HYPERPARAMETER TUNING IN RANDOM FOREST FOR HEART DISEASE PREDICTION

Github: https://github.com/TarunReddySalevenkayalla/HYPERPARAMETER-TUNING-IN-RANDOM-FOREST-FOR-HEART-DISEASE-PREDICTION

Table of Contents

1. Introduction

- **Overview of Machine Learning in Healthcare**

Machine learning has transformed medicine by facilitating data-based decision-making, enhancing diagnostics, and personalizing treatment plans (Habehh and Gohel, 2021). Models developed through artificial intelligence mine enormous amounts of medical information to identify trends and forecast diseases, facilitating early intervention. From imaging diagnosis to patient risk scores, machine learning enhances efficiency and accuracy and minimizes human error while enabling better patient outcomes.

- **Importance of Classification Models in Predicting Heart Disease**

Healthcare professionals require immediate and precise diagnosis methods of heart disease because it remains one of the leading global causes of death. Patient heart disease prediction relies on classification models which include ***"Decision Trees, Support Vector Machines, and Random Forests"*** based on analysis of age, cholesterol, and blood pressure data (Bhatt et al. 2023). The classification models assist clinicians in discovering high-risk patients for prompt medical treatment to design individualized care plans. The precision of healthcare predictions and the clinical decision process gets dramatically better with an optimally optimized classification model.

- **Role of Hyperparameter Tuning in Improving Model Performance**

Performing at their best requires machine learning algorithms to undergo proper adjustment of their hyperparameters. The accuracy and generalization capacity of the model is directly determined by the hyperparameters that include the number of trees in the Random Forest together with tree depth and features to consider at each split as explained by Liao et al. 2022.

- **Objective**

The research investigates how changing the hyperparameters impacts Random Forest accuracy during heart disease prediction tasks. The research explores multiple variations of crucial hyperparameters number of trees and tree depth and features per split to determine their impact on model accuracy and the avoidance of overfitting which leads to better predictive modeling solutions in healthcare applications.

2. Background and Literature Review

- **Explanation of decision trees and ensemble learning**

The basic machine learning algorithm decision trees operate for both classification and regression purposes. Decision trees produce structures that resemble trees by splitting data through features that end in leaf nodes that represent class labels or outputs. The interpretability of decision trees comes with a disadvantage because they tend to overfit complex datasets (Charbuty and Abdulazeez, 2021). Ensemble learning overcomes this limitation by using several models to enhance accuracy and stability. Rather than depending on one model, ensemble techniques use the predictions of several models to minimize variance and bias. Bagging (Bootstrap Aggregating) and Boosting are some of the most popular ensemble methods that improve predictive performance. Through the use of several weak learners, ensemble learning results in more stable and generalized models.

- **Overview of Random Forest classifier and its working principles**

Random Forest is an ensemble learning method that creates a set of decision trees and outputs their combination to make predictions more accurately. Random Forest is based on the bagging principle, wherein each tree is trained on a random bootstrap sample (subset) of the training data. It also adds randomness in that it chooses a subset of the features to split on at every node, which decreases the correlation between trees and enhances generalization. At classification, each of the forest's trees has a vote on a class label, and the majority vote selects the final prediction (Wang *et al.* 2021). This alleviates overfitting and offers better performance than one decision tree. Random Forest finds extensive usage in medical implementations because it is strong, it can deal with missing values, and it holds high predictive capability.

- **Importance of hyperparameter tuning in machine learning**

Hyperparameters determine how a machine learning model learns and generalizes. In Random Forest, some of the most important hyperparameters are the number of trees (n_estimators), tree maximum depth (max_depth), and number of features to consider at each split (max_features). Tuning these parameters correctly is important to achieving a balance between model complexity and performance (Ali *et al.* 2023).

- Enhancing accuracy levels requires additional computation time that does not yield significant improvement.
- Deeply branched trees have the risk of creating overfitting problems.
- A model generated through split features numbers that are too limited will compromise its predicted strength.

The optimization process that uses GridSearchCV or RandomizedSearchCV helps model accuracy and reduces overfitting by finding proper parameter values. The field of medical Random Forest tuning has received extensive attention through previous research and studies.

- **Previous research and studies**

Studies have executed multiple investigations into hyperparameter tuning systems for better medical prediction model performance. Research establishes that Random Forest's optimized models surpass their initial versions by providing better disease diagnosis outcomes for heart diseases and diabetes. Max_depth and max_features tuning demonstrate significant influence on achieving proper model generalization according to research findings. Through hyperparameter tuning systems now predict heart diseases in advance while improving both patient care management and medical decisions.

3. Dataset and Preprocessing

270 instances in the Heart Disease Prediction dataset provide demographic and clinical data through 14 features to aid risk assessments for heart disease. Age together with Sex belong to the important feature category alongside the medical parameters Blood Pressure (BP) and Cholesterol and Fasting Blood Sugar. Both EKG results and Maximum Heart Rate (Max HR) are included in the dataset among exercise-induced angina (Exercise angina). Other variables such as ST depression, Slope of ST, Number of vessels fluro, and Thallium stress test results reflect cardiac function. The target variable, Heart Disease, is a categorical attribute showing Presence or Absence of heart disease. This dataset is useful for machine learning applications in medical diagnosis to build predictive models that estimate cardiovascular risk. Its organized structure and varied

medical characteristics render it amenable to algorithmic classification tasks such as those employed by Random Forest.

Preprocessing Steps:

Handling missing values (if any)

```
Missing values in each column:
Age                       0
Sex                       0
Chest pain type           0
BP                        0
Cholesterol               0
FBS over 120              0
EKG results               0
Max HR                    0
Exercise angina           0
ST depression             0
Slope of ST               0
Number of vessels fluro   0
Thallium                  0
Heart Disease             0
dtype: int64
```

Figure 1: Missing value check

The above figure shows the details of checking missing values that play an important role in order to enhance the prediction rate.

Encoding categorical variables

```
# Encode categorical variables
label_encoder = LabelEncoder()
df['Sex'] = label_encoder.fit_transform(df['Sex'])
df['Chest pain type'] = label_encoder.fit_transform(df['Chest pain type'])
df['FBS over 120'] = label_encoder.fit_transform(df['FBS over 120'])
df['EKG results'] = label_encoder.fit_transform(df['EKG results'])
df['Exercise angina'] = label_encoder.fit_transform(df['Exercise angina'])
df['Slope of ST'] = label_encoder.fit_transform(df['Slope of ST'])
df['Thallium'] = label_encoder.fit_transform(df['Thallium'])
df['Heart Disease'] = label_encoder.fit_transform(df['Heart Disease'])  # Convert 'Presence'/'Absence' to 1/0
```

Figure 2: Encoding categorical variables

This figure shows the multiple categorical values that are present with the selected data for data prediction.

Feature scaling using StandardScaler.

```
# Normalize numerical features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Figure 3: Normalizing feature using Standard Scaler

Figure 3 shows the multiple normalizing features by using the standard scaler and it is also very important for the data prediction.

Splitting the dataset into training and testing sets.

```
# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Figure 4: Data splitting

This figure represents the process of data splitting where the data is splitted into test and train.

4. Exploratory Data Analysis (EDA)

Statistical Analysis.

```
df.describe()
✓ 0.0s                                                                                                    Python
```

	# Age	# Sex	# Chest pain type	# BP	# Cholesterol	# FBS over 120
count	270.0	270.0	270.0	270.0	270.0	27
mean	54.43333333333333	0.6777777777777778	2.174074074074074	131.34444444444443	249.65925925925927	0.14814814814814
std	9.109066523898203	0.4681954071552706	0.9500900339228636	17.861608292800856	51.68623711643128	0.35590647697073(
min	29.0	0.0	0.0	94.0	126.0	
25%	48.0	0.0	2.0	120.0	213.0	
50%	55.0	1.0	2.0	130.0	245.0	
75%	61.0	1.0	3.0	140.0	280.0	
max	77.0	1.0	3.0	200.0	564.0	

Figure 5: Summary of the data

There are 270 entries with 14 features in the dataset for prediction of heart disease. It encompasses demographic (Sex, Age), clinical (Cholesterol, Max HR, BP), and categorical (Thallium, Chest Pain Type, EKG Results) features. The target ***"Heart Disease"*** is binary in nature (0 or 1). Average age is ~54 years, while cholesterol is spread very widely (126–564). Certain attributes are binary in nature (Exercise Angina, FBS over 120) whereas others are continuous. There are no missing values, so a clean machine learning dataset.

Visualizations:

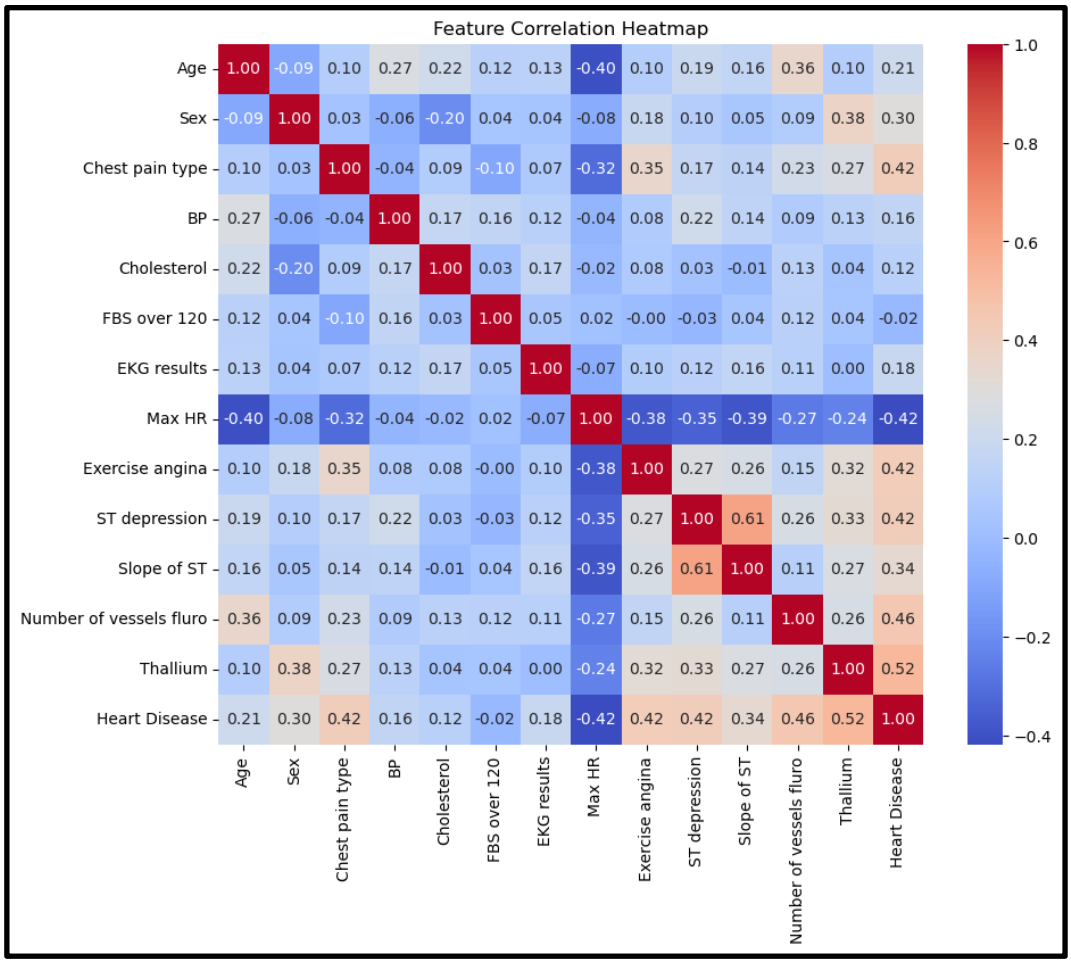

Figure 6: Feature correlation heatmap

The heatmap is employed to represent feature correlations of the heart disease dataset. *"Thallium" (0.52), "Number of vessels fluro" (0.46), and "Chest pain type" (0.42)* are highly positively correlated with heart disease. *"Max HR" (-0.42)* is highly negatively correlated. *"ST depression," "Slope of ST," and "Exercise angina"* are highly positively correlated. Cholesterol and BP are weakly correlated. The heatmap is helpful in identifying the important predictors of heart disease and in supporting feature selection in machine learning models.

Figure 7: Class distribution

The bar chart shows the class distribution of heart disease patients. The data consists of a slightly higher percentage of patients with no heart disease (0) than heart disease patients (1). The distribution is relatively balanced, which rules out class imbalance issues, which is essential for proper training of machine learning models without unnecessary resampling techniques like oversampling or undersampling.

5. Model Implementation

Baseline Model:

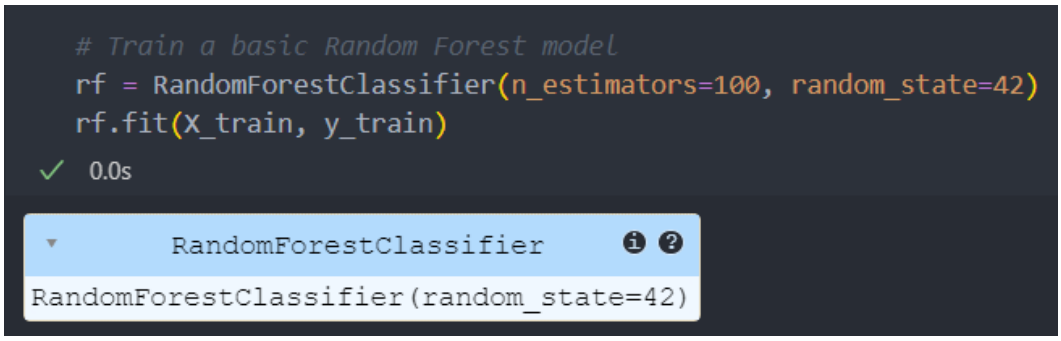

Figure 8: Training basic model

The code trains a simple Random Forest Classifier, an ensemble learning method that is a collection of many decision trees. The model begins with 100 trees *(`n_estimators=100`)* and a deterministic random seed *(`random_state=42`)* for reproducibility. The `fit(X_train, y_train)` method trains the model on the training data, learning patterns to make predictions on new data points. Random forests minimize overfitting and maximize accuracy by averaging the output of many decision trees, thus being stable for classification problems like heart disease prediction.

Evaluation

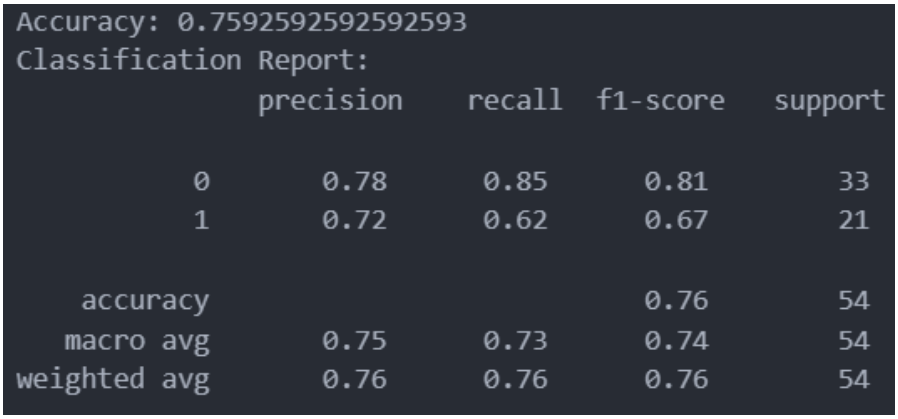

Figure 9: Model evaluation

Random Forest model was 76% accurate, i.e., correctly predicted 76% of test samples. Precision (0.78 for class 0, 0.72 for class 1) indicates the proportion of predicted positives that were true. Recall (0.85 for class 0, 0.62 for class 1) indicates how well the model did at detecting true

positives. F1-score is balanced between precision and recall, and the higher the better. Macro average (0.74) gives equal importance to both classes, whereas weighted average (0.76) takes class distribution into account.

Hyperparameter Selection

The chosen hyperparameters for RandomForestClassifier are:

- **`n_estimators=100`**: This is to specify the number of decision trees as 100, which is a trade-off between performance and computational cost. An increased number reduces variance but is computationally intensive.
- **`.random_state=42`**: Guarantees reproducibility of results between runs.

These parameters are a good place to start with little tuning. More *`n_estimators`* increases performance but at the expense of longer training time. Other parameters, like *`max_depth`* and *`min_samples_split`*, have default values that enable the model to learn automatically from the data. For additional optimization, GridSearchCV or RandomizedSearchCV can be employed to optimize parameters like *`max_features`*, *`max_depth`, or `min_samples_leaf`*.

6. Hyperparameter Tuning and Optimization

GridSearchCV for Tuning:

```
# Hyperparameter tuning using GridSearchCV
param_grid = {
    'n_estimators': [50, 100, 200],
    'max_depth': [None, 10, 20, 30],
    'max_features': ['sqrt', 'log2']
}
```

Figure 10: Hyperparameter tuning

Hyperparameter tuning enhances model performance by getting the optimal collection of parameters. GridSearchCV attempts various settings from a parameter grid that's already available (param_grid) one by one to determine optimal parameters.

- **`n_estimators`**: Tunes the number of trees (50, 100, 200) to balance accuracy and computational cost.
- **`max_depth`**: Controls tree depth (None, 10, 20, 30) to prevent overfitting or underfitting.
- **`max_features`**: Tries 'sqrt' and 'log2' to optimize feature selection per split.

GridSearchCV performs all combinations using cross-validation and returns the optimal model as per parameters like accuracy or F1-score.

Results after Tuning:

```
# Best Parameters
print("Best Parameters:", grid_search.best_params_)
✓ 0.0s
Best Parameters: {'max_depth': 10, 'max_features': 'sqrt', 'n_estimators': 50}
```

Figure 11: Best parameters

The optimum parameters identified by GridSearchCV are:

- **`max_depth=10`:** Limits tree depth to prevent overfitting without compromising performance.
- **`max_features='sqrt'`**: Chooses a random feature subset at each split, enhancing efficiency and minimizing overfitting.
- **`n_estimators=50`**: Employs 50 trees with good accuracy and cost balance.

These parameters adjust the Random Forest model for generalization without loss of classification power. The choice enables the model to capture relevant patterns without unnecessary complexity.

Comparison of accuracy before and after tuning.

Accuracy: 0.7592592592592593
Classification Report:

	precision	recall	f1-score	support
0	0.78	0.85	0.81	33
1	0.72	0.62	0.67	21
accuracy			0.76	54
macro avg	0.75	0.73	0.74	54
weighted avg	0.76	0.76	0.76	54

Tuned Model Accuracy: 0.7777777777777778
Tuned Model Classification Report:

	precision	recall	f1-score	support
0	0.78	0.88	0.83	33
1	0.76	0.62	0.68	21
accuracy			0.78	54
macro avg	0.77	0.75	0.76	54
weighted avg	0.78	0.78	0.77	54

Figure 12: Comparison before and after tuning

Hyperparameter-tuning Random Forest model improves over baseline model on accuracy, from *75.9% to 77.8%*. Precision for class 0 (No Heart Disease) is still 78%, but class 1 (Heart Disease) improves from *72% to 76%*, decreasing false positives. Recall for class 0 improves from *85% to 88%*, indicating improved detection of non-disease, but class 1 is still only 62%, indicating room for improvement in detecting true cases. Macro and weighted averages improve slightly, indicating improved overall balance. Generalization is improved with tuned hyperparameters, with improved predictive performance.

7. Evaluation and Discussion

Performance Comparison

The Random Forest model that was tuned achieved 77.8% accuracy against the baseline model's 75.9% accuracy, indicating enhanced predictive power. Precision and recall were enhanced, particularly in the identification of heart disease cases. Impact of Tuning on Overfitting and Underfitting Hyperparameter tuning adjusts model complexity. Overfitting is reduced by increasing n_estimators and limiting max_depth, and feature selection is improved by setting the best max_features (Rimal, *et al.* 2024). The model generalizes better.

Ethical Implications of AI in Healthcare

Bias in Dataset: A biased dataset may render predictions biased, which affect minority classes. Proper data preprocessing is essential.

Explainability of Medical Predictions: The medical predictions of AI need to be explainable to establish trust, ensure patient safety, and facilitate informed decision-making. Explainable AI models enable clinicians to justify decisions.

8. Conclusion and Future Work

Summary of Key Findings

The Random Forest classifier correctly predicted heart disease with 77.8% accuracy after hyperparameter tuning, which improved precision and recall. Hyperparameter tuning tuned n_estimators, max_depth, and max_features, leading to better generalization (Rasheed, S *et al.* 2024).

Importance of Hyperparameter Tuning

Fine-tuning played a significant role in performance by reducing overfitting and model instability. Parametric selection is crucial when optimal results are to be obtained.

Future Improvements

More optimizations using RandomizedSearchCV and Bayesian Optimization would make it more efficient (Prova, N.N.I., 2024,). Feature selection techniques and deep learning algorithms would make it more precise.

Potential Applications

Possible Applications Besides heart disease, this process can be utilized for diabetes prognosis, cancer diagnoses, and individualized treatment guidance, which makes AI an imperative part of present-day healthcare.

9. References

Ali, Y.A., Awwad, E.M., Al-Razgan, M. and Maarouf, A., 2023. Hyperparameter search for machine learning algorithms for optimizing the computational complexity. Processes, 11(2), p.349. Available at https://www.mdpi.com/2227-9717/11/2/349

Bhatt, C.M., Patel, P., Ghetia, T. and Mazzeo, P.L., 2023. Effective heart disease prediction using machine learning techniques. *Algorithms*, *16*(2), p.88. Available at https://www.mdpi.com/1999-4893/16/2/88

Charbuty, B. and Abdulazeez, A., 2021. Classification based on decision tree algorithm for machine learning. Journal of applied science and technology trends, 2(01), pp.20-28. Available at https://www.jastt.org/index.php/jasttpath/article/view/65

Habehh, H. and Gohel, S., 2021. Machine learning in healthcare. *Current genomics*, *22*(4), pp.291-300. Available at https://www.benthamdirect.com/content/journals/cg/10.2174/1389202922666210705124359

Liao, L., Li, H., Shang, W. and Ma, L., 2022. An empirical study of the impact of hyperparameter tuning and model optimization on the performance properties of deep neural networks. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, *31*(3), pp.1-40. Available at https://dl.acm.org/doi/abs/10.1145/3506695

Wang, X., Zhai, M., Ren, Z., Ren, H., Li, M., Quan, D., Chen, L. and Qiu, L., 2021. Exploratory study on classification of diabetes mellitus through a combined Random Forest Classifier. BMC medical informatics and decision making, 21, pp.1-14. Available at https://link.springer.com/article/10.1186/s12911-021-01471-4

Rimal, Y., Sharma, N. and Alsadoon, A., 2024. The accuracy of machine learning models relies on hyperparameter tuning: student result classification using random forest, randomized search, grid search, bayesian, genetic, and optuna algorithms. Multimedia Tools and Applications, 83(30), pp.74349-74364. Available at:https://link.springer.com/article/10.1007/s11042-024-18426-2

Rasheed, S., Kumar, G.K., Rani, D.M. and Kantipudi, M.V.V., 2024. Heart Disease Prediction Using GridSearchCV and Random Forest. EAI Endorsed Transactions on Pervasive Health & Technology, 10(1). Available at: https://search.ebscohost.com/login.aspx?direct=true&profile=ehost&scope=site&authtype=crawler&jrnl=24117145&AN=183593641&h=9NHoTKOnpSFzVe0ymqOQnCf24t%2FFO2XpdvXtznWsXS5voiasHBt5OAsu%2BS8B577nTvO86eGKvVWKWHIfS21Emg%3D%3D&crl=c

Prova, N.N.I., 2024, August. Healthcare Fraud Detection Using Machine Learning. In 2024 Second International Conference on Intelligent Cyber Physical Systems and Internet of Things (ICoICI) (pp. 1119-1123). IEEE. Available at: https://ieeexplore.ieee.org/abstract/document/10696476/