Building a Predictive Medical Diagnostic System Using Advanced Machine Learning Models

**Abstract**

In this report, I have developed a predictive medical diagnostic system using advanced machine learning models. This system aims to predict disease diagnosis based on symptoms, using Gradient Boosting, XGBoost, LightGBM, and a Voting Classifier. During building the models I addressed the correlation between symptoms and missing data. My pipeline includes data preprocessing, exploratory data analysis (EDA), and model evaluation using accuracy, precision, recall, F1-score, and ROC-AUC metrics. The outcomes demonstrate how well ensemble models work to increase diagnostic precision.

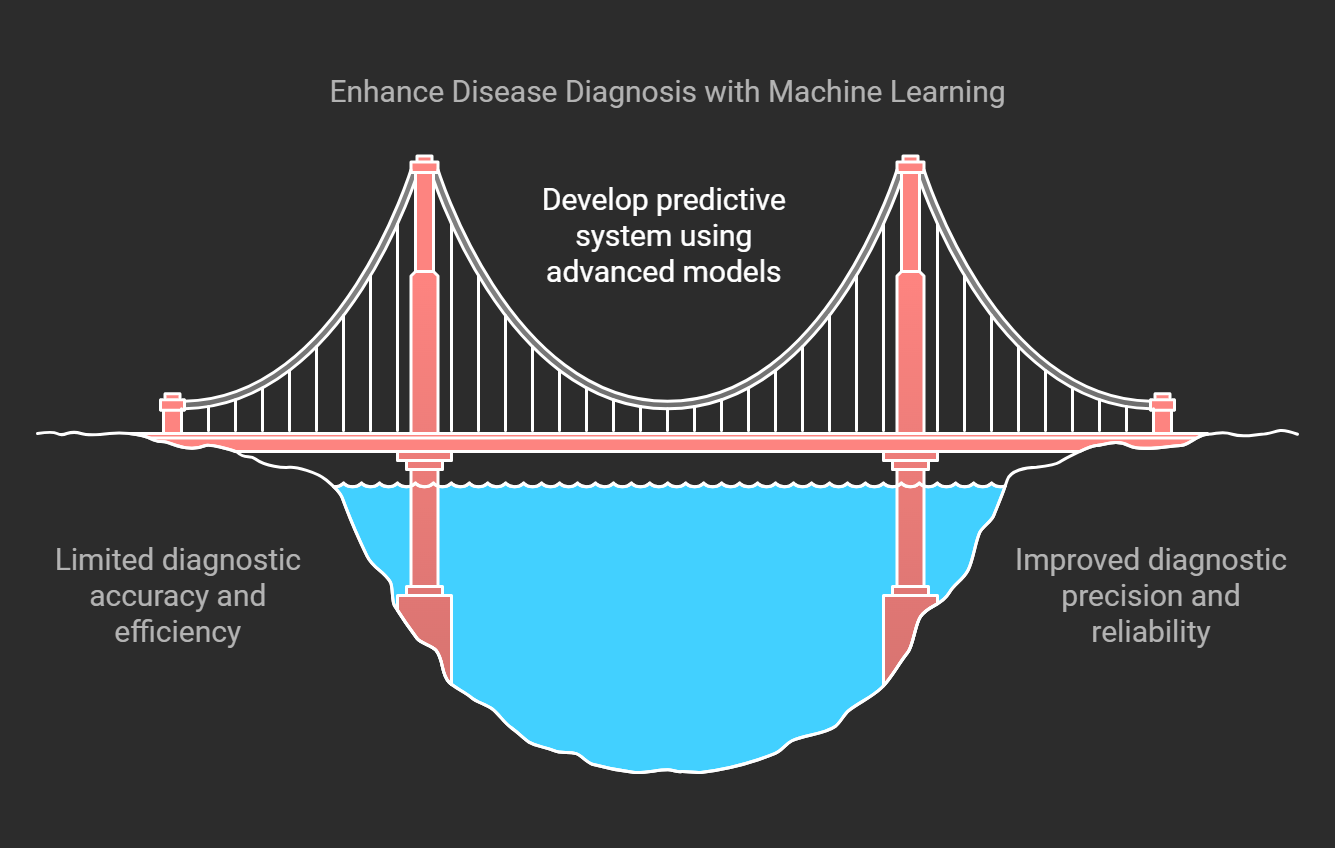
**I. Introduction**

**A. Problem Statement**

Manual diagnostic systems in healthcare are frequently delayed and inaccurate, which has an influence on patient outcomes. By using machine learning techniques to predict diseases based on patient symptoms, this initiative aims to automate the diagnostic procedure.

**B. Objectives**

1. Develop an end-to-end predictive analytics pipeline using advanced ML models.
2. Address class imbalance to ensure unbiased predictions in the dataset
3. Leverage advanced ensemble models for better diagnostic accuracy.

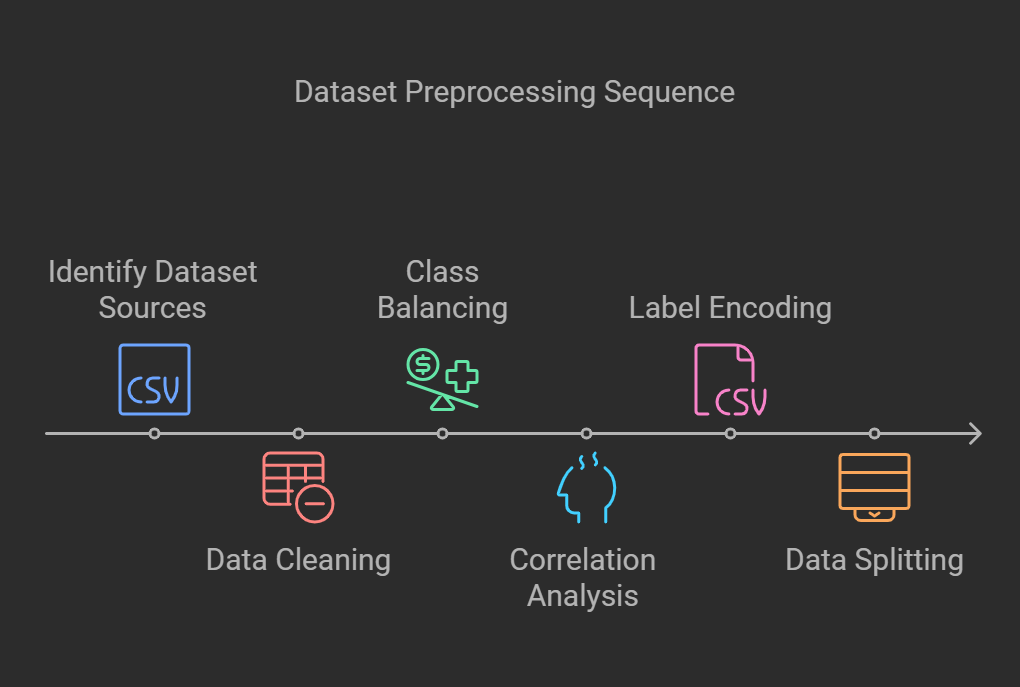


(figure 1)

**C. Significance**

Predictive Analysis in diagnostics can significantly enhance healthcare delivery by improving accuracy and timeliness. This project aims to bridge the gap between traditional and predictive medical diagnostic systems.

**II. Dataset and Preprocessing**

****

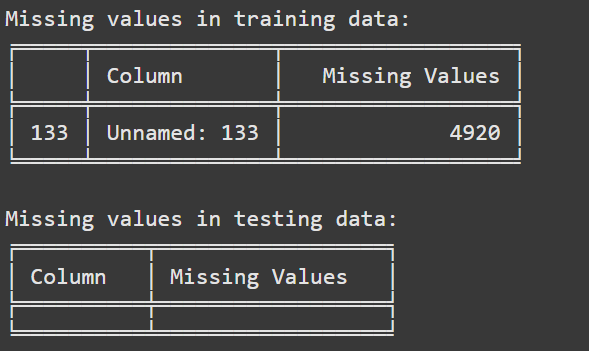
(figure 2)

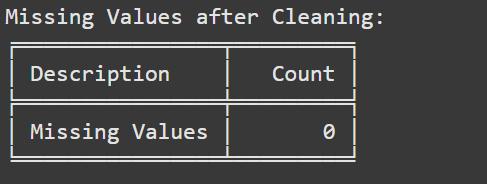
**A. Dataset Description**

* **Source**: I used two different files named 'DiseaseTraining.csv' & 'DiseaseTesting.csv' for testing and training the models. The training dataset comprised 132 symptoms and 41 diseases with 4920 patient records while the Testing dataset comprised 42 patients with the same number of symptoms.

**B. Preprocessing Steps**

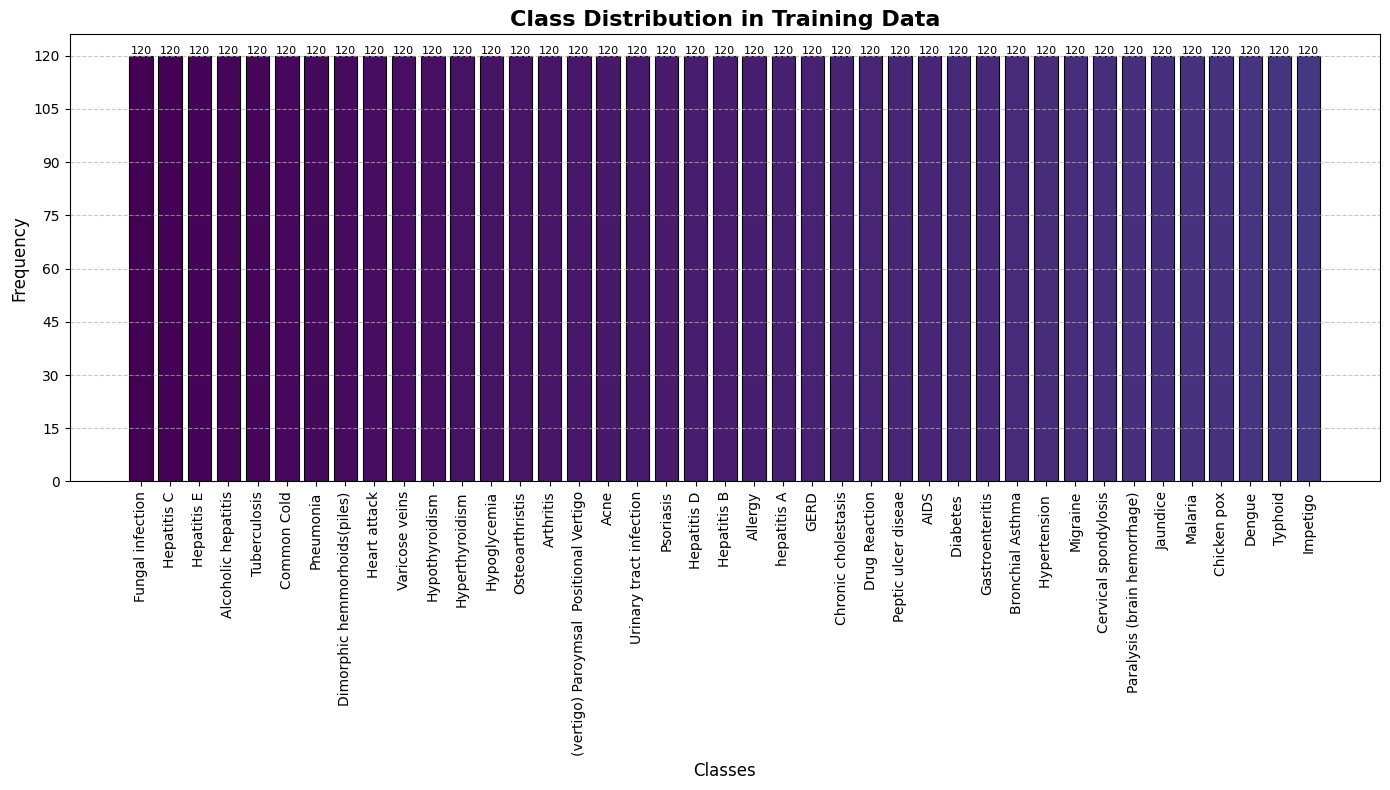
1. **Data Cleaning**: I looked for any missing values in the training and testing datasets. However, there are 4920 missing data in the "Unnamed: 133" column. This implies that it might be an unimportant or empty column, which ought to be eliminated.



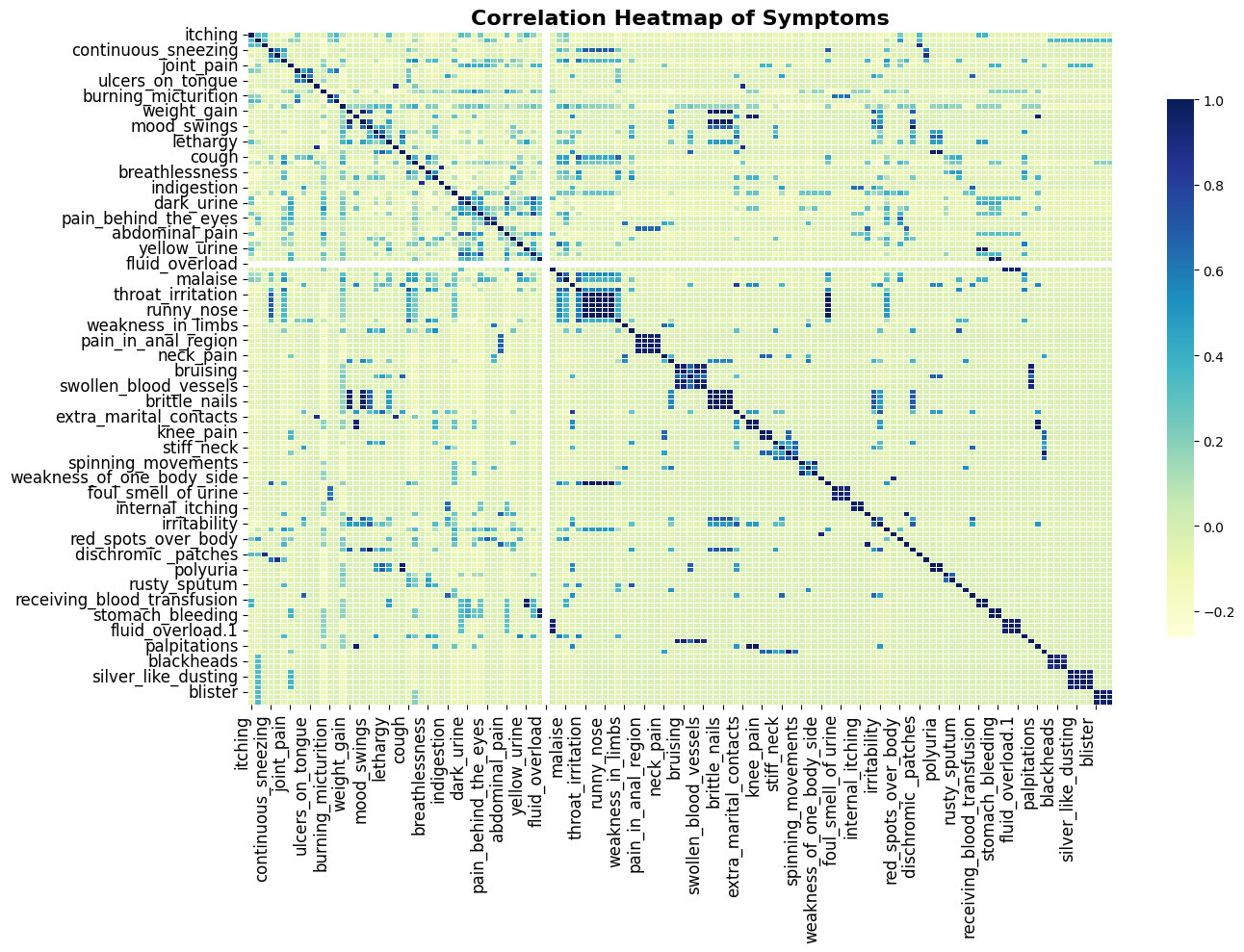


(figure 3)

1. **Balancing Classes**: Each disease class in the sample has precisely 120 cases, indicating a perfectly balanced class distribution. This suggests that methods of oversampling or under-sampling to rectify class imbalance are not required for this dataset.

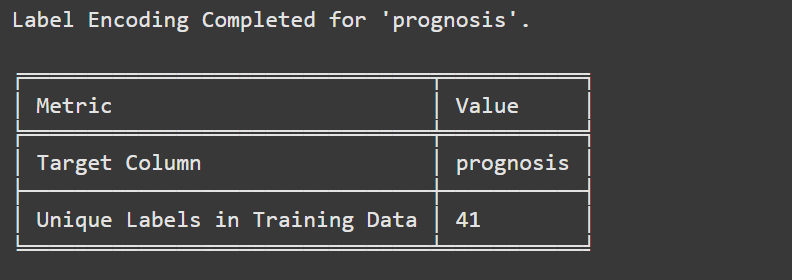


(figure 4)

1. **Correlation Analysis**: Correlations between symptoms (features) were found using a correlation heatmap. Since I am using a tree-based model XGBoost, it's not a big deal. It can handle correlations well by splitting the data into, decision rules, so we don't need to worry about removing correlated features. However, for the other models like SVM and Logistic Regression, we need to handle these correlations.
2. 

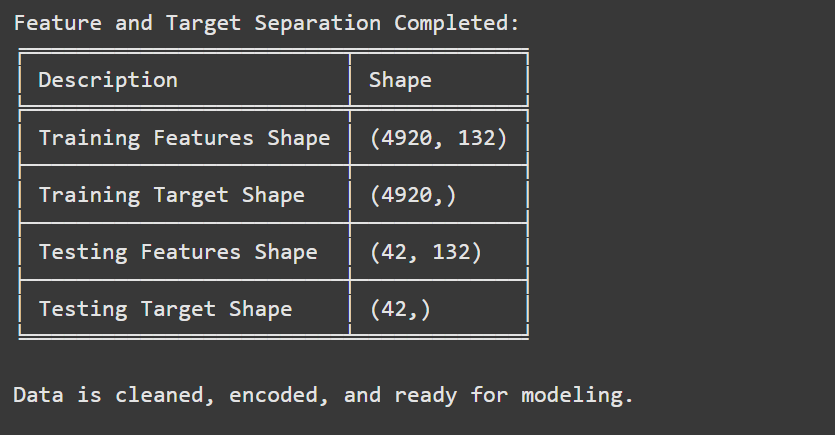
(figure 5)

1. **Label Encoding**: Encoding is always required if our target variable contains text or categorical data. We need to encode the target variable if it is not already numeric. In our dataset, the prognosis appears to be categorical, so the prognosis column has been successfully encoded in both the training and testing datasets.



(figure 6)

1. **Splitting the data**: Training and testing datasets has been created using ‘DiseaseTraining.csv’ and ‘DiseaseTesting.csv’.

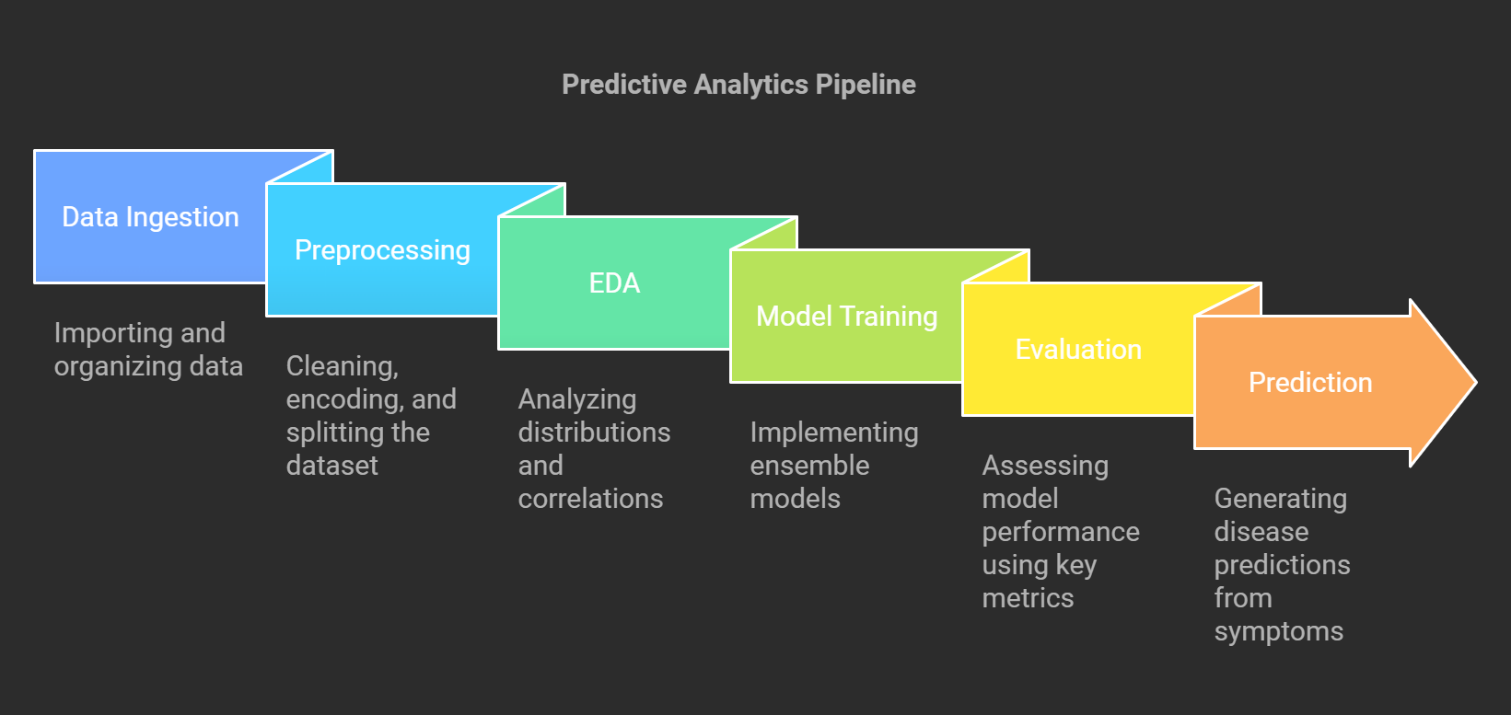


(figure 7)

**III. Methodology**

**A. Predictive Analytics Pipeline**

The pipeline consists of six stages:



(figure 8)

**B. Models and Algorithms**

In this project, various machine learning models were tested in order to configure an optimum model to solve my classification problem, each cast with its strengths and unique features. Here's how I approached each of these and the reasons behind my final decision.

1. **XGBoost**: XGBoost is a powerful algorithm for fast performing efficient gradient boosting. I selected this model as it performs excellently on larger datasets and handles missing values straight through the algorithm.
2. **Support Vector Machine (SVM):** It forms very precise decision boundaries even with very complicated data sets. I have reduced its dimensions by applying PCA as well as improving training efficiency using RandomSearchCV.
3. **Logistic Regression:** Easy interpretable models like logistic regression can serve as excellent baseline models against which to compare. Despite its simplicity, it performed perfectly on each metric. I have used ElasticNet regularization to deal with correlations and used RandomSearchCV to find the best hyperparameters.
4. **Voting Classifier:** The perfect ensemble method is created by such a voting classifier, combining multiple models. I have built this classifier with XGBoost, SVM, and Logistic Regression incorporated into the Voting Classifier so that it performs better with generalization.

**C. Evaluation Metrics**

The models were evaluated using the following metrics:

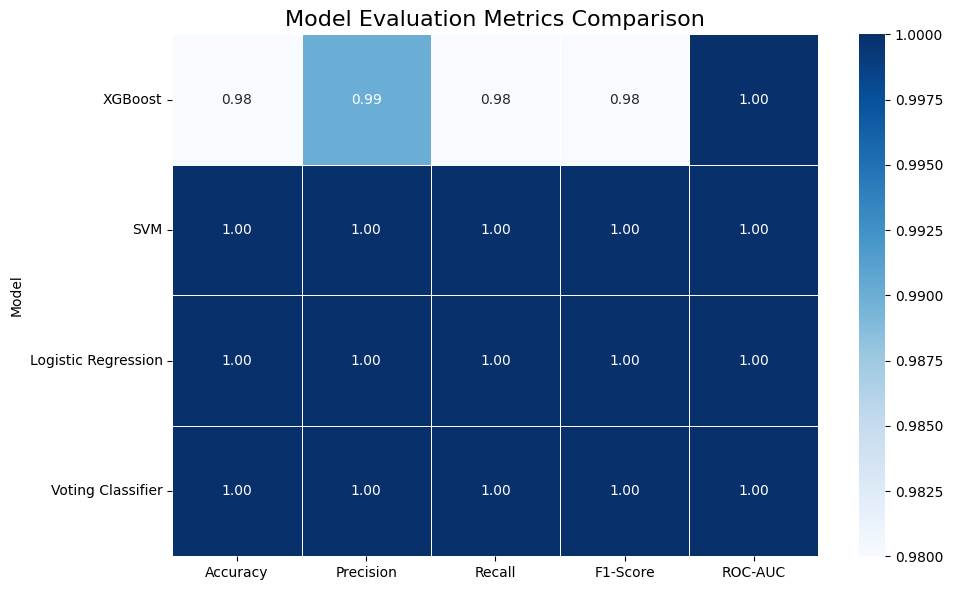
* Accuracy
* Precision
* Recall
* F1-Score
* ROC-AUC
* Training Time

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Model** | **Accuracy** | **Precision** | **Recall** | **F1-Score** | **ROC-AUC** | **Training Time (seconds)** |
| **XGBoost** | 0.98 | 0.99 | 0.98 | 0.98 | 1 | 0.9993 |
| **SVM** | 1 | 1 | 1 | 1 | 1 | 2.2554 |
| **Logistic Regression** | 1 | 1 | 1 | 1 | 1 | 9.9572 |
| **Voting Classifier** | 1 | 1 | 1 | 1 | 1 | 6.6139 |

(figure 9)

**IV. Results and Discussion**

**A. Performance Metrics**



(figure 10)



(figure 11)

**B. Insights**

* **Performance:**

SVM, Logistic Regression, and Voting Classifier performed similarly, A little better than XGBoost as they are the ones with the highest accuracy and other metrics for achieving the perfect score.

Regarding XGBoost, it is just a little below the others but still brought an impressive ROC-AUC score of 1.00.

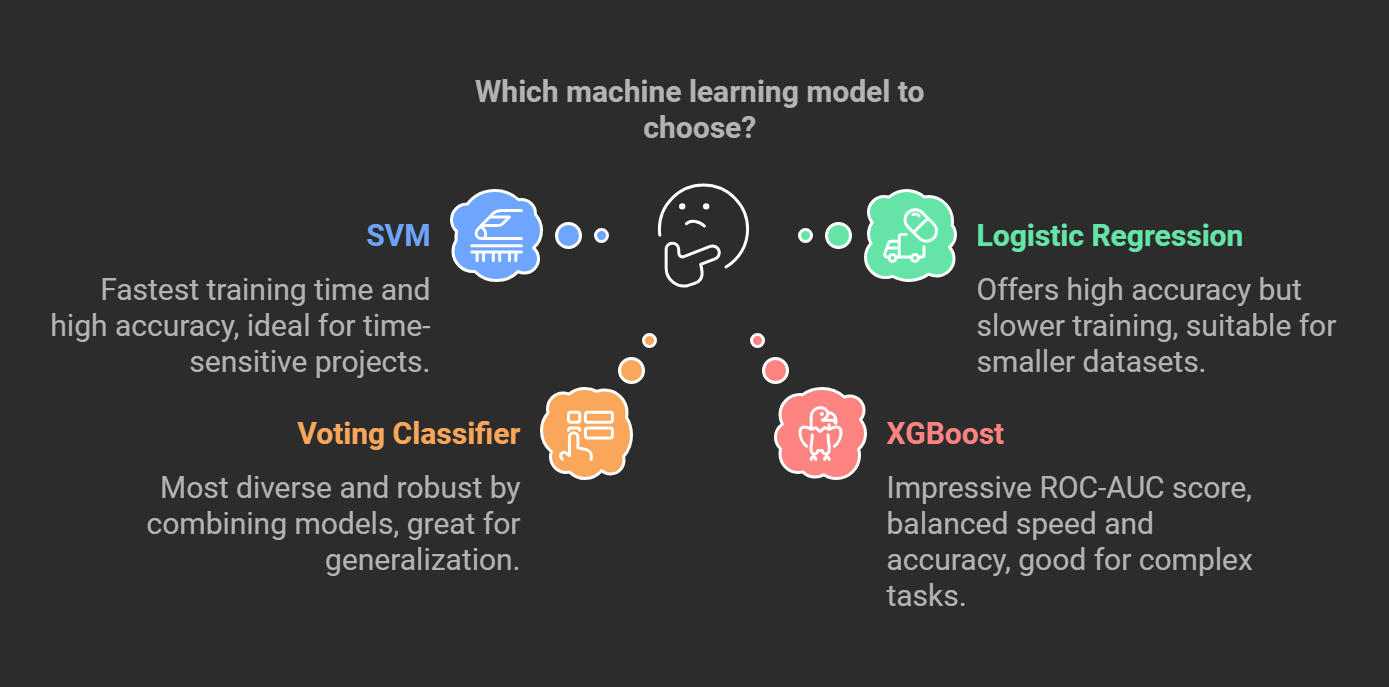
* **Time for training:**

SVM was the fastest, followed by XGBoost, making them the most time-efficient models while Logistic Regression and Voting Classifier took longer time to train, which could be not preferred for larger datasets.

* **Diversity:**

The Voting Classifier offers the most diversity by combining multiple models, making it the most robust and generalizable choice.

XGBoost provides a unique approach with its gradient-boosting algorithm, while SVM and Logistic Regression offer simpler, complementary strategies.



(figure 12)

**V. Conclusion**

**A. Summary**

This project successfully developed a predictive medical diagnostic system leveraging advanced ML models. The system demonstrated high diagnostic accuracy while effectively handling class imbalance.

**B. Future Work**

1. Experiment with deep learning techniques for improved accuracy.
2. Expand the dataset to include more diverse diseases and symptoms.
3. Investigate real-time deployment in clinical environments.

**References**

1. Scikit-learn (2024) *Support Vector Machines (SVM)*. Available at: <https://scikit-learn.org/stable/modules/svm.html> (Accessed: 11 December 2024).
2. Scikit-learn (2024) *Logistic Regression*. Available at: [https://scikit-learn.org/stable/modules/linear\_model.html](https://scikit-learn.org/stable/modules/linear_model.html%20) (Accessed: 11 December 2024).
3. Scikit-learn (2024) *Voting Classifier*. Available at: [https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.VotingClassifier.html](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.VotingClassifier.html%20) (Accessed: 11 December 2024).
4. Scikit-learn (2024) *RandomizedSearchCV*. Available at: [https://scikit-learn.org/stable/modules/generated/sklearn.model\_selection.RandomizedSearchCV.html](https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html%20) (Accessed: 11 December 2024).
5. Scikit-learn (2024) *Cross-validation: evaluating estimator performance*. Available at: [https://scikit-learn.org/stable/modules/cross\_validation.html](https://scikit-learn.org/stable/modules/cross_validation.html%20) (Accessed: 11 December 2024).
6. Scikit-learn (2024) *Model Evaluation and Selection*. Available at: <https://scikit-learn.org/stable/modules/model_evaluation.html> (Accessed: 11 December 2024).
7. Scikit-learn (2024) *Principal Component Analysis (PCA)*. Available at: <https://scikit-learn.org/stable/modules/decomposition.html> (Accessed: 11 December 2024).
8. Chen, T. and Guestrin, C. (2016) ‘XGBoost: A scalable tree boosting system’, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794. Available at: [https://dl.acm.org/doi/10.1145/2939672.2939785](https://dl.acm.org/doi/10.1145/2939672.2939785%20) (Accessed: 11 December 2024).