

SC5010 Introduction to Data Analysis

| Group Member Name | Matriculation Number | Contributions |
| --- | --- | --- |
| Ruth Kang Ming Ying | U214511C | 33.3% |
| Jambulingam Madhumita | U2240347L | 33.3% |
| Pang Teck Hock, Aldan | U2140205L | 33.3% |

Table of Contents

[Abstract 3](#_3obkbgqb60q4)

[Problem Description 3](#_b0ld53awhz92)

[Approach 3](#_ufc2n4i4hbia)

[Methodology 5](#_fqiafjlh5yic)

[Implementations 7](#_4fnaoj2o9g5w)

[Experimental Results and Analysis 8](#_1hhp5ank9i7f)

[XGBoost 8](#_9zer77uq6que)

[Random Forests 9](#_tvhlqk4drft0)

[SVM 10](#_r41mcyibxu3s)

[Logistic Regression 11](#_rej8mw4vo9xm)

[Pros and Cons 12](#_bb53spvmz3ki)

[Classification Accuracy 13](#_5ohg5jjhzv5)

[Confusion Matrix 13](#_lp64uf59cspa)

[Outliers and Missing Values 13](#_5xfbpn1zzizh)

[Interpretability 13](#_kc21v4ia9lz2)

[Feature importance 14](#_ijn5zi6djd8n)

[Sensitivity of Hyperparameters 16](#_qhnlszjkq04n)

[Conclusions 17](#_siqwqndhru1r)

[References 19](#_fxup3ffactvb)

# Abstract

Heart disease remains a significant public health concern worldwide, requiring accurate and reliable prediction models for early diagnosis and intervention. In this report, we present a comprehensive comparative analysis of four machine learning algorithms—XGBoost, Random Forests, Support Vector Machines (SVM), and Logistic Regression—for the prediction of heart disease. Leveraging a dataset from the UC Irvine Machine Learning Repository, we employed these models to predict the presence of heart disease. Our study focuses on evaluating the performance metrics, including accuracy and confusion matrices to assess the predictive capabilities of each model. Additionally, we explore the interpretability of these models through feature importance analysis and discuss their robustness to noise and sensitivity to hyperparameters. Our findings provide insights into the strengths and limitations of each algorithm in heart disease prediction, and contribute to the ongoing development of models for heart disease prediction.

# Problem Description

Heart disease develops from complex relationships between multiple measurements and risk factors. Due to this complexity, it is difficult to predict whether someone will develop heart disease, especially when current predictive tools do not capture all relevant risk factors effectively (Houssein et al., 2023). As such, our project experiments with different machine learning methods to find out which model is the best for heart disease prediction. In this project, we used Gradient Boosting, Random Forests, SVM and Logistic Regression.

## Approach

We obtained the Heart Disease dataset from the UCI Machine Learning Repository, where a subset of 14 variables (where all published experiments are performed on) are extracted from a database of 76 variables (Table 1). It is important to note that the dataset excluded quite a number of important features that could prevent more meaningful insights: ie. whether the patient is a smoker, the number of cigarettes they smoke a day, family history of coronary heart disease. Nevertheless, the full list of variables that we used are as follows and are further summarised in the source code:

| **Continuous variables** | |
| --- | --- |
| **Variable name** | **What it represents** |
| trestbps | Resting blood pressure on admission to hospital |
| chol | Serum cholesterol |
| thalach | Maximum heart rate achieved |
| oldpeak | ST depression induced by exercise relative to rest |
| ca | Number of major vessels (0-3) coloured by flourosopy |
| num | Diagnosis of heart disease |
| **Categorical variables** | |
| **Variable name** | **What it represents** |
| cp | chest pain type (1-4) |
| fbs | whether fasting blood sugar > 120mg/dl |
| restecg | resting electrocardiographic results |
| exang | exercise induced angina |
| slope | the slope of the peak exercise ST segment |
| thal | 3 = normal; 6 = fixed defect; 7 = reversable defect |

Table 1: Continuous and Categorical Variables

Due to varying methods, we prepared the dataset differently to obtain the highest predictive accuracy for each model. We tested our data preparation using a combination of dropping or imputing missing values, and dropping or imputing outliers with the median. We also considered the use of normalisation for our models, which is discussed below in each of the models and their accuracies.

## Methodology

We approached the project by comparing 4 different classification models for predicting presence of heart disease, using every feature available in the dataset.

Firstly, for data exploration, we observed the summary statistics and obtained the insights shown below:

* Patient’s ages ranges from 29 to 77, where 25% of them are 61 or older
* 25% of patients experience typical/mild angina (chest pain, a symptom of coronary heart disease). On average, people have non-anginal pain.
* Lastly, 75% of patients have < 50% diameter narrowing of their major vessels, indicating that most patients are relatively healthy and have no coronary heart disease.

We visualised the data with a correlation heat map (Fig 1) to give us an insight on which features are highly related to the presence of heart disease. In the context of heart disease, the more positively (resp. negatively) correlated the feature is, the higher (resp. lower) the presence of heart disease. Most features are observed to be positively correlated to heart disease presence, excluding thalach. In the next page we provide the heatmap obtained from our source code before preprocessing.

To pre-process the data, we first visualised outliers using boxplots to see if we needed to remove any. Furthermore, we observe every variable to see the percentage of missing values they contain. We removed/imputed missing values/outliers according to the respective model’s needs.

# 

# 

Fig 1: Correlation heatmap between all features

# Implementations

We employ 4 different models to compare their respective predictive accuracies: Random Forest, Gradient Boosting, Logistic Regression and SVM.

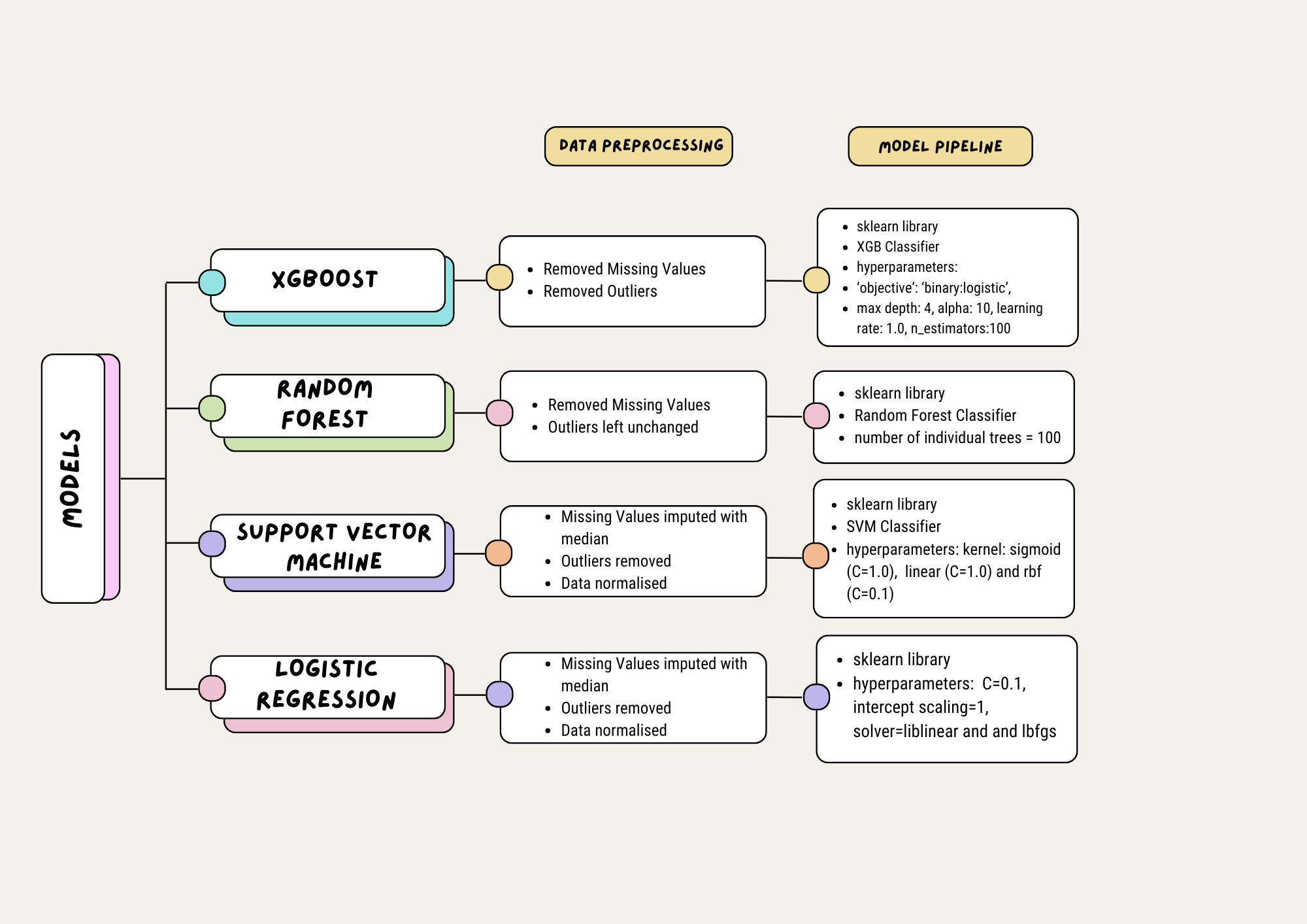


Fig 2: Flowchart summarising the implementations

# Experimental Results and Analysis

## XGBoost

XGBoost was used for gradient boosting. In the preprocessing steps taken, we first experimented with outlier and missing values removal. The following table represents the changes to the datasets and its corresponding predictive accuracies:

| Dataset changes | | Predictive accuracy (%) |
| --- | --- | --- |
| Original dataset (No preprocessing - possible because XGBoost handles NaN values during training) | | 81.97 |
| Missing values (Found in ‘ca’ and ‘thal’). Outliers left constant. | Row removed | 86.67 |
| Imputed with median | 81.97 |
| Outliers (detected with interquartile range metric) | Outlier rows removed | 79.31 |
| Imputed with median | 81.97 |
| Missing values removed, Outlier removed  (Note that any other combination results in lower predictive accuracy) | | 87.72 |

Table 2: Dataset Changes vs Predictive Accuracy for XGBoost

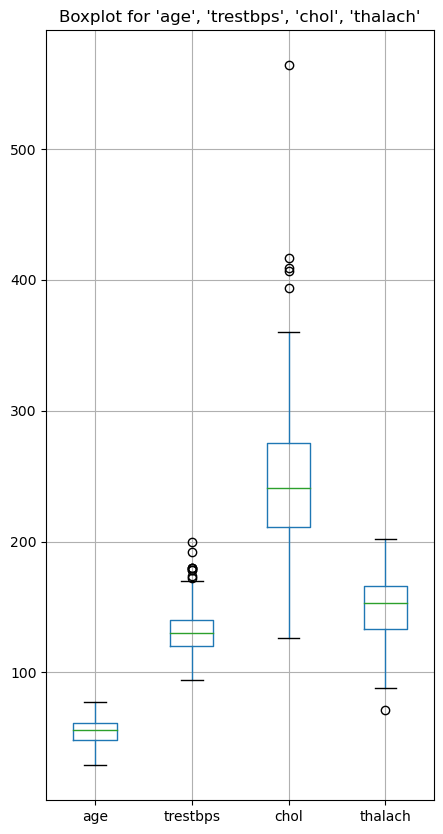


Fig 3: Boxplot for continuous variables except ‘oldpeak’

During the outlier removal process it’s noted that row 152 in the ‘chol’ column is much higher than the rest (‘chol’ = 564). We tried removing just that outlier’s row while keeping everything constant to compare classification accuracy and noticed that it is lower than 87.72% (our peak performance). This information is omitted from the table above but is important to mention to show ensemble classifier’s resilience to outliers.

In our final code we included imputing outliers with the median of that column as it increased our final predictive accuracy. This contradiction can be explained by the fact that the outliers might have led to splits in the decision trees that are not optimal for the majority. Imputing outliers with the median reduces their influence on the model which improves overall performance.

Normalisation using sci-kit learn’s MinMaxScaler was used on the dataset but classification accuracy fell regardless of how we dealt with missing values and outliers. We posit that because XGBoost is a tree-based model, scaling features to a smaller range [0,1] greatly reduces the discriminatory ability of the model - thus reducing the model’s ability to determine feature importance, which affects the predictive accuracy of the model.

We look at the confusion matrix and feature importance of all models under pros and cons.

## Random Forests

| Dataset changes | | Predictive accuracy |
| --- | --- | --- |
| Missing values (Found in ‘ca’ and ‘thal’) | Row removed | 88.33% |
| Imputed with median | 86.88% |
| Missing values (Row removed) | Outlier removed | 85.71% |
| Outlier imputed with median | 88.33% |
| Missing Values (Row imputed with median) | Outlier removed | 87.71% |
| Outlier imputed with median | 86.88% |
| Missing values removed, Outlier rows unchanged (Combination with the highest accuracy) | | 88.33% |

Table 3: Dataset changes vs Predictive accuracy for Random Forest

The sklearn library that was used to build the Random Forest model is not robust enough to work with missing values as it does not have a built-in function to deal with them like XGBoost does. Hence the classification model does not run without any data preprocessing. The accuracy either decreases or remains constant when outlier rows are excluded or retained or imputed with the median. Hence, in our final code we only removed the missing values and left the outliers unremoved and did not impute the values either. Ensemble algorithms such as random forest are not sensitive to outliers since the effects of outliers on a single tree is averaged out across multiple trees, reducing the outlier’s effect on the final mode.

Normalisation of the data does not improve or decrease the accuracy. Similar to gradient boosting, normalisation does not affect ensemble algorithms like Random Forest.

## 

## SVM

| Dataset changes | | Predictive accuracy | | |
| --- | --- | --- | --- | --- |
| rbf | linear | sigmoid |
| Missing values (Found in ‘ca’ and ‘thal’) | Row removed | 81.67% | 83.33% | 83.33% |
| Imputed with median | 78.69% | 81.97% | 77.05% |
| Missing values (Row removed) + Outliers (detected with interquartile range metric) | Outlier rows removed | 83.93% | 82.14% | 82.14% |
| Imputed with median | 81.67% | 83.33% | 83.33% |
| Missing values (Imputed with median) + Outliers (detected with interquartile range metric) | Outlier rows removed | 75.44% | 85.96% | 84.21% |
| Imputed with median | 80.33% | 80.33% | 78.69% |

Table 4: Dataset changes vs Predictive accuracy for SVM

Similar to Random Forest, SVM requires the data to have NaN values removed/replaced before preprocessing since scikit-learn does not have built-in functions to handle NaN values, and NaN values may disrupt the computation of distance between data points.

Across all 3 different kernels, the linear kernel generally retains the most accurate performance. Overall, by imputing the missing values with median and removing outliers from the data set using the linear kernel, we get the best performance accuracy of 85.96%. Normalisation of the data was used to help SVM to ensure the features have the same scale, and we do so by using sklearn.preprocessing’s StandardScaler. This helps to reduce SVM’s sensitivity to the scale of features, since it relies on computing the distances between data points.

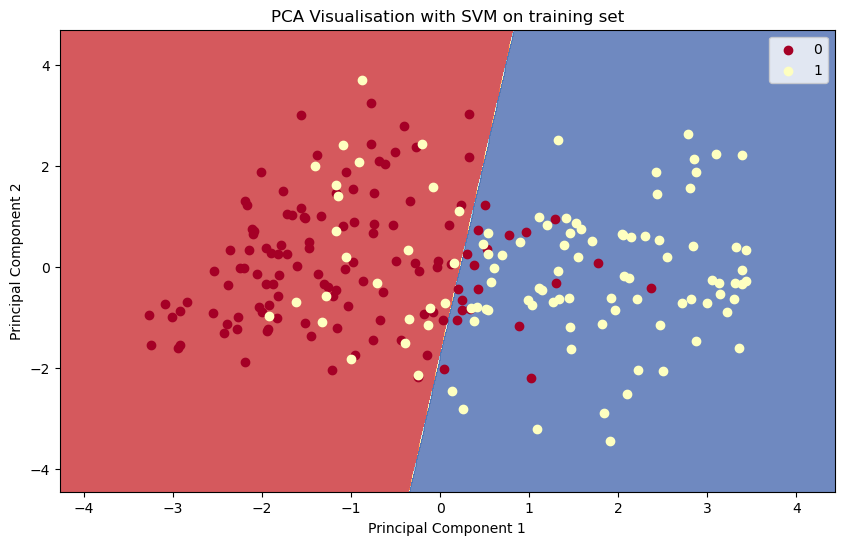
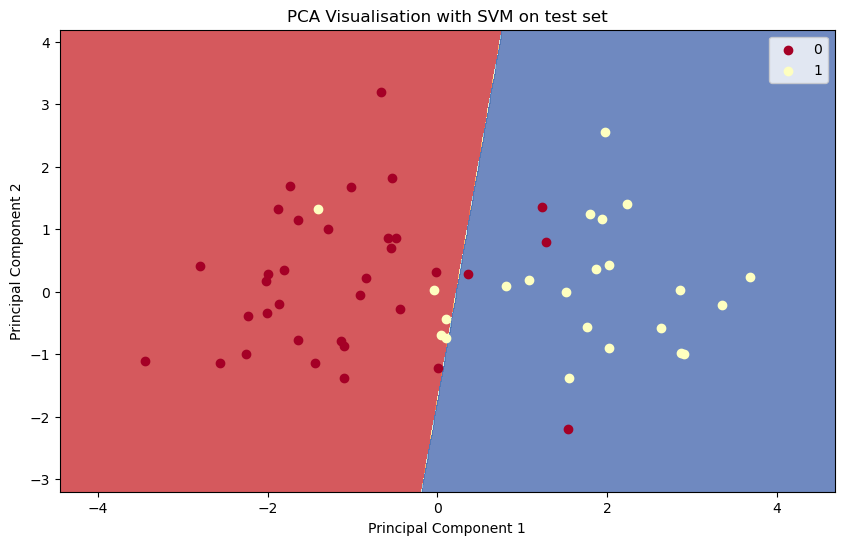


Fig 4: PCA Visualisation with SVM on training set

The accuracies provided could be due to the data being almost “linearly separable”. When using a PCA visualisation, the training set shows how SVM is able to separate the classes of patients into 0 (no heart disease) and 1 (have heart disease). The accuracy given after training the dataset was 80.18%.

Fig 5: PCA Visualisation with SVM on test set

The test set shows an improvement in separating the classes of patients, and hence making it close to be “linearly separable”. The accuracy given after testing the dataset increased to 84.21%. However, it should be noted that when using PCA and SVM, the accuracy will drop due to a reduction of dimension by PCA which could lead to the loss of information. In reality, it should also be noted that datasets could never really be linearly separable (Chen, 2019).

## Logistic Regression

| Dataset changes | | Predictive accuracy | |
| --- | --- | --- | --- |
| liblinear | lbfgs |
| Missing values (Found in ‘ca’ and ‘thal’) | Row removed | 83.33% | 83.33% |
| Imputed with median | 80.33% | 78.69% |
| Missing values (Row removed) + Outliers (detected with interquartile range metric) | Outlier rows removed | 83.93% | 83.93% |
| Imputed with median | 83.33% | 83.33% |
| Missing values (Imputed with median) + Outliers (detected with interquartile range metric) | Outlier rows removed | 85.96% | 85.96% |
| Imputed with median | 80.33% | 80.33% |

Table 5: Dataset changes vs Predictive accuracy for Logistic Regression

Similarly, Logistic Regression also does not run if NaN values are not removed or replaced, due to the calculation of log-odds and probabilities of the target classes that are required in this model.

Using Logistic Regression, it would seem that imputing the missing values and removing outlier rows has the highest predictive accuracy across the 2 different solvers (85.96%). It can be observed that both solvers give almost the same accuracy throughout the different combinations to the dataset, which can only mean that the dataset could be linearly separable. Normalisation was used, and it helps to ensure that the coefficients used in the data set to represent changes in target variable (num) is for a one-unit change in each feature. It does not get affected by the scale of the feature like SVM, however it benefits from using the same function from sklearn.preprocessing’s StandardScaler to scale the data.

# Pros and Cons

| **Classification Accuracy Rates and Confusion Matrix** | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Models | Accuracy (%) | True Positive | True Negative | False Positive | False Negative | Precision (%) | Recall (%) | F1 Score (%) |
| XGBoost | 87.72 | 27 | 23 | 2 | 5 | 93 | 84 | 89 |
| Random Forest | 88.33 | 32 | 21 | 4 | 3 | 90 | 84 | 87 |
| SVM | 85.96 | 28 | 21 | 2 | 6 | 93 | 82 | 88 |
| Logistic Regression | 85.96 | 28 | 21 | 2 | 6 | 93 | 82 | 88 |

Table 6: Classification Accuracy Rates and Confusion Matrix

| **Brief comparison between models on certain aspects** | | | | | |
| --- | --- | --- | --- | --- | --- |
| Model/  Aspects | Handles Missing Values | Feature Importance | Interpretability | Sensitivity to hyperparameters | Robustness to noise |
| XGBoost | Yes | Yes | Moderate | High | Moderate |
| Random Forest | Yes | Yes | Moderate | Moderate | High |
| SVM | No | No | Low | High | Moderate |
| Logistic Regression | No | Yes | High | Low | High |

Table 7: Comparison between Models

## Classification Accuracy

Random Forest has the highest classification accuracy at 88.33% followed by XGBoost which is only slightly lower at 87.72%. Ensemble algorithms such as Random Forest and XGBoost use multiple decision trees to predict the classification label. Hence they have a more accurate prediction compared to single classification models such as SVM or Logistic Regression.

## Confusion Matrix

Across all 4 models, it is noted that Random Forests outclasses all other models in predictive accuracy. Regarding precision, XGBoost, SVM and Logistic Regression provide the highest - indicating the highest accuracy of positive predictions by them. High precision (low false positive rates) is necessary as false positives are costly in medical diagnosis. On the other hand, XGBoost and Random Forest have the highest recall - a low false negative rate - indicating that XGBoost and Random Forests has less instances where it is wrong in classifying people as healthy. Overall, XGBoost has the highest F1 score which means that combining recall and precision, XGBoost has the best overall performance in terms of classification accuracy.

## Outliers and Missing Values

Random Forest and XGBoost are also more robust to outliers due to their ensemble nature. The influence of outliers or missing values is mitigated usually as ensemble classifiers tend to average out an individual tree’s predictions, which prevents predictive accuracy from being affected as much. Logistic Regression handles outliers well (unless they are extreme values) as the sigmoid function tapers them and SVM handles outliers by ignoring them to find the hyperplane with maximum margin. For missing values, both Logistic Regression and SVM do not handle them by themselves and require removal or imputation.

## Interpretability

In this section, we study the model’s decision making process rather than just blindly using their results.

XGBoost and Random Forests are moderately interpretable given that we’re able to obtain feature importance scores to understand which features played a greater role in the final model. However, given that they are ensemble classifiers, understanding the decision-making process from the final decision tree can be difficult and therefore prevents us from explaining the results. However, we can understand what features played a more important role in the decision trees through plotting their feature importance.

SVM is typically difficult to interpret due to its complex decision boundaries in a high-dimensional space. Although we can use PCA to reduce the dimension of the dataset to possibly allow easier interpretation, doing so results in loss of important information that could affect model accuracy.

Logistic Regression has high interpretability as the estimated coefficients associated to each feature tells us directly the impact of the feature in the prediction of heart disease, making it easy to understand Logistic Regression’s model behaviour.

## Feature importance

In this section, we observe what feature contributed more weight to the predictions in each model.

For one, since XGBoost is able to capture complex nonlinear relationships between the features and the target variable (num) directly from a trained predictive model. It calculates the usefulness of each feature when constructing decision trees within the model, where it weights the number of observations each node is responsible for. Afterwhich, the feature importance is averaged across all the decision trees in the model.

Another factor could include the model’s own use of metrics, where XGBoost often uses metrics such as Gini importance, which measures the total decrease in node impurity, thus capturing features that contribute more in helping the model to reduce impurity of decision trees. A similar argument can be made for Random Forest, which does the same steps as XGBoost for finding feature importance.

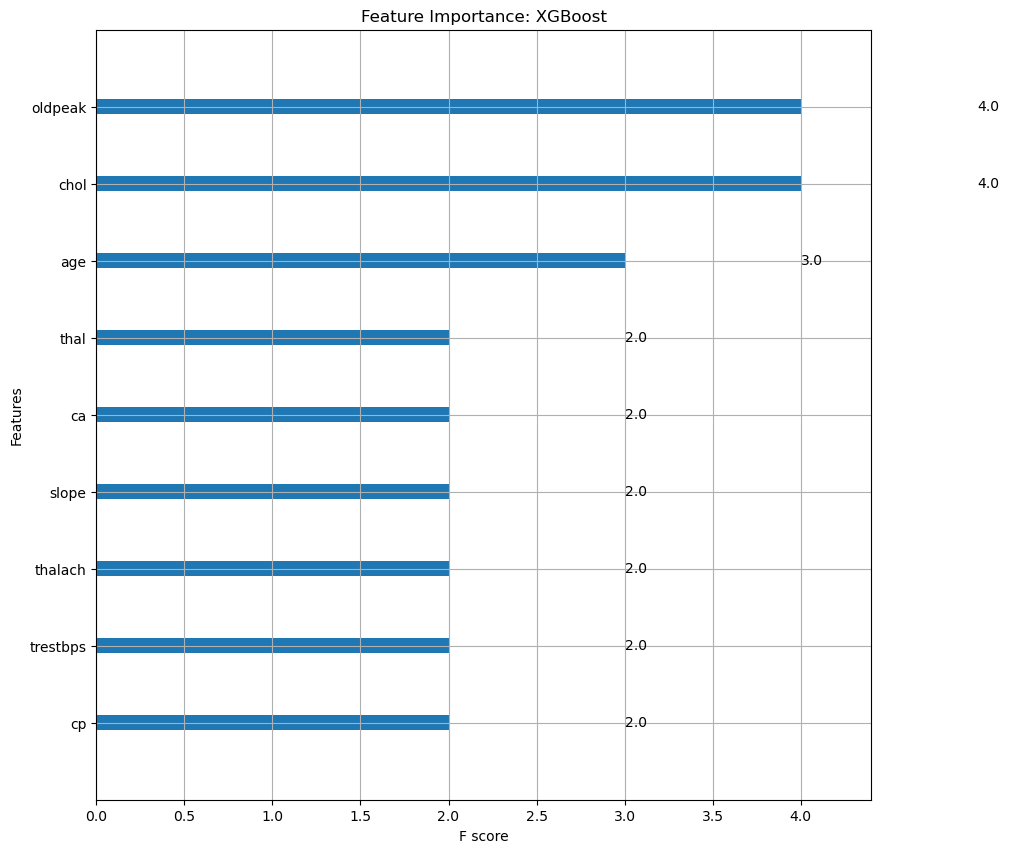


Fig 6: Feature Importance of XGBoost

As seen in this bar graph, “oldpeak”, “chol” and “age” have the highest feature importance at 4.0 and 3.0 as compared to the other features.

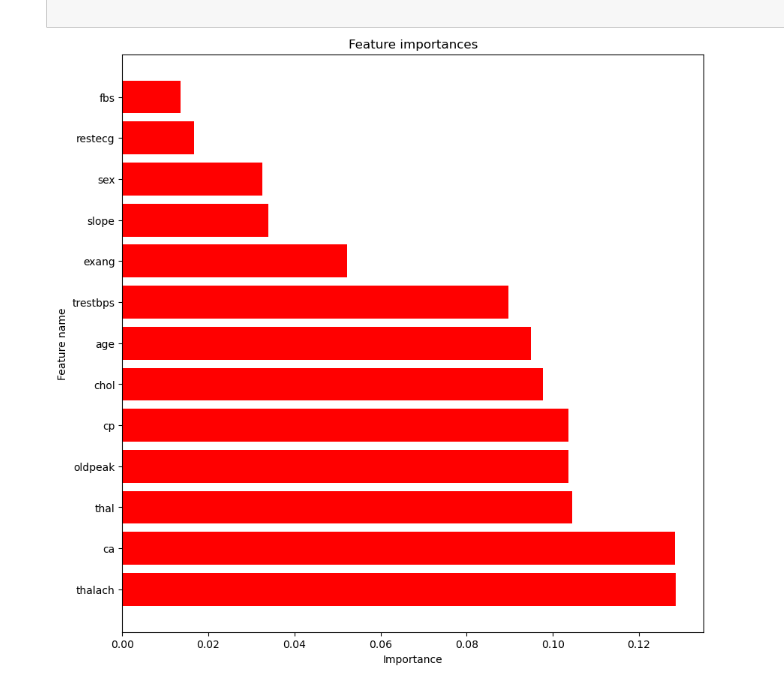


Fig 7: Feature Importance of Random Forest

As seen, ‘ca’ and ‘thalach’ have the highest feature importance for the Random Forest Model at 0.13

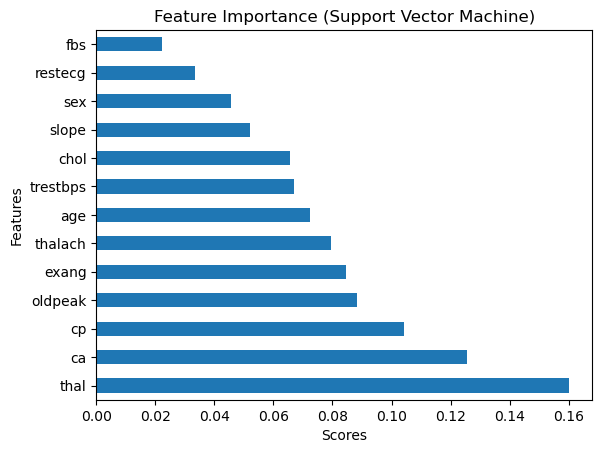


Fig 8: Feature Importance of SVM

Using a chi-squared statistical test, we can find the feature importance for SVM. Note that “thal”, “ca” and “cp” are the features with the highest importance.

SVM and Linear Regression however differ, since they do not have built-in functions to compute feature importance. We can however still compute manually to find out feature weights from linear SVM, though it may not be comprehensive as compared to a non-linear SVM. This is especially so for linear models, which assumes the linear relationship between features and target variables.

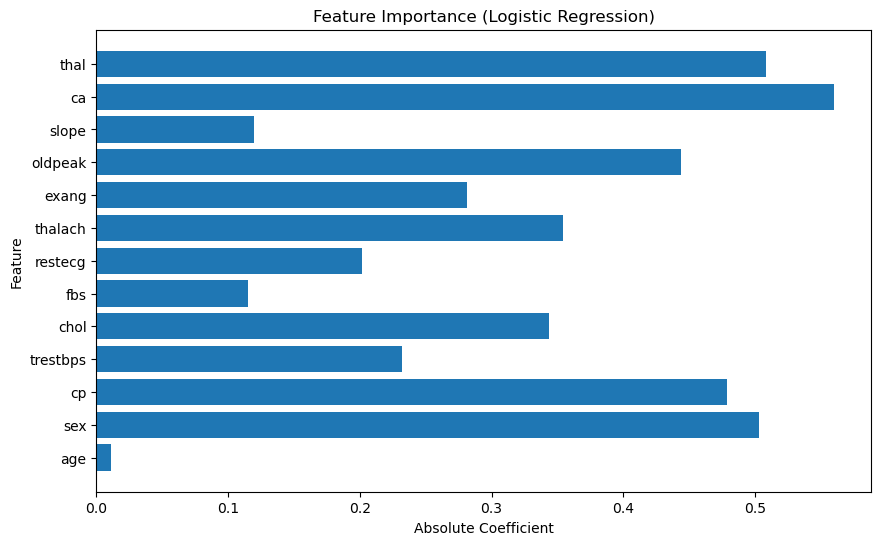


Fig 9: Feature Importance of Logistic Regressions

Logistic Regression uses a different method of computing feature importance. It takes in the absolute values of the coefficients of the logistic regression model. The features that are of higher importance in Logistic Regression differ slightly – “ca”, “thal”, “sex” and “cp”.

## 

## Sensitivity of Hyperparameters

Models such as Random Forests and Logistic Regression were less sensitive to hyperparameters and thus were able to obtain accuracies similar to that of the highest accuracies found in the UCI Irvine Machine Learning (UCI ML) Repository, with just a little hyperparameter tuning.

On the other hand, models like XGBoost and SVM are very sensitive to hyperparameters. For SVM, choice of kernel and regularisation parameters significantly affected performance but results were still similar to that of the UCI ML Repository. For XGBoost, we attempted hyperparameter tuning using GridSearchCV but realised that because of computational limitations, not many parameters can be put into the grid to find a good combination of hyperparameters. This is likely the reason why our final accuracy for XGBoost is 2% short of the highest one found in UCI ML Repository.

# 

# Conclusions

Overall, we managed to train 4 different models for the prediction of heart disease and compared their strengths and weaknesses.

First we experimented with data preprocessing to see how it affects model accuracy and explained the rationale behind the numbers we obtained. Furthermore, we see why some preprocessing steps work on some models but not on others.

Second, we compared the models against each other based on their confusion matrices and on some points of comparisons. We conclude that generally XGBoost is the best model for classification due to the reasons stated below:

1. Highest F1 score with predictive accuracy that can be the highest with the right hyperparameter tuning. Better accuracy means less error in diagnoses and therefore less medical resources are wasted.
2. Handles missing values and is robust to noise – although Random Forest is also an ensemble classifier, scikit-learn’s decision trees do not have built in functions for dealing with missing values (shown in our control experiments where no preprocessing was done). SVM and Logistic Regression do handle outliers well, but aren’t able to do the same for missing values.

Lastly, we discuss areas where we can improve in our project.

The dataset we used is merely a subset of 14 features, out of 76 in total. This means our results overlook features that could be important in determining presence of heart disease, such as whether the patient is a smoker or not, or whether their family has a history of coronary heart disease. Further work can be done in a dataset with more relevant features.

Furthermore, it is important to note that in UC Irvine’s Machine Learning Repository, XGBoost is capable of reaching classification accuracies of 89.47% – outclassing every other model attempted. It is likely that there are better ways of preprocessing the data/hyperparameter tuning they have done that caused our results to differ, but random search/grid search for best hyperparameters is computationally expensive as was not done.

XGBoost is known to be black-box and thus is less interpretable than the other models. We conceded interpretability for higher predictive accuracy as that was the main goal of our project, but in the area of healthcare, explainable AI models are imperative as we need clear explanations on why decisions are made - especially since we cannot be reckless with human lives.

We could also consider under-sampling cases where people do not have heart disease from the dataset before preprocessing it and using it for model training. However, given that our data is considerably small (with 303 records), sampling such a small dataset could lead to overfitting - thus skewing our predictions. Therefore, we should instead look into alternative ways of handling imbalanced classes.

# References

* Houssein, E. H., Mohamed, R. E., & Ali, A. A. (2023, May 3). *Heart disease risk factors detection from electronic health records using advanced NLP and Deep Learning Techniques*. Nature News. <https://www.nature.com/articles/s41598-023-34294-6#citeas>
* Chen, L. (2019, January 7). *Support Vector Machine - simply explained.* Medium. <https://towardsdatascience.com/support-vector-machine-simply-explained-fee28eba5496>