Project Proposal 4: Heart Disease Data Set

Ricardo Pinto rjsp15@ua.pt, Inês Castro icbaptista@ua.pt
Department of Electronics, Telecommunications and Informatics (DETI)
University of Aveiro, Portugal

Course: Machine Learning Topics, Course Instructor: Prof. Pétia Georgieva
Workload per Student (50/50): Ricardo's contributions include data analysis, Logistic Regression, and K-Nearest Neighbors (KNN). Inês, on the other hand, focused on Neural Networks, Support Vector Machines (SVM), and Random Forest.

***Abstract*—Heart disease is one of the leading causes of death globally. Early detection and accurate diagnosis of heart disease can help prevent further complications and improve treatment outcomes. Machine learning models can be trained on large datasets to accurately predict the presence or absence of heart disease in patients. In this project, we will use and analyse the Cleveland heart disease dataset to build ML models with good accuracy, as well as a comprehensive evaluation of them.**

***Index Terms*—machine-learning, heart disease, classification, prediction, logisitic regression, neural networks, random forest, support vector machine, k nearest neighbours**

I. Introduction

In this work, 5 ML models were used on the Cleveland heart disease dataset to predict presence or absence of heart disease. They are: logistic regression, neural network, SVM, random forest and K nearest neighbours. We start by analysing the dataset, then selecting features, then encode and normalize them. After that, we compare our models to eachother and also compare them to other works on this dataset (references). Finally, we explore in more detail the models we trained. Techniques such as 5-fold cross validation, hyperparameters tuning, regularization, feature selection, 75/25 dataset split, learning curves, ROC curves, and others are explored and used.

II. State of the art

Machine Learning Repository mentions 3 relevant papers related to this dataset: [2], [3] and [4]. This dataset is quite famous, and since it is quite old (1988), there are many works on it, that use a range of different models and techniques, such as [5], [6], [7] and [8]. These 4 are very recent. Work [5] in particular, was published this year (2023) and reached amazing results, such as a 93.44% accuracy.

III. Dataset

The Cleveland dataset is a publicly available heart disease dataset from the UCI Machine Learning Repository. The Cleveland dataset from the UCI Machine Learning Repository contains a total of 297 examples, each with the attributes of the patient and whether they have heart disease (165) or not (138). The original dataset contains 76 features, but only the 13 most important ones are used. These are the most common and important ones used in heart disease diagnosis. The goal is to distinguish precense and absence of heart disease in the patient. The dataset can be accessed from the following link: https://archive.ics.uci.edu/ml/datasets/Heart+Disease

A. Dataset description

1) **age**: in years (numeric)
2) **sex**: 0 = female, 1 = male
3) **chestPainType**: chest pain type (1 = typical angina, 2 = atypical angina, 3 = non-anginal pain, 4 = asymptomatic)
4) **restingBP**: resting blood pressure (mm Hg) upon admission to the hospital (numeric)
5) **serumCholesterol**: serum cholesterol in mg/dl (numeric)
6) **fastingBloodSugar**: fasting blood sugar in mg/dl (0 if ¡= 120, 1 if ¿ 120)
7) **restingEcg**: resting electrocardiographic results (0 = normal, 1 = having ST-T wave abnormality, 2 = showing probable or definite left ventricular hypertrophy)
8) **maxHeartRate**: maximum heart rate achieved during exercise (numeric)
9) **exerciseInducedAngina**: exercise-induced angina (0 = no, 1 = yes)
10) **stDepression**: ST segment depression induced by exercise relative to rest (numeric)
11) **stSlope**: the slope of the peak exercise ST segment (1 = upsloping, 2 = flat, 3 = downsloping)
12) **majorVessels**: number of major vessels (0-3) colored by fluoroscopy (numeric)
13) **thalassemia**: thallium heart scan results (3 = normal, 6 = fixed defect, 7 = reversible defect)
14) **diagnosis**: the target/output (0 = no heart disease, 1 = has heart disease)

IV. Methodology

The objective of this project is to analyze the data and build a machine learning models that can accurately predict the presence or absence of heart disease in patients using the Cleveland dataset. The project will involve the following steps:

- Data preprocessing and analysis
- Feature selection
- Features encoding and normalization
- Model training: logistic regression, neural network, SVM, random forest, K nearest neighbours

- Hyperparameter tuning and regularization
- Model evaluation and analysis: metrics (such as accuracy, since the dataset is balanced), training and test subsets, cross-validation, cost functions, learning curves, ROC curves
- Models comparison

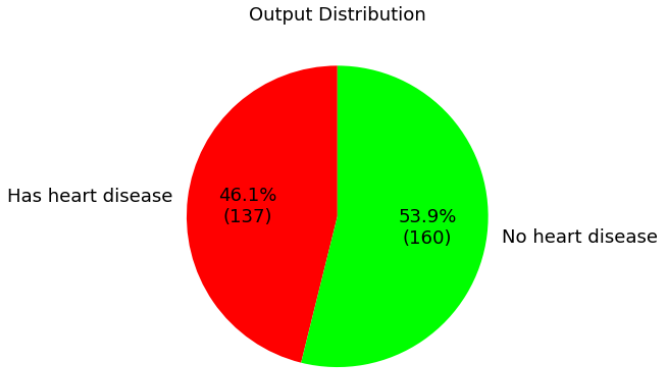
V. Data exploration and preprocessing

The original Cleveland dataset had 303 examples, but 6 had missing data. After removing them, we ended up with 297 examples.

The original dataset also had output values 1-4 to represent presence of heart disease; to manage the data more easily, we clipped the output values higher than 1 to be equal to 1. Now, the output is 0 (no heart disease) or 1 (has heart disease).

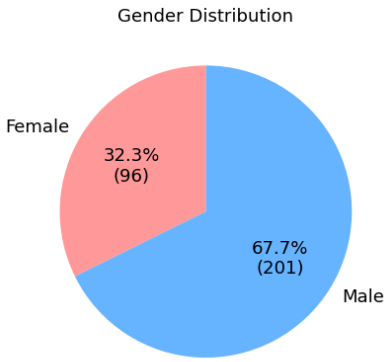
We also converted all data to float. Finally, we split the features into categorical features and numeric (continuous) features, since they require different ways to be analysed (for example, different types of plots).

A. Output Distribution



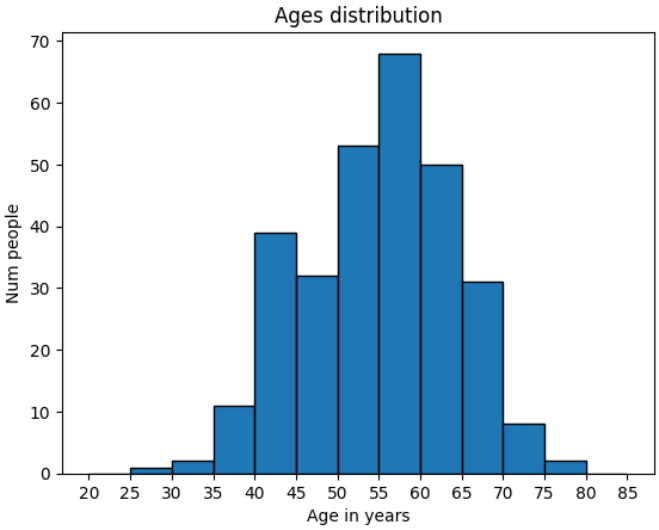
As we can see from the pie chart above, the dataset is balanced. This means accuracy will be a good metric to evaluate our ML models.

B. Gender Distribution



The gender distribution in the dataset is skewed towards males, representing 68% of the sample compared to 32% females.

C. Age Distribution



The histogram indicates a predominant presence of individuals in the older age group, specifically between 50 and 65 years old. Nevertheless, there is also a noticeable representation of middle-aged adults, falling within the range of 40 to 50 years old.

D. Categorical Features

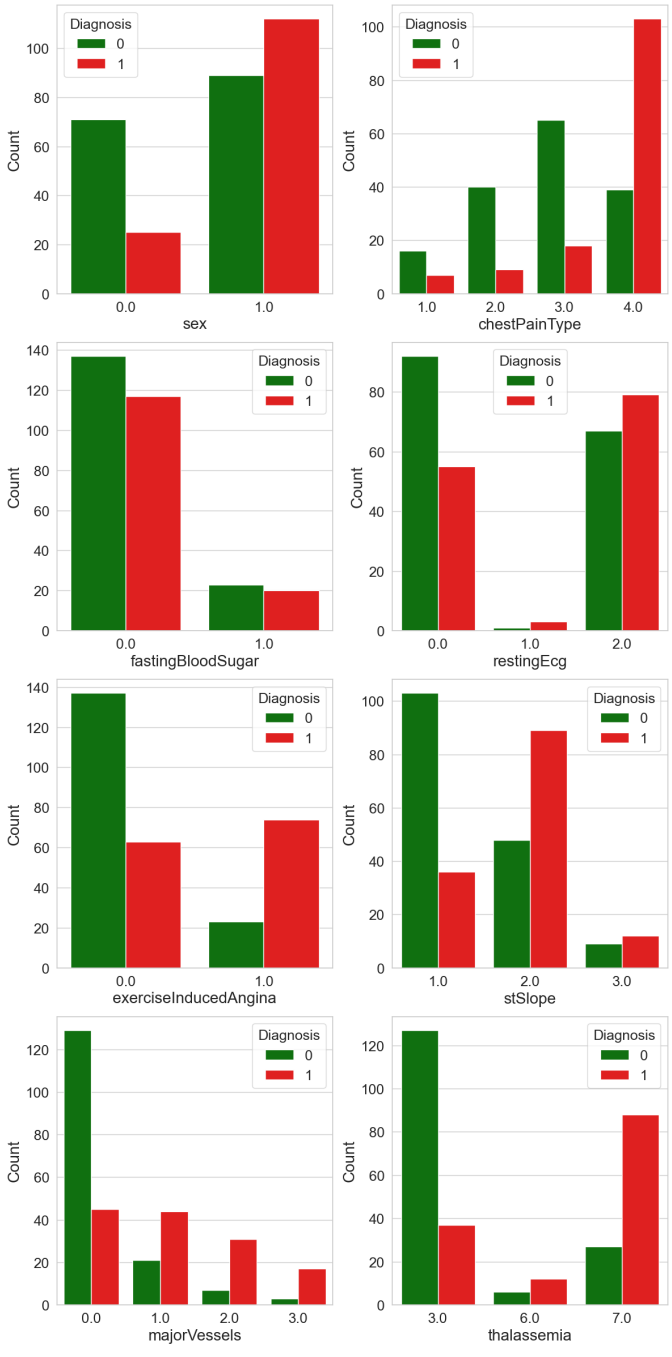


Fig. 1. Bar charts for categorical features. Green bars represent absence of heart disease, while red represents presence of heart disease

According to the bar charts above, the features sex, chestPainType, exerciseInducedAngina, thalassemia, stSlope and majorVessels have a significant impact on the heart disease diagnosis as they have a visibly different diagnosis distribution across their different values. For instance, a majority of the females do not have heart diseae, while there are as many males with heart disease as males without.

On the other hand, restingEcg and fastingBloodSugar have no impact on the diagnosis. In feature selection, these 2 features could be removed.

E. Numeric Features

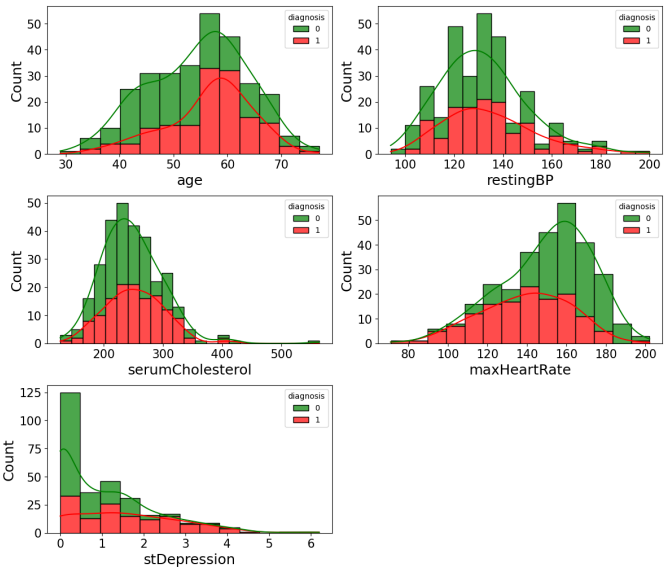


Fig. 2. Histograms for numeric features. Green bars represent absence of heart disease, while red represent presence of heart disease

The histograms above reveal that age, max heart rate, ST depression, have a strong relationship with the diagnosis of heart disease. As expected, older ages show more presence of heart disease. Higher max heart rates correspond to less presence of heart disease. At very high serum cholesterol levels, there is way more presence of heart disease than absence, as expected.

In feature selection, restingBP could be removed, as it doesn't seem to have impact on the diagnosis/output.

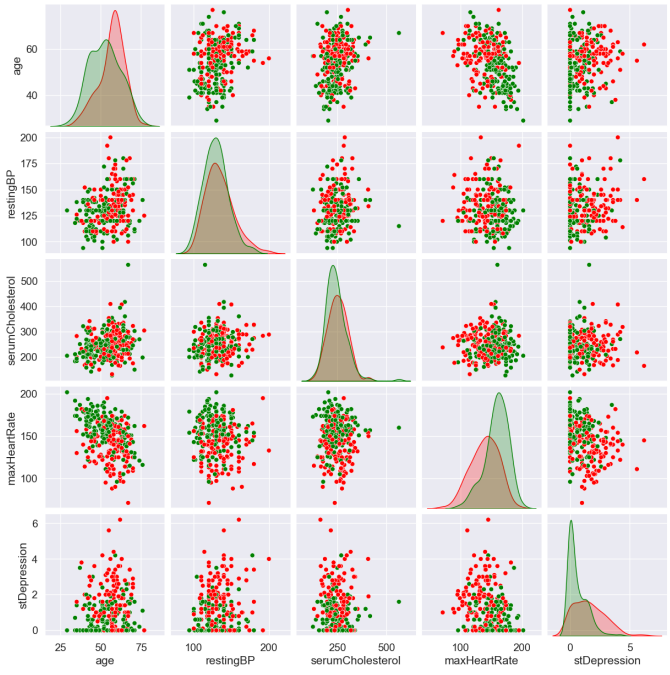


Fig. 3. Scatter matrix for numeric features. Red dots represent presence of heart disease, while green dots represent absence.

From the scatter matrix, we can see that age and max heart rate are the most separatable numeric features. At higher ages and lower max heart rates, there is more presence of heart disease.

F. Correlation Analysis

Correlation analysis can help identify features that are highly correlated with eachother and with the target variable (heart disease diagnosis, in this case). Correlation values close to -1 indicate that when a variable decreases, the other increases. Correlation values close to 1 indicate a perfect correlation; as one variable increases, so does the other. Correlation value of 0 represents no correlation.

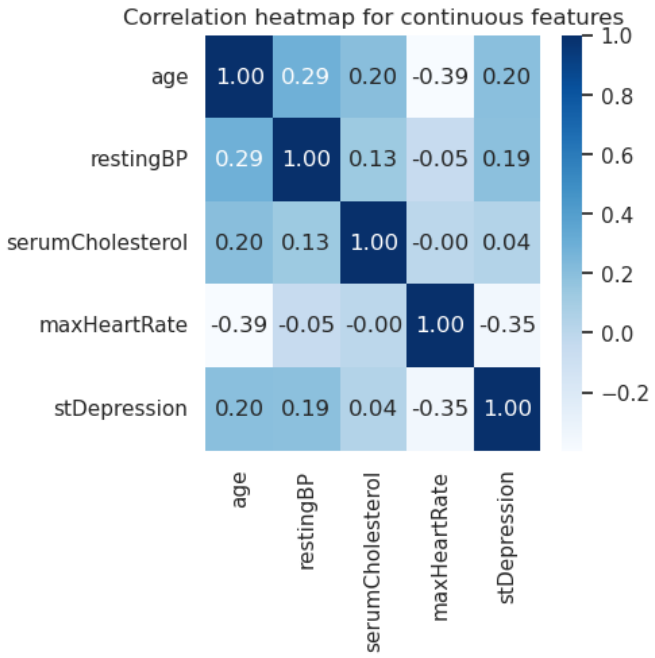


Fig. 4. Correlation heatmap for continuous features.

The correlation heatmap reveals a strong negative correlation (-0.39) between age and maxHeartRate. This finding supports our observation from the scatter matrix, indicating that as individuals age, their maximum heart rate tends to decrease, which aligns with expectations.

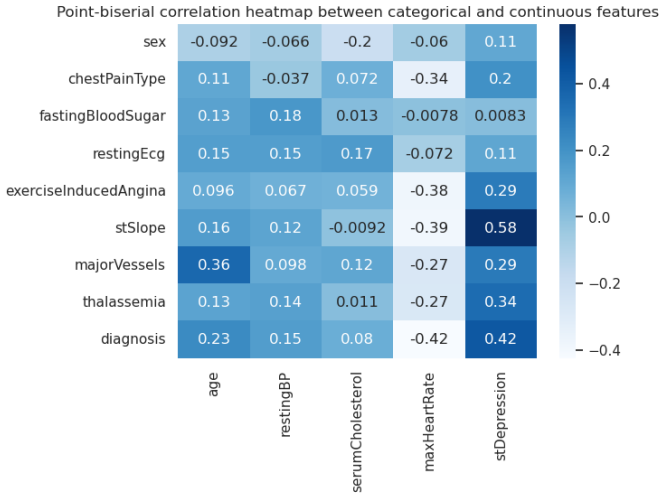


Fig. 5. Point-biserial correlation heatmap between categorical and continuous features.

From the correlation heatmap above, we can see that stDepression and stSlope have a very high correlation (0.58). As one variable increases, so does the other. We can also conclude that maxHeartRate and stDepression have a lot of impact on the diagnosis/output (-0.42). Serum cholesterol seems to have very little impact on the diagnosis (0.08).

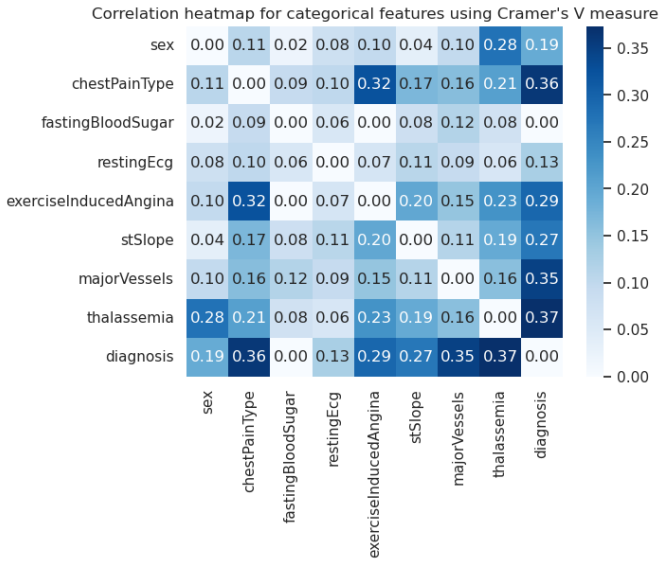


Fig. 6. Coorelation heatmap for categorical features using Cramer's V measure.

This last correlation heatmap (above), for categorical features, is a little different, as it uses Cramer's V measure. Here, there are no negative correlation values; instead, higher correlation values represent a stronger association between the 2 features. To confirm what we concluded in subsection D, we can see that chestPainType, majorVessels and thalassemia have a lot of impact on the diagnosis, while fastingBloodSugar has no impact.

VI. Feature selection

Based on the analysis in the section before, the features fastingBloodSugar, restingEcg and serumCholesterol have very low impact on the diagnosis/output. We will remove these 3 features if needed.

The features stDepression and stSlope have a very high correlation (0.58), meaning we can remove one of them as having both is redundant. If needed, we will remove stSlope, since stDepression has more impact on the diagnosis/output.

VII. Encoding and normalization, dataset split

Before training our models, we encoded the categorical features chestPainType, restingEcg, stSlope and thalassemia using One-Hot enconding. We didn't use label or ordinal enconding because these features do not have any hierarchical ordering. We also normalized the numeric features using Scikit's standard scaler.

If there's no feature selection and we use all 13 features, we have 22 features after enconding. If we remove the 3 low impact features, we have 17 features after encoding. And if we also remove stSlope, we end up with 14 features after enconding.

We split the dataset into 2 subsets: 25% for the test set and 75% for the train set.

VIII. Comparison

We applied 5 ML models: logistic regression, neural network, SVM, random forest and K nearest neighbours.

	Log Reg	NN	SVM	RF	KNN
Train accuracy	82%	85%	83%	84%	84%
Test accuracy	87%	85%	82%	86%	84%
ROC AUC	0.93	0.93	0.94	0.93	0.93

The table above shows the best results of each model. Regularized logistic regression is the best model since it has the highest test accuracy, at 87%, while being a perfect fit (no underfitting or overfitting). SVM is the worst model, at 82% test accuracy.

Ref	Year	Max accuracy
[5]	2023	93.44%
[6]	2023	87.91%
[7]	2022	93%
[8]	2020	87%

[5] trained 6 models (random forest, KNN, logistic regression, Naive Bayes, gradient boosting and AdaBoost) and also got the highest accuracy in logistic regression, at 90% test accuracy. After applying SVE (soft voting ensemble) to all models, [5] obtained a very good accuracy of 93.44%.

The next 5 sections explore the models we trained in more detail.

IX. Logistic Regression

A. Logistic regression from scratch

In a first version, we implemented logistic regression from scratch like we learned in classes.

The cost of initial theta (all zeros) is 0.693. With 5000 iterations, we tested 5 learning rates (alphas) - 0.001, 0.01, 0.1, 0.5 and 1 - and found that the best one is 0.1, since it results in the lowest cost (0.3613).

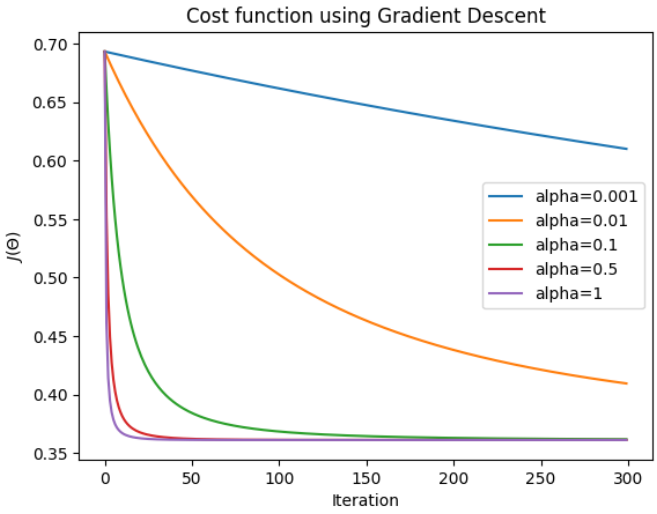


Fig. 7. Cost function using Gradient Descent, in unregularized logistic regression, for different learning rates.

The cost function up to 300 iterations is above. We can see that alpha=1 converges the fastest (in about 20 iterations), even if alpha=0.1 has the lowest cost after 5000 iterations.

The train accuracy of this unregularized logistic regression is 84.85%.

We then applied regularization. To find the best hyperparameters (learning rate and lambda), we ran a grid search. Tested learning rates: 0.001, 0.01, 0.1, 0.5, 1. Tested lambdas: 0, 0.001, 0.01, 0.1, 0.5, 1. The best combination is a 0.1 learning rate and lambda=0, since they result in the lowest cost after 5000 iterations.

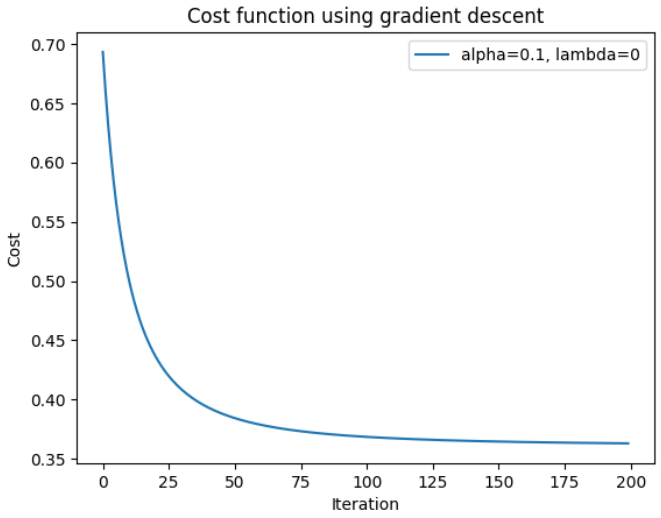


Fig. 8. Regularized logistic regression - Cost function for the best parameters (alpha = 0.1 and lambda = 0)

The cost function up to 200 iterations is above. As we can see, it converged in about 100 iterations.

Even with this regularization, the train accuracy stayed the same: 84.85%.

B. Logistic regression with scikit-learn

Here, we removed the 4 least impactful features explained in section "Feature Selection", since removing them fixed overfitting.

We trained both unregularized and regularized models with 10000 iterations and lbfgs solver (scikit default solver).

Without regularization, the model had a train accuracy of 84% and a test set accuracy of 85%. For the test set prediction, we used the optimal classification threshold of 0.32; this maximizes accuracy.

	Actual Pos	Actual Neg
Predicted Pos	TP = 34	FP = 8
Predicted Neg	FN = 3	TN = 30

In the confusion matrix above, we can see that this unregularized logistic regression has more difficulty correctly predicting the negative class (absence of heart disease).

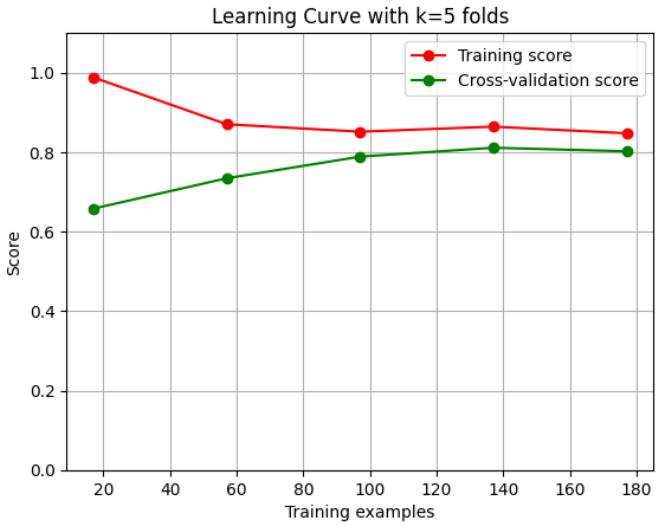


Fig. 9. Learning curve for non regularized logistic regression

In the learning curve above, we only used k=5 folds to make sure they have enough data to be reliable, since the whole dataset only has 297 examples.

To combat the very slight overfitting suggested by the learning curve above (training score very slightly higher than cross-validation-score), we applied regularization with l2 penalty. To tune the C hyperparameter, we tested 6 C values - 0.001, 0.01, 0.1, 0.5, 1, 10 - and found that the best C is 0.01, as it has the highest cross-validation score. To tune it, we used k=5 folds for the reason explained before.

In this regularized logistic regression, the train accuracy and test accuracy are 82% and 87%, respectively. In this case, the optimal classification threshold is 0.44 (see ROC curve below).

	Actual Pos	Actual Neg
Predicted Pos	TP = 36	FP = 6
Predicted Neg	FN = 4	TN = 29

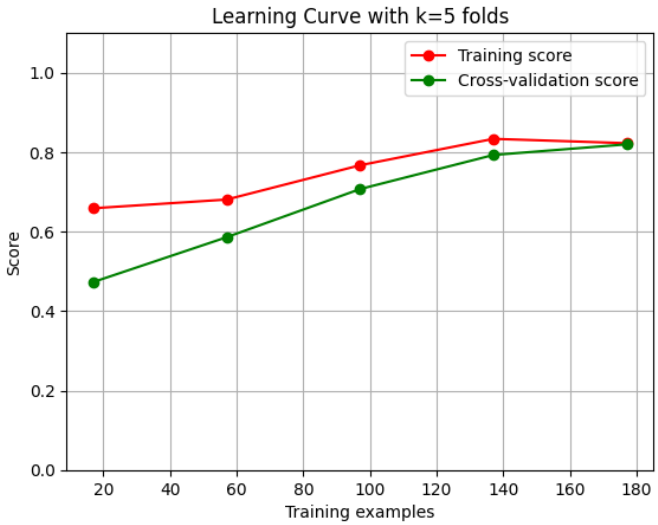


Fig. 10. Learning curve for regularized logistic regression

In the learning curve above, we can see an improvement over the learning curve before (unregularized). Now, we have a perfect model (no overfitting or underfitting), since the training score and cross-validation score are high and equal.

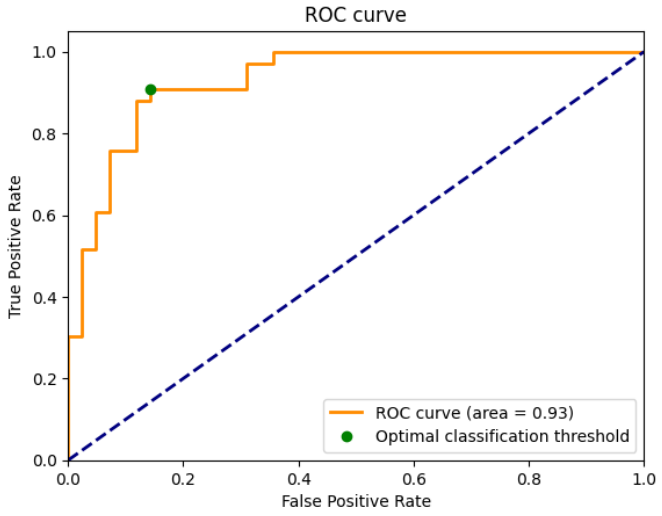


Fig. 11. ROC curve for regularized logistic regression

In the ROC curve above, the area is 0.93, which means our model is very good at predicting.

X. Neural Network

The next ML algorithm we used is Neural Networks. For our implementation, we utilized the MLPClassifier class from the Scikit-learn library, which is a feed-forward neural network algorithm based on multilayer perceptron architecture. We removed the 4 least impactful features explained in section "Feature Selection", since removing them results in the least overfitting plus another one, "restingBP" (5 features removed in total).

A. Unregularized Neural Network

We initially trained an unregularized and unoptimized neural network model with two hidden layers, each containing 25 neurons. The model was configured to use ReLU activation function and the SGD solver with a maximum of 500 iterations to prevent overfitting. We achieved the following confusion matrix.

	Actual Pos	Actual Neg
Predicted Pos	TP = 33	FP = 8
Predicted Neg	FN = 6	TN = 26

After training, the training accuracy was 84% and the test accuracy was 81%. The cross-validation scores showed variations across subsets, with a mean score of 79%. These results show some overfitting therefore we needed to adjust the hyperparameters to achieve higher accuracy.

B. Hyperparameter Tuning

We performed systematic hyperparameter tuning using GridSearchCV from scikit-learn. The best hyperparameters found were: activation='tanh', hidden layer sizes=(10,) (1 hidden layer and 10 neurons), learning rate='constant', and solver='adam'.The solver 'adam' in the MLP classifier by scikit-learn uses the Adam optimization algorithm, which minimizes the cost function using adaptive learning rates and momentum.

The following confusion matrix of the optimized model demonstrates higher predictive capability than the unoptimized model:

	Actual Pos	Actual Neg
Predicted Pos	TP = 34	FP = 7
Predicted Neg	FN = 4	TN = 28

The model in Figure 12 achieved an improved training accuracy of 84.5% and an improved test accuracy of 84.9%. These results indicate that the model achieved a reasonably good fit to the data, capturing the underlying patterns and making accurate predictions. The cross-validation scores also improved, with a new mean score of 81.3% which suggests that the model is less prone to overfitting.

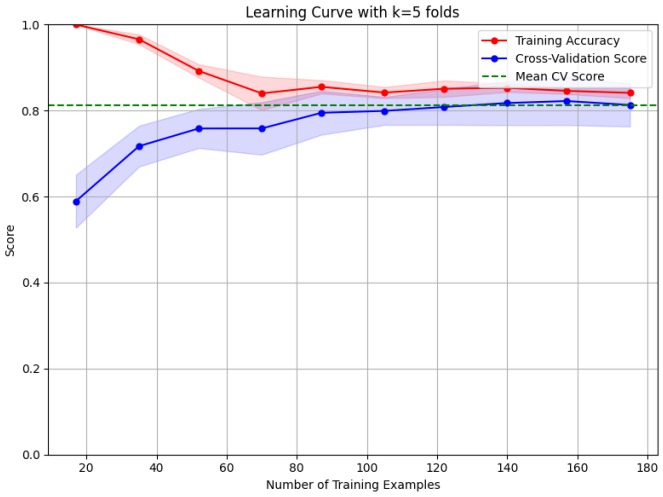


Fig. 12. Learning curve of the optimzed neural network model.

The classification report shows F1-score of 0.86 for the negative class (0.0) and 0.84 for the positive class (1.0) which suggests a good ability to discriminate between the two classes.

C. Regularized Neural Network

Regularization was incorporated into the MLPClassifier model using the "alpha" parameter to mitigate overfitting. By adding a penalty term to the loss function, the model was encouraged to have smaller weights. Hyperparameter tuning with gridsearchCV identified the optimal alpha value of 0.0001.

The new found confusion matrix, shown below, reveals that the model correctly predicts 34 instances of the negative class (Actual No) and 28 instances of the positive class (Actual Yes) which is an improvement relative to the unregularized version.

	Actual Pos	Actual Neg
Predicted Pos	TP = 34	FP = 7
Predicted Neg	FN = 4	TN = 28

The model achieves a training accuracy of 84.5% and a test accuracy of 84.9%. The cross-validation scores are also the same, with a mean score of 81.3%. These results are the same as the unregularized version. The ROC curve (Figure 13) of the model shows good predictive capability, with an AUC of 0.93.

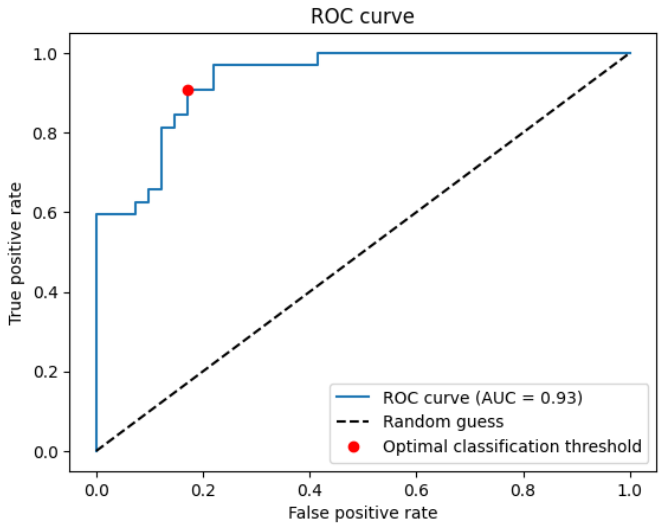


Fig. 13. ROC Curve for the Regularized Neural Network.

Upon analyzing the graph in the figure 14, we observe a decreasing trend in the validation error initially, indicating that the model's predictions are gradually improving with each iteration.

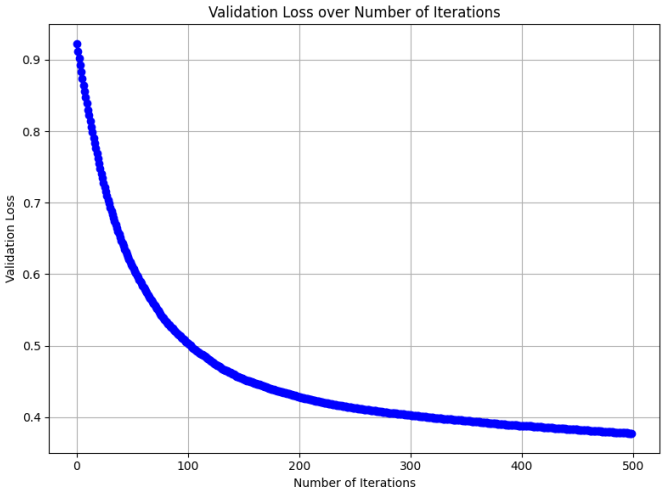


Fig. 14. Validation loss or validation error against the number of iterations. This graph illustrates the convergence of the model.

A stabilization in the validation loss graph would indicate minimal error between the model's predictions and the true values and therefore sufficient training and good predictive capability. However the graph doesn't stabilize till the 500 iterations mark which we attribute to the small sample size of our dataset.

XI. Support Vector Machines (SVM)

We also applied the Support Vector Machines (SVM) algorithm to the dataset. We used the scikit-learn library for training and evaluating the SVM models. We experimented with both non-linear (RBF kernel) and linear SVMs. We removed the 4 least impactful features explained in section "Feature Selection", since removing them results in the least overfitting plus another one, "restingBP".

A. Non-linear SVM with RBF Kernel

We trained the SVM classifier with the RBF kernel using different hyperparameters and evaluated its performance. The best hyperparameters were found using a grid search with 5-fold cross-validation. The tested hyperparameters were the regularization parameter C (0.1, 1, 10) and the kernel coefficient gamma (0.1, 1, 10). The best combination was C=1 and gamma=0.1.

	Actual No	Actual Yes
Predicted No	33	8
Predicted Yes	6	26

The accuracy of the non-linear SVM on the training set was 84%, and on the testing set was 81%. This 3% difference suggests that the model may be overfitting to the training data and struggling to generalize its findings to new, unseen data. The model achieved a mean cross-validation score of 0.82.

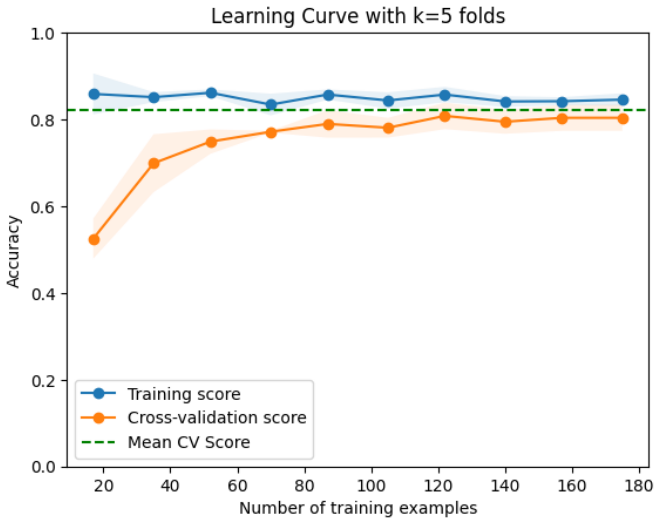


Fig. 15. Learning curve for non linear SVM with optimized hyperparameters.

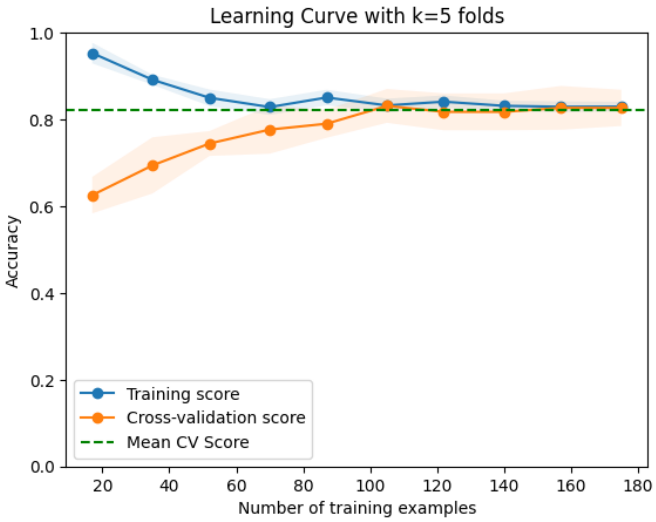


Fig. 16. Learning curve for Linear SVM with optimized hyperparameters.

We can confirm the suspect of overfitting in the learning curve as there is a slight gap between the training score and the cross-validation score. The precision and recall for class 1.0 (heart disease) were slightly lower, with values of 0.76 and 0.81, respectively. This imbalance in performance between the two classes could be another indication of overfitting.

B. Linear SVM

We also trained a linear SVM on the dataset. The linear SVM does not use a kernel function and tries to find the best linear decision boundary to separate the classes. The best hyperparameters found for the linear SVM were C=1 and gamma=0.1. The confusion matrix for the linear SVM is shown below:

	Actual No	Actual Yes
Predicted No	35	6
Predicted Yes	7	25

The model also achieved a training accuracy of 83%, a validation accuracy of 82%, and a testing accuracy of 82%. The model also achieved a mean cross-validation score of 0.82. The training accuracy is 83%, and the testing accuracy is 82%, indicating a relatively small gap between the two. Bellow is the learning curve for this algorithm.

The precision, recall, and F1-score for both classes were is quite good - for class 0.0, 0.84 and for class 1.0, 0.79. The precision, recall, and F1-score for both classes again exhibit slightly better performance on the training set compared to the testing set.

C. Summary

Both models demonstrated good accuracy and effectively classified both positive and negative cases and could serve as reliable approaches for heart disease classification. The linear SVM performed slightly better in terms of accuracy, although the difference is minimal. The linear SVM also presents less overfitting.

XII. Random Forests

We removed the 4 least impactful features explained in section ”Feature Selection”, since removing them results in the least overfitting. To further improve the performance of the Random Forest classifier, we performed hyperparameter tuning using grid search with 5-fold cross-validation. The selected parameter grid for tuning was as follows:

- **n_estimators**: 200
- **max_depth**: 3
- **min_samples_split**: 2
- **min_samples_leaf**: 1
- **max_features**: ’sqrt’

The Random Forest model achieved a training accuracy of 84.02% and a test accuracy of 86.30%. The learning curve shows that the model’s training and validation accuracies converge and stabilize as more data is used for training. This indicates that the model is not suffering from high bias or high variance and is able to generalize well to unseen data.

The Random Forest model shows a good overall performance in classifying instances of each class, with an F1-score of 0.88 for class 0 and 0.84 for class 1. This indicates that the model can effectively distinguish between benign and malignant breast tumors.

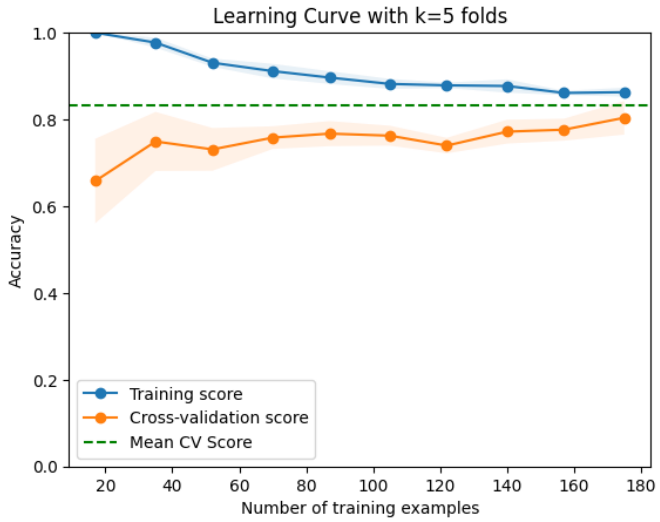


Fig. 17. Learning curve for Random Forest with optimized hyperparameters.

Class	F1-Score
0	0.88
1	0.84

The confusion matrix further confirms the model's performance, where it correctly predicted 36 instances of class 0 and 27 instances of class 1.

	Actual No	Actual Yes
Predicted No	36	5
Predicted Yes	5	27

The ROC curve shows that the Random Forest model achieves a good trade-off between true positive rate and false positive rate, further confirming the model effectiveness in distinguishing between the two classes. The optimal classification threshold of 0.52, along with an AUC of 0.93, indicates that the Random Forest model has a high discriminatory power.

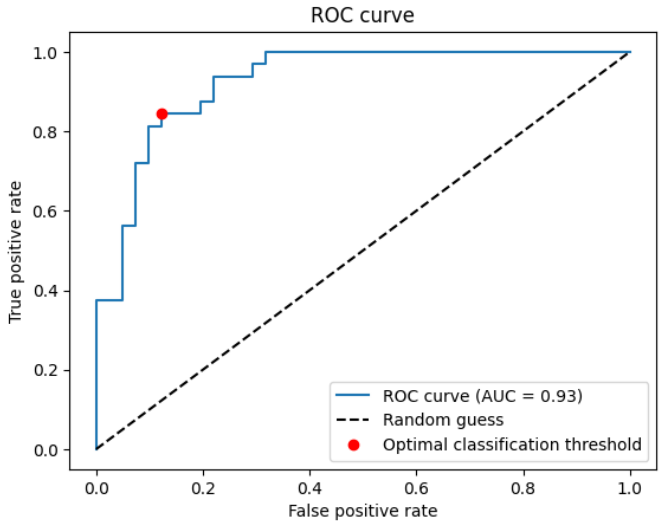


Fig. 18. ROC curve for Random Forest with optimized hyperparameters.

Analyzing the results, we can see that the Random Forest model outperforms the Decision Tree model which we also trained but are not analyzing in this report due to poor performance.

XIII. K Nearest Neighbours

Here, like in logistic regression, we removed the 4 least impactful features explained in section "Feature Selection", since removing them results in the least overfitting.

We used k=5 folds both for tuning the hyperparameters and for the learning curve, for the reason explained in Logistic Regression section, subsection B.

We tuned the 2 hyperparameters number of neighbours and leaf size with a grid search. We concluded that the best combination is 13 neighbours and leaf size of 1.

The unregularized version has the same train accuracy and test accuracy: 84%. We used the optimal classification threshold of 0.54 (see ROC curve below).

	Actual Pos	Actual Neg
Predicted Pos	TP = 37	FP = 5
Predicted Neg	FN = 7	TN = 26

From the confusion matrix above, we can conclude that this unregularized KNN is good at predicting both classes (negative and positive).

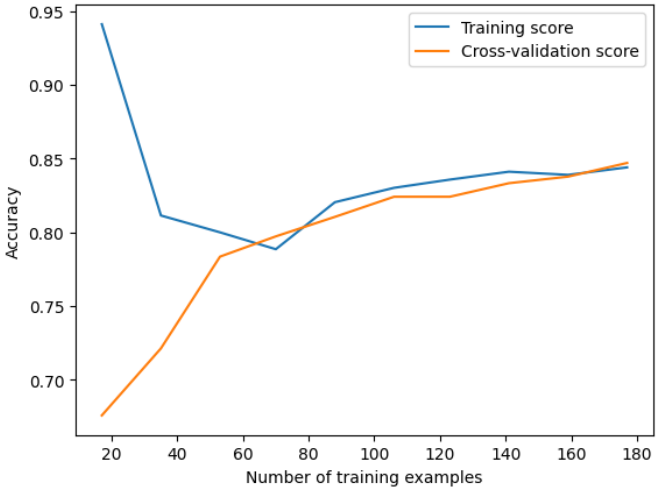


Fig. 19. Learning curve for unregularized KNN (k=5 folds)

In the learning curve above, the cross-validation score and training score are high and equal, which means it is a good model that doesn't overfit or underfit.

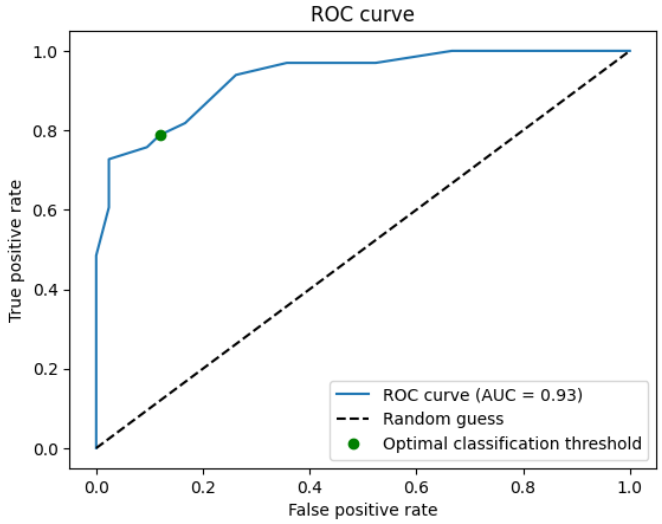


Fig. 20. ROC curve for unregularized KNN

In the ROC curve above, the AUC (area under curve) is 0.93, which means this unregularized KNN model is very good at predicting.

Surprisingly, the regularized version overfits even with different feature selections and extensive hyperparameters tuning. In its learning curve, the training score is always 1, while the cross-validation score is around 0.83. Hence, the unregularized version is better.

XIV. References

References

[1] Heart disease dataset at Machine Learning Repository: https://archive.ics.uci.edu/ml/datasets/Heart+Disease

[2] Detrano, R., Janosi, A., Steinbrunn, W., Pfisterer, M., Schmid, J., Sandhu, S., Guppy, K., Lee, S., & Froelicher, V. (1989). International application of a new probability algorithm for the diagnosis of coronary artery disease. American Journal of Cardiology, 64, 304–310.

[3] David W. Aha & Dennis Kibler. "Instance-based prediction of heart-disease presence with the Cleveland database."

[4] Gennari, J.H., Langley, P, & Fisher, D. (1989). Models of incremental concept formation. Artificial Intelligence, 40, 11–61.

[5] Chandrasekhar, N.; Peddakrishna, S. Enhancing Heart Disease Prediction Accuracy through Machine Learning Techniques and Optimization. Processes 2023, 11, 1210. https://doi.org/10.3390/pr11041210

[6] Ahamad, G.N.; Fatima, H.; Zakariya, S.M.; Abbas, M. Influence of Optimal Hyperparameters on the Performance of Machine Learning Algorithms for Predicting Heart Disease. Processes 2023, 11, 734.

[7] Rustam, F.; Ishaq, A.; Munir, K.; Almutairi, M.; Aslam, N.; Ashraf, I. Incorporating CNN Features for Optimizing Performance of Ensemble Classifier for Cardiovascular Disease Prediction. Diagnostics 2022, 12, 1474.

[8] Reddy, K.V.V.; Elamvazuthi, I.; Aziz, A.A.; Paramasivam, S.; Chua, H.N.; Pranavanand, S. Heart Disease Risk Prediction Using Machine Learning Classifiers with Attribute Evaluators. Appl. Sci. 2021, 11, 8352.