A Comparison of Naïve Bayes and Random Forest on Predicting Liver Disease

Name: Sajeel Nadeem Alam **Email**: sajeel.alam@city.ac.uk **Coursework**: INM431 Machine Learning

Description and Motivation

The goal of this study is to conduct a fair comparison between two probabilistic machine algorithms: Naïve Bayes and Random Forest. For this purpose, both algorithms will be tasked to solve a binary classification when applied to the UCI Indian Liver Patient Dataset [1] to determine if a patient has liver disease or not. They will follow the same methodology and will be evaluated using identical evaluation metrics. The results will be compared to existing literature using the same dataset to validate the findings of this study and draw meaningful conclusions. The motivation is to critically analyse why one algorithm may outperform the other.

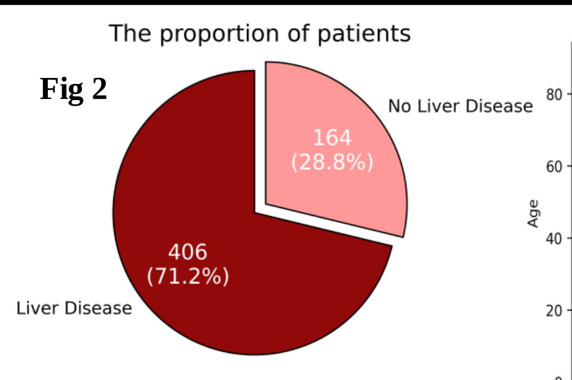
Exploratory Data Analysis

- The dataset consists of 583 rows and and 11 columns; 10 are predictor features and 1 is a binary class column (Fig 1) which contains 2 (changed to 0) if the patient does not have liver disease and 1 if they do have liver disease
- Two predictors are Age and Gender (encoded to numerical binary) and rest are continuous medical indicators
- Has 4 rows with missing values in A_GRatio that are filled with the mean and 13 duplicate rows that are removed [2]
- According to Fig. 2, there are 164 patients without liver disease (0) and 406 with liver disease (1) resulting in a 30:70 ratio which is not severely unbalanced hence this study does not apply any balancing techniques
- TB, DB, Alkphos, Sgpt, Sgot have severe outliers hence Z-scores are used to detect them and they are eliminated from the training data [3]
- Fig 3 shows the difference in distributions of the predictor variables in the classes without the effect of outliers. The mean for Age, TB, DB, Alkphos, Sgpt, Sgot is slightly higher in patients with liver disease whereas for TB, ALB and A_GRatio it is slightly lower
- Fig 4 shows high correlations between TB/DB, Sgpt/Sgot, TP/ALB and ALB/A_GRatio. TB, DB, Alkphos, Sgpt and Sgot are positively correlated with the Selector whereas ALB and A_GRatio have a negative correlation

Age	Age of the patient
Gender	Gender of the patient
TB	Total Bilirubin
DB	Direct Bilirubin
Alkphos	Alkaline Phosphotase
Sgpt	Alamine Aminotransferase
Sgot	Aspartate Aminotransferase
TP	Total Proteins
ALB	Albumin
A/G Ratio	Albumin and Globulin Ratio
Selector	Target

Fig 1

Fig 2



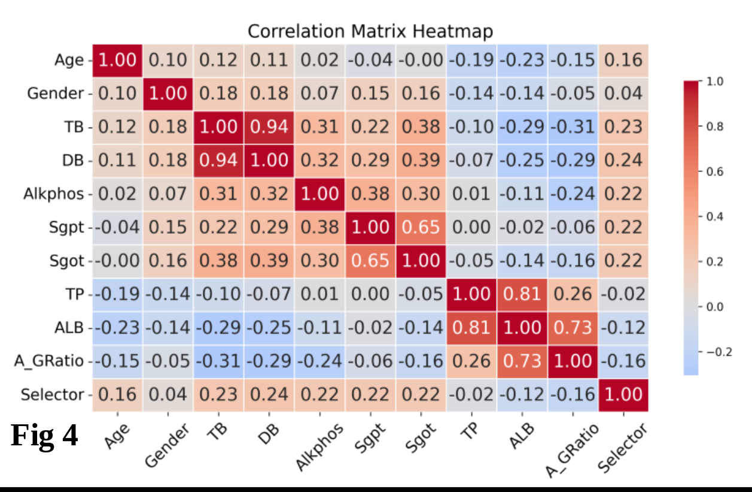


Fig 4

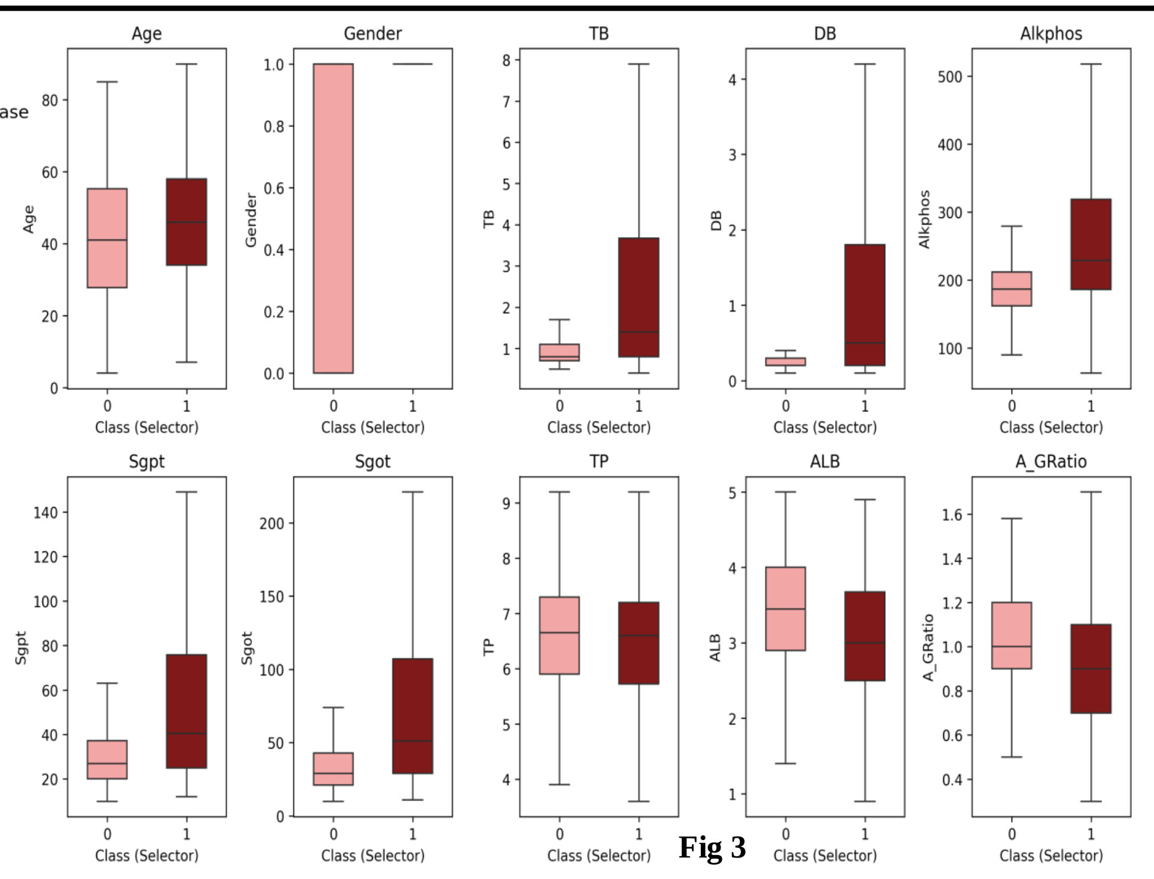


Fig 3

Naïve Bayes

- Naïve Bayes is a probabilistic classification model which uses Bayes Theoroem to calculate the conditional probability of a class C_k given a datapoint x i.e. $P(C_k|x)$
- It is called 'naïve' because it assumes the predictors are independent of each other given the class variable [4]
- The algorithm calculates the prior class probabilities $P(C_k)$ from the data
- For discrete predictors, it uses a multinomial distribution and for continuous predictors it assumes a Gaussian distribution to calculate the conditional probabilites $(P(x_i|C_k)$ however, this can be changed to using KDE [4]
- Using the chain rule, it calculates the conditional probability of a datapoint *x* belonging to a class C_k for all the classes and uses a decision rule such as argmax to select the class with the highest conditional probability

Advantages

- Simple and efficient model; can be trained quickly
- It is robust as it performs well irrespective of violation of assumptions and missing values [4]
- Needs less hyperparameters to be estimated [4]
- Outperforms most models with small datasets [4]

Disadvantages

- Can have poor performance if features are not independent [4]
- Not reliable in the case of zero observations problem [4]
- Vulnerable to Bayesian poisoning [4]
- Sensitive to how input data is prepared [5]

Random Forests

- Random Forest is an ensemble technique which uses the results of multiple decision trees to give its output [6]
- It uses bootstrapping with replacement to create training sets for each tree. The predictors used for splitting are also selected at random. The combination of boostrapping and aggregating results is termed as "Bagging"
- Each individual tree is trained on its subset and validated using out of bag (OOB) values that were not in its training set
- In each decision tree, the data is split into branches using a threshold that is determined using information gain
- For predictions in classification, it selects the class with the highest count across all trees whereas for regression it adds the gaussians of all the leaf nodes (one from each tree) and selects the maximum value

Advantages

- Avoids overfitting and good at generalisation [7]
- Robust against missing values and outliers [7]
- Feature importance is measured during training [7]
- It is capable of both classification and regression and can be used for a varity of applications [7]

Disadvantages

- Lack of interpretability
- Computationally expensive in terms of training time
- High memory usage to store data for each tree
- Can be prone to overfitting if data has noise in it resulting in poor generalisation in some cases

Hypothesis Statement

- According to existing literature, it is expected that amongst the two methods, Random Forests will outperform Naïve Bayes due to its ability to prevent overfittng and generalize better [8]
- Naïve Bayes will improve its performance if it is accurately able to estimate the probability distributions of the predictors instead of assuming them to be normally distributed
- Random Forests will have a longer training time as it has to aggregate the results of all the decision trees to generate an output

Methodology

1. The data is divided into train and test sets using a 70:30 ratio [9]. The test set is kept separate
2. Feature selection is done on the training set. Features that have a correlation above a threshold (1.5 for this study) with the target class are selected. Out of the feature pairs that are highly correlated, only one is kept
3. Both models are trained on the same training set with and without feature selection to compare results
4. Grid search is performed to fine tune selected hyperparameters
5. 10-fold cross validation is implemented to get robust evaluation metrics for each set of hyperparameters [9]
6. For each set of hyperparameters the validation accuracy, precision, recall, F1 score and AUC is calculated
7. Training times for both Naïve Bayes and Random Forest are recorded
8. The set of hyperparameters that gives the lowest cross fold validation loss is selected
9. The models are retrained on the entire training data using their respective best set of hyperparameters
10. Both models are run on the test set and the predictions are compared with the true labels
11. Evaluation metrics such as test accuracy, precision, recall, F1 score and AUC are compared for both models

Experimental Results and Parameter Choices

Naïve Bayes

- The hyperparameters that were fine-tuned are prior class probabilties, predictor probability distribution and the kernel wdith in the case of KDE. These help improve performance if there is a class imbalance in the training set and if the probability distributions of the predictors are not normally distributed
- Hyperparameters were first optimized automatically using Bayesian Optimization to get an idea of which values perform better than the others. These values were used as a starting point for further fine-tuning
- During a grid search, the set of hyperparameters that gave the lowest cross-fold validation loss (without feature selection) were Prior: 'empirical', Distribution: 'kernel' and kernel width: 0.5 (Fig. 5)
- The model was then re-trained on the entire training set and the average training time recorded was 0.02 seconds

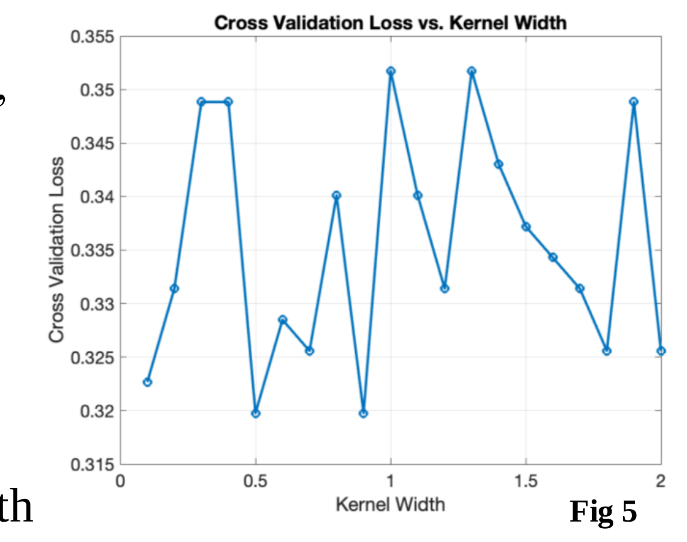


Fig 5

Random Forest

- For Random Forest, the hyperparameters to be fine-tuned are number of learning cycles (trees), maximum number of splits in each tree and the number of variables to consider at each split. These helps to control the depth of each tree and experiment with using different number of features for splitting
- For a fair comparison, the hyperparamters were optimized using Bayesian Optimization which were then used as a starting point for further fine-tuning
- In the grid search, the set of hyperparamters that gave the lowest cross-fold validation loss (without feature selection) were NumLearningCycles: 60 (Fig 6), MaxNumberSplits: 140 and NumVariablesToSample: 1
- The model was re-trained on the entire training set and the average training time was recorded as 0.11 seconds

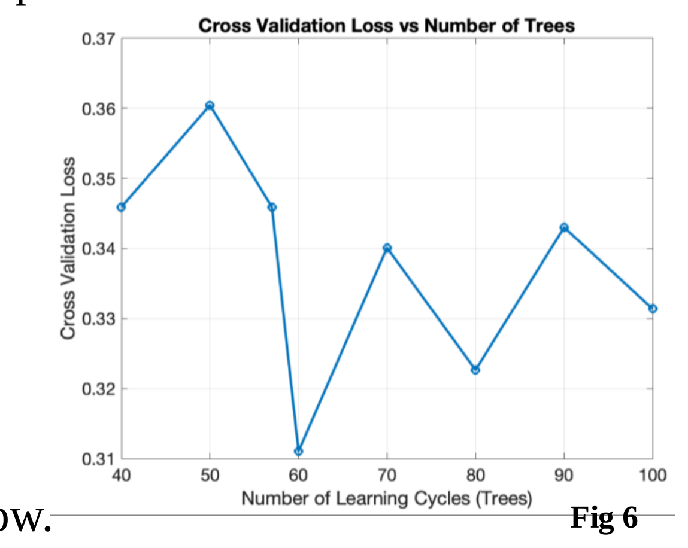


Fig 6

Results

- Both trained models made predictions on the test set and their predictions were compared to the true labels. The evaluation metrics can be viewed below.

Model	Validation Loss	Training time (s)	Test Accuracy (with FS)	Test Accuracy (w/o FS)	Precision (w/o FS)	Recall (w/o FS)	F1 Score (w/o FS)	AUC (w/o FS)
NB	0.32	0.02	64.33	70.76	78.74	81.30	80.00	0.64
RF	0.31	0.11	67.84	76.61	78.23	93.50	85.19	0.74

Fig 7

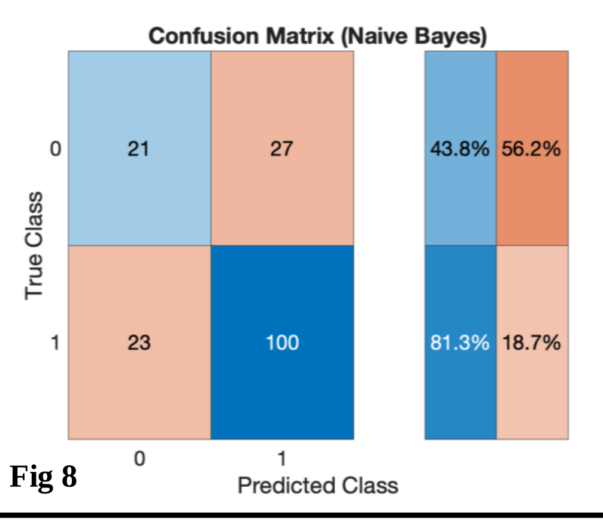


Fig 8

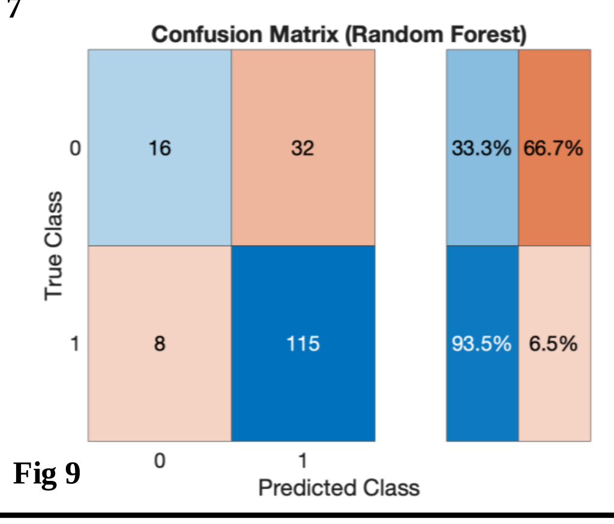


Fig 9

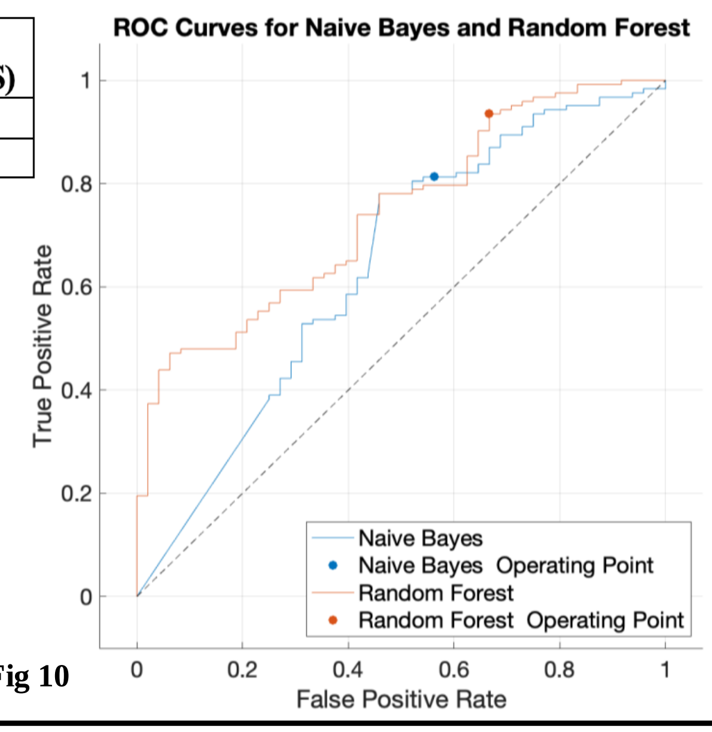


Fig 10

Analysis and Critical Evaluation of Results

- Figure 7 shows that Naïve Bayes has a shorter training time (0.02 seconds) as compared to Random Forest (0.11 seconds) which agrees with the third point of the hypothesis statement. The reason being that Naïve Bayes assumes feature independence hence has fewer and simpler probability calculations making it efficient [5] whereas, Random Forest trains multiple decision trees and each tree has it owns calculations related to splitting resulting in a longer training time
- Both models have similar lowest cross fold validation losses which shows that they perform identical in recognising patterns in the validation set. The expectation was that Naïve Bayes would have a poorer performance as it has a high bias however, the hyperparameters chosen improves its performance
- For both models, the test accuracy with feature selection (FS) is less as compared to without feature selection (w/o FS). The expectation was that with feature selection, Naïve Bayes would perform better since highly correlated features are removed however, correlation does not imply dependency as seen by the results. Random Forest was expected to perform the same since it uses the feature with the highest information gain at each split making it robust against irrelavant features. However, in this study the RF model considers only 1 variable while splitting (due to hyperparameter fine-tuning) and it is possible that feature selection removed features that were contributing to the predictive power of both models thus resulting in a decrease in accuracy
- Without feature selection, Random Forest outperforms Naïve Bayes with a test accuracy of 76.62 as opposed to Naïve Bayes' accuracy of 70.76. This agrees with the first point of the hypothesis statement. The reason being that Random Forest implements bagging to train each tree on a random subset of the data and uses a random selection of features at each split. This reduces the chance of the individual trees overfitting to the training data [3]. Moreover, Random Forest aggregates the results of multiples trees which reduces variance. This combination of reduced overfitting and low variance allows Random Forest to generalize better to unseen data as compared to Naïve Bayes
- Naïve Bayes also has low variance but high bias causing it to underfit sometimes. On other hand, the Random Forest model in this study has 57 trees and each tree can have maximum 140 splits allowing them to grow deep and learn patterns in the training set resulting in low bias. This may cause them to overfit however, since each tree is trained on a separate random subset and 1 random feature is considered at each split, the generated trees are random and thus the overall model has less chances of overfitting and reduced variance [7]. This balance of bias and variance allows Random Forest to perform better than Naive Bayes in terms of accuracy
- In this study, the test accuracy of Random Forest is slightly higher as compared to existing works however, the test accuracy of Naïve Bayes is significantly higher [6] [9]. This can be because of the hyperparameters of Naïve Bayes. Instead of assuming prior class probabilities to be uniform, the model sets them as the class relative frequencies in the training data when the Prior hyperparamter is set to 'empirical'. This improves model performance in the case of unbalanced data. With the Distribution hyperparameter set to 'kernel', instead of assuming the predictor probability distributions to be normal, the model estimates their probability distributions using kernel density estimation. Setting an appropriate kernel width allows the model to accurately estimate the the probability distributions of the predictors given the class, and with better estimates of probability values, Naïve Bayes is able to classify datapoints more accurately
- Since the dataset is unbalanced it is essential to consider evaluation metrics apart from the accuracy as well
- Both models have similar precision however, Random Forest has a significantly higher recall thus resulting in a higher F1 Score. This means that Random Forest is better at identifying true positives. This is because Naïve Bayes assumes conditional independence between features, failing to learn their interactions and relies heavily on the assumption that the probabilities estimated from the training set will hold true in the test set which may not always be the case. On the other hand, due to the aggregated result of multiple complex decision trees, Random Forest is able to capture even the faintest patterns in the both classes resulting in a higher recall
- This also explains why Random Forest is better at distinguishing between postive and negative classes resulting in a higher AUC and a slightly better ROC curve as illustrated in Fig 10

Lesssons Learned

- Even if the 'naïve' assumption does not hold true, Naïve Bayes performs well if it is given the correct hyperparameters especially for prior probabilities and distribution of predictor probabilities
- Feature selection using just univariate correlation is not effective enough as it does not capture the complex relationship between features and how interact with each other which is essential in high dimensional data

Future Work

- Use better feature selection methods such as the minimum redundancy maximum revelance (MRMR) algorithm. This may increase the performance of Naïve Bayes as it will only predict using relevant features
- Balance the dataset using balancing techniques such as SMOTE. This may also improve the performance of Naïve Bayes as the prior class probabilities will be uniform
- Instead of removing outliers using Z-scores, domain knowledge can be used to check which outliers are errors and which ones are edge cases that offer unique and signifiant information about the patterns in the data

1. "ILPD (Indian liver patient dataset)", UCI Machine Learning Repository
2. U. Sinthuja, V. Hatti, and S. Thavamani, "Analysis and prediction of liver disease for the patients in India using various machine learning algorithms," *Lecture Notes on Data Engineering and Communications Technologies*, pp. 433–442, Mar. 2022.
3. K. Gupta, N. Jiwani, N. Afreen, and D. D, "Liver disease prediction using machine learning classification techniques," *2022 IEEE 11th International Conference on Communication Systems and Network Technologies (CSNT)*, pp. 221–226, Apr. 2022.
4. I. Wickramasinghe and H. Kalutarage, "Naive bayes: Applications, variations and vulnerabilities: A review of literature with code snippets for implementation," *Soft Computing*, vol. 25, no. 3, pp. 2277–2293, Sep. 2020. doi:10.1007/s00500-020-05297-6
5. N. Kalcheva, M. Todorova, and G. Marinova, "Naive Bayes classifier, decision tree and ADABOOST ensemble algorithm – advantages and disadvantages," *International Scientific Conference ERAZ - Knowledge Based Sustainable Development*, pp. 153–157, Jan. 2020
6. D. K. Choubey, P. Dubey, B. P. Tewari, M. Ojha, and J. Kumar, "Prediction of liver disease using soft computing and data science approaches", *6G Enabled Fog Computing in IoT*, pp. 183–213, Oct. 2023. doi:10.1007/978-3-031-30101-8_8
7. J. Ali, R. Khan, N. Ahmad, and I. Maqsood, "Random Forests and Decision Trees", *International Journal of Computer Science Issues (IJCSI)*, vol. 9, Sep. 2012
8. V. Singh, M. K. Gourisaria, and H. Das, "Performance analysis of machine learning algorithms for prediction of liver disease", *2021 IEEE 4th International Conference on Computing, Power and Communication Technologies (GUCON)*, pp. 1–7, Sep. 2021. doi:10.1109/gucon50781.2021.9573803
9. R. Bhardwaj, R. Mehta, and P. Ramani, "A comparative study of classification algorithms for predicting liver disorders", *Lecture Notes in Electrical Engineering*, pp. 753–760, Dec. 2019. doi:10.1007/978-981-15-0214-9_78