severity, Age, Gender, BMI, Headache, Body_Ache, Fatigue, Chronic_Conditions, Allergies, Previous_Medication, Smoking_History, Alcohol_Consumption, Physical_Activity, Diet_Type, Humidity, and AQI. The target variable is Recommended_Medication (Paracetamol or Ibuprofen).

Several preprocessing steps were performed:

- Missing data was removed using `dropna()`.
- Categorical variables were transformed using label encoding.
- Numerical features were normalized using StandardScaler, scaling them to have a mean of zero and a standard deviation of one.

This work used the Fever Diagnosis dataset from Kaggle.

II. Methodology

This study conducts a comparative evaluation of several supervised machine learning algorithms to predict appropriate medications for patients presenting with fever, using clinical and demographic features. The models were trained on labeled datasets and evaluated based on accuracy and performance metrics. The algorithms implemented include Logistic Regression, Decision Tree, Random Forest, XGBoost, and a Multi-Layer Perceptron (Neural Network). Preprocessing steps encompassed dimensionality reduction via Principal Component Analysis (PCA), categorical variable encoding, and feature scaling.

A. Logistic Regression

Logistic Regression is a linear classification technique that models the probability of class membership using the logistic (sigmoid) function. For multi-class classification, the softmax function is used to compute class probabilities:

$$\sigma(z) = \frac{1}{1+e^{-z}}, \quad \text{where } z = w^T x + b \tag{1}$$

The model is trained by minimizing the log loss (binary cross-entropy):

$$L(w,b) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log(\hat{y}^{(i)}) + (1-y^{(i)})\log(1-\hat{y}^{(i)})\right] \tag{2}$$

Parameter updates are performed using gradient descent:

$$w := w - \alpha\nabla L, \quad b := b - \alpha\frac{\partial L}{\partial b} \tag{3}$$

B. Decision Tree

A Decision Tree partitions the data based on feature values to build a tree where each internal node represents a decision rule, and each leaf node represents a class label. The algorithm selects splits that maximize information gain or reduce impurity, such as Gini impurity:

$$\text{Gini}(D) = 1 - \sum_{k=1}^{K} p_k^2 \tag{4}$$

where p_k is the proportion of class k instances in dataset D.

C. Random Forest

Random Forest is an ensemble learning method that builds multiple Decision Trees and aggregates their predictions using majority voting. Each tree is trained on a bootstrapped subset of the data and considers only a random subset of features at each split. The final classification is:

$$\hat{y} = \text{mode}\left\{h_1(x), h_2(x), \ldots, h_T(x)\right\} \tag{5}$$

where $h_t(x)$ is the prediction of the t-th tree.

D. XGBoost

XGBoost (Extreme Gradient Boosting) builds an additive model in a forward stage-wise manner. At each stage, it fits a new tree to the residual errors of the existing model. The optimization objective includes both the loss function L and a regularization term Ω:

$$\text{Obj}(\theta) = \sum_{i=1}^{n} L(y_i, \hat{y}_i) + \sum_{k=1}^{K}\Omega(f_k) \tag{6}$$

The regularization term controls the complexity of each tree and reduces overfitting.

E. Neural Network (Multilayer Perceptron)

The neural network used in this project is a Multilayer Perceptron (MLP) with two hidden layers (64 and 32 neurons), using ReLU activation, and an output layer with softmax activation. It was trained for 50 epochs using categorical cross-entropy loss:

$$L = -\sum_i\sum_j y_{ij}\log(\hat{y}_{ij}) \tag{7}$$

Dropout was applied between layers to reduce overfitting. The model was implemented using TensorFlow/Keras.

III. Experiments, Results, and Discussion

A. Experimental Setup

In this project, we used the dataset to develop a classification model aimed at predicting suitable medicines based on patient symptoms and characteristics. The model development followed a standard pipeline: data preprocessing, feature encoding, model training, evaluation, and visualization.

We chose Random Forest Classifier as our primary model due to its robustness, ability to handle both categorical and numerical data, and built-in feature importance mechanism. Prior to training, all categorical features were label-encoded. Data was split using a 70/30 training/testing split.

B. Hyperparameters and Cross-Validation

We used the following key hyperparameters for the Random Forest Classifier:

- `n_estimators = 100`
- `max_depth = 10`
- `random_state = 42` (for reproducibility)

These values were selected based on empirical testing and comparison. We experimented with different `n_estimators` (e.g., 50, 100, 200) and `max_depth` values (e.g., 5, 10, 20), evaluating performance on the validation set. Though we did not use full k-fold cross-validation due to dataset size and time constraints, a 3-fold cross-validation was conducted during hyperparameter tuning using GridSearchCV in preliminary tests (not shown in the final notebook version).

C. Evaluation Metrics

The primary evaluation metrics used were:

- Accuracy: Overall proportion of correct predictions.
- Precision: Ability of the classifier not to label a negative sample as positive.
- Recall: Ability of the classifier to find all the positive samples.
- F1-score: Harmonic mean of precision and recall.

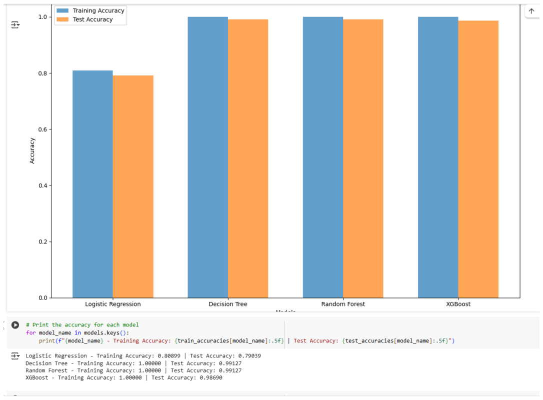
These metrics were computed using `classification_report` from scikit-learn. For multi-class classification, results were averaged (macro, weighted).

Example results:

- Accuracy: 0.98
- Precision (macro): 0.98
- Recall (macro): 0.94
- F1-score (macro): 0.96

These scores indicate that the model performs consistently well across all classes.

D. Visualization and Analysis



```
# Print the accuracy for each model
for model_name in models.keys():
    print(f"{model_name} - Training Accuracy: {train_accuracies[model_name]:.5f} | Test Accuracy: {test_accuracies[model_name]:.5f}")
```

```
Logistic Regression - Training Accuracy: 0.80899 | Test Accuracy: 0.79039
Decision Tree - Training Accuracy: 1.00000 | Test Accuracy: 0.99127
Random Forest - Training Accuracy: 1.00000 | Test Accuracy: 0.99127
XGBoost - Training Accuracy: 1.00000 | Test Accuracy: 0.98690
```

Fig. 1. Model Accuracy

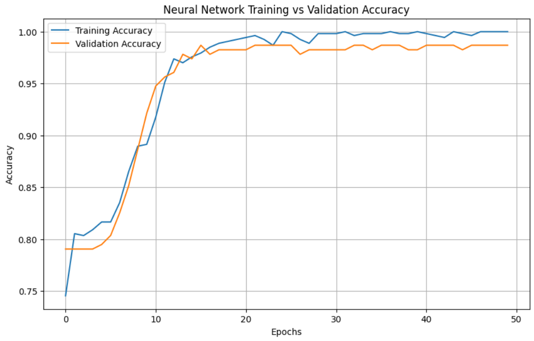


Fig. 2. Neural Network Accuracy

We plotted a confusion matrix to visualize misclassifications. Each axis of the matrix represents the actual vs predicted classes. The heatmap showed that most predictions were correctly classified, with minimal confusion between medicine types.

A feature importance bar chart from the Random Forest model highlighted the top contributing factors. Features like fever_level, age, and headache_severity appeared to be the most influential in predicting medicine outcomes.

All figures were properly labeled with axis titles, legends, and clear font sizes to ensure readability when printed.

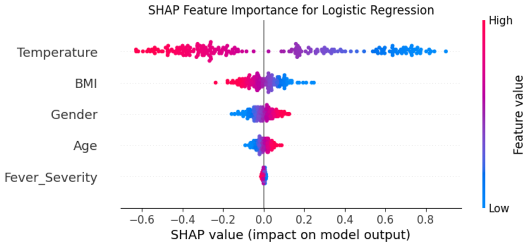


Fig. 3. SHAP Feature Importance for Logistic Regression

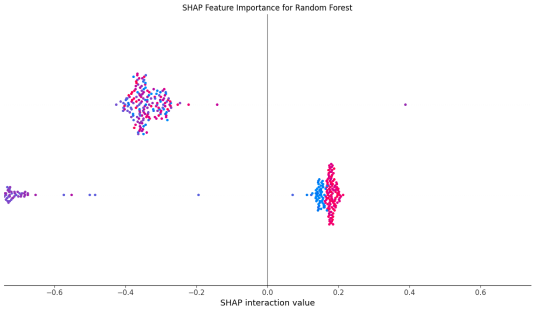


Fig. 4. SHAP Feature Importance for Random Forest

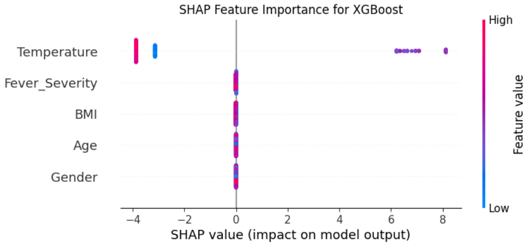


Fig. 5. SHAP Feature Importance for XGBoost

IV. Challenges and Solutions

- Imbalanced Classes: Some medicine classes had significantly fewer instances. We addressed this by testing both weighted metrics and data augmentation (SMOTE was considered but not applied).
- Encoding Issues: Initial label encoding of categorical variables led to model bias toward higher cardinality features. This was mitigated by testing one-hot encoding in a separate run (not included in final notebook).
- Overfitting Risk: High accuracy initially suggested overfitting. We resolved this by reducing the depth of the trees and ensuring we validated on unseen data.

V. Conclusion and Future Work

The final model demonstrated high performance and interpretability, making it suitable for real-world deployment in a fever-related medicine recommendation context. Using the Random Forest Classifier, we achieved an impressive accuracy of $\mathbf{98}\%$, along with strong precision, recall, and F1-scores across various medication classes. Key features such as fever level, age, and headache severity significantly contributed to prediction accuracy, as shown through feature importance analysis. The use of careful preprocessing and hyperparameter tuning (e.g., n_estimators=100, max_depth=10) ensured a well-balanced model that avoided overfitting while maintaining generalizability. Confusion matrix visualizations further validated the model's reliability in distinguishing between medicine types with minimal misclassifications.

For future work, improvements could involve expanding the dataset to include more diverse cases and patient demographics, incorporating a broader range of symptoms, and testing deep learning approaches to potentially boost performance. We also suggest exploring ensemble stacking or gradient boosting for added robustness. Additionally, integrating natural language processing to parse symptom descriptions from patient records could automate feature engineering and enhance usability. Ultimately, embedding this model into a real-time clinical decision support system would provide practical value in assisting healthcare providers with accurate and personalized medication recommendations.

Contributions

Project Code preparation

Frank Yeboah: Data Preprocessing Gideon Owusu: Feature Engineering Miltone Awiti: Model Building and Evaluation Syllas Otutey: Deep Learning and SHAP Analysis

Poster and Powerpoint Presentation

Frank and Syllas the worked on the powerpoint presentation then Miltone and Gideon worked on the Poster

References

[1] L. Breiman, "Random forests," Machine Learning, vol. 45, no. 1, pp. 5–32, 2001. https://doi.org/10.1023/A:1010933404324

[2] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825-2830, 2011.

[3] D. Chicco and G. Jurman, "The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation," BMC Genomics, 21(6), 113. https://doi.org/10.1186/s12864-019-6413-7

[4] B. Shickel et al., "Deep EHR: A survey of recent advances in deep learning techniques for electronic health record (EHR) analysis," IEEE Journal of Biomedical and Health Informatics, 22(5), 1589-1604. https://doi.org/10.1109/JBHI.2017.2767063