**Project Report**

Student Name (Matric Number) : Soh Le E Celest

Matriculation Number : 2301028C

Jenny LI

NG (TP)

Introduction

This application implements a machine learning model that predicts if a patient has heart disease or not, based on data gathered from clinical tests.

Heart disease encompasses various conditions including coronary artery disease, heart valve disease and arrhythmia. Individuals who are older, have high blood pressure, high cholesterol and/or have high blood pressure tend to be at greater risk.[2]

The chosen dataset for training is Heart Disease Prediction Dataset from Kaggle, a compilation of patient data from Hungarian Institute of Cardiology, University Hospital (Zurich, Switzerland), University Hospital (Basel, Switzerland), Medical Center (Long Beach) and Cleveland Clinic Foundation.[1]

The dataset contains the following columns:

|  |  |  |
| --- | --- | --- |
| Name | Data Type | What It Represents |
| id | int64 | The unique id of each patient |
| age | int64 | The patient’s age (years) |
| sex | object | The patient’s gender |
| dataset | object | Where the data was from; location of medical institution |
| cp | object | The type of chest pains the patient experienced.  (typical angina, atypical angina, non-anginal, asymptomatic) |
| trestbps | float64 | The patient’s resting blood pressure (mm Hg) when admitted to the hospital |
| chol | float64 | The patient’s serum cholesterol (mg/dl) |
| fbs | object | Whether or not the patient’s fasting blood sugar is above 120 mg/dl (True/False) |
| restecg | object | The patient’s resting electrocardiographic results  (normal, st-t abnormality, lv hypertrophy) |
| thalch | float64 | The maximum heart rate the patient achieves during a stress test [3]  (bpm) |
| exang | object | Whether or not the patient experiences angina from exercise  (True/False) |
| oldpeak | float64 | The length (mm) of ST depression induced by exercise relative to rest.  The unit of measurement for ST depression is decided based on typical measurements. [4] |
| slope | object | The type of slope of the peak exercise ST segment  (flat, upsloping, downsloping) |
| ca | float64 | The number of major vessels coloured by fluoroscopy (0-3) |
| thal | int64 | The types of Thalassemia the patient suffers.  (normal, reversable defect, fixed defect) |
| num | int64 | Whether or not the patient has heart disease |

Data Exploration

Data exploration began with converting the dataset to a DataFrame and viewing a summary of what columns there are in the dataset, how many non-null rows each column has and what data type each column is using pandas’ info() method. Then, the dataset was checked for duplicate columns, of which none were found.

Numeric Columns

For each, a histogram was used to view distribution and box plots were used to view range and check for outliers, including those that may represent invalid values.

A correlation matrix containing the correlation coefficients for all numeric columns was also produced to check for multicollinearity and potential relationships between these columns. Using information from the initial summary of the dataset to find which columns have missing values, for each of these columns, the percentages of rows with missing values in comparison to the present total number of rows in the dataset is calculated to check the proportion of missing values.

Categorical Columns

For categorical columns, the pandas unique() method was used to look at every distinct value under the column and to assess the presence of missing values. Then, pie chart or bar chart was used to examine distribution and balance of categories.

To check for multicollinearity and relationships, a Cramer’s V [5] calculation was used to check for correlation between categorical columns instead of correlation coefficient as it caters only to numeric values. When examining each categorical column, a contingency table for it and another categorical column was created to get a chi-square statistic [6]. Afterwards, the chi-square statistic along with the sample size, the number of rows and number of columns in the contingency table were put in the Cramer’s V formula [5] to get the Cramer’s V value. This is repeated to get the correlation between the current categorical column and every other categorical column as Cramer’s V compares 2 features. This process utilised numpy and scipy.stats methods.

Using information from the initial summary of the dataset and the values returned from unique() to identify which columns have missing values, a percentage of missing values is calculated similar to with numeric columns.

Columns with moderate relationships (0.3-0.5 correlation coefficient/Cramer’s V) or relationships based on domain knowledge are cross-examined to find potential patterns in missing values. For example:

* Patients with 0.00mm oldpeak tend to have missing values in ca because they could have been deemed to not need a fluoroscopy as their ST depression indicated no risk of heart defects.
* Patients asymptomatic for cp (displaying no chest pains) may have missing values for exang as patients without chest pain can be deemed well and may not be asked to go for the stress test compared with patients displaying pains and requiring further tests to ascertain the cause.

cp and exang were further cross-examined using a stacked bar chart to see if their values overlapped since both columns have values involving angina. For instance, if solely patients who display anginal pains for cp have exang as True (meaning they display anginal pains from exercise), then exang would be unnecessary as cp encompasses its values already.

Pre-Processing of Data

To remove distractions for the model, these columns were dropped:

1. id – The id of the patient has no bearing on their condition.
2. sex – Though men tend to be more vulnerable to heart disease, the risk for women can be comparable after menopause [2], the age at which they experience this depending on their hormones which cannot be determined from the data. Furthermore, as the male sex vastly dominated the dataset (80% representation), the model may be misled to link men to heart disease and ignore female patients.
3. dataset – Where the patient was tested has nothing to do with their risk.

num’s values 0 and 1-4 were binned to 0 and 1 respectively. This is because binary classification provides more distinct output and the values 2-4 in num did not provide much insight into how the level of disease is more severe. Moreover, binning values 2-4 into 1 made the target column far more balanced.

For clarity, the following columns were renamed based on the data dictionary given in Kaggle:

|  |  |
| --- | --- |
| Original Column | Renamed Column |
| cp | chest\_pain\_type |
| trestbps | top\_rest\_bps  This was renamed after observing the range of values in the column and finding that it suits the range for systolic blood pressure[2], ie, top blood pressure in addition to the name starting with ‘t’. |
| chol | cholesterol |
| fbs | high\_fasting\_blood\_sugar |
| thalch | max\_heart\_rate |
| exang | exercise\_induced\_angina |
| oldpeak | st\_depression |
| ca | colored\_vessels |
| thal | thalassemia |
| num | has\_heart\_disease |

For columns top\_rest\_bps and cholesterol, values of 0 were observed which are invalid because a person’s blood pressure cannot be 0 unless their heart stops beating [7], and cholesterol cannot be 0 as it is an essential substance for the body’s cell membranes, hormones, and vitamins [8]. Therefore, rows with these values were removed.

There was a typo in a value for thalassemia as well--reversable defect--which was corrected to ‘reversible defect’.

After checking the percentage of rows with missing values relative to the present total number of rows, columns with missing percentages of 5% or below had their rows with missing values dropped as the proportion was small and would not affect the model’s performance much. This is preferred over having to impute even more columns’ values also because it can retain more of the true distribution of data.

The columns whose rows with missing values were removed are:

* cholesterol
* high\_fasting\_blood\_sugar
* restecg

Imputation is reserved for columns with over 5% missing values. The table below shows the imputation methods used for each column with missing values:

|  |  |  |
| --- | --- | --- |
| Column | Imputation Method | Justification |
| top\_rest\_bps | Median | The column has outliers and displays little correlation with other numeric columns for there to be a relationship for K-Nearest Neighbors to rely on.  Using median imputation will give a moderate value unaffected by the outliers. |
| max\_heart\_rate | K-Nearest Neighbors | It is moderately correlated with age, giving K-Nearest Neighbors a relationship to take advantage of. |
| exercise\_induced\_angina | K-Nearest Neighbors | It is moderately correlated with chest\_pain\_type, slope, and thalassemia, giving K-Nearest Neighbors multiple relationships to rely on.  Moreover, most missing values appear when patient have asymptomatic chest\_pain\_type, meaning they show no chest pains, and when their chest pain is non-anginal. This could be because patients without angina are not tested further for exercise-induced angina whereas patients with angina are tested to verify the cause. Since the data was not missing at random either, K-Nearest Neighbors can leverage this pattern for more accurate imputing. |
| st\_depression | Median | The column has little correlation with other numeric columns and has outliers, so K-Nearest Neighbors has little relationships to utilise, and the mean may be affected by the outliers.  Thus, median imputation is used. |
| slope | K-Nearest Neighbors | It is moderately correlated with exercise\_induced\_angina and most missing values occur when exercise\_induced\_angina is false and when chest\_pain\_type is asymptomatic or atypical angina. This could mean patients exhibiting no angina or non-exercise-induced angina do proceed to have their slope checked as angina is usually a symptom of heart disease. [9]  Thus, the data has relationships to utilise and the data not being missing at random creates patterns for K-Nearest Neighbors. |
| colored\_vessels | K-Nearest Neighbors | colored\_vessels is moderately correlated with age.  Most missing values in colored\_vessels occur when chest\_pain\_type is asymptomatic and when st\_depression is 0.0mm. This suggests that patients without chest pains (a symptom of heart disease [2]) and with no ST depression are deemed not at risk of heart disease [15] (an ST depression typically indicates myocardial ischemia which can be caused by heart disease [16]), so they do not undergo fluoroscopy as a further test.  Hence, K-Nearest Neighbors has both a relationship and pattern to rely on for more accurate imputation. |
| thalassemia | K-Nearest Neighbors | thalassemia has moderate correlation with exercise\_induced angina, and missing values occur mainly when exercise\_induced\_angina is False (possibly because patients who have never been tested for thalassemia, which leads to weakness or even heart defects [19], are usually healthy and are unlikely to suffer from angina during exercise), providing a relationship and pattern for K-Nearest neighbors to leverage. |

Prior to K-Nearest Neighbors (KNN) imputation, categorical columns were one-hot encoded to retain their nominal relationship in numeric form that KNN accepts. After imputation, the imputed DataFrame’s encoded columns are converted back to their original categorical columns. It uses a process where for each categorical column, their encoded columns are searched for which contains the highest value, which is either 1 if the column had no missing values or the highest value if it was imputed. This encoded column’s name was then used to retrieve the original categorical value. This step ensures accurate restoration of categorical values, as KNN imputation returns the mean, which for one-hot encoded columns results in values between 0 and 1 that are displayed as 0 when in encoded form.

After the above steps, the target column (has\_heart\_disease) was separated from the DataFrame, where it was assigned to variable ‘y’ while the remaining DataFrame of features was assigned to ‘X’. 2 rows were then extracted from X and the corresponding targets of these rows were extracted from y using their index. The rows extracted were verified to display different target values as they will be used to test the deployed machine learning model, so a varied representation is preferred.

Afterwards, y and X were split such that 80% was a test set and 20% was a training set.

Pipelines for each model to be used were later created to consistently conduct one-hot encoding and normalisation of training and testing data separately before fitting the model to the training data and predicting the test data. The pipeline also fits the scaler on the training data only when using .fit(0 before using it to transform the training and test data separately when using .predict(). A robust scaler is used for normalisation as it does well with outliers which the final dataset has, and so that values are treated with fair weights.

Methods and Improvements

Each model is defined within a pipeline, allowing one-hot encoding, then normalisation to occur consistently before prediction whenever the model is used.

Here is an analysis of the algorithms used for model building:

|  |  |  |  |
| --- | --- | --- | --- |
| Algorithm | Description | Pros | Cons |
| Logistic Regression | Logistic Regression, commonly used for binary classification, outputs probabilities of each outcome.  Probabilities below 0.5 result in output 0, and probabilities above output 1.  It is chosen as it suits the purpose of the model in predicting binary values. [20] | * Uses regularisation to prevent over-fitting. * Data has little to no multicollinearity which is suitable for logistic regression. [21] | * Assumes linearity between features and target which may not be true. * Cannot grasp more complex relationships. [21]   To illustrate this flaw, high\_fasting\_blood\_sugar does not directly lead to heart disease but can lead to diabetes which increases risk of heart disease. Logistic regression may not be able to grasp this. |
| Naïve Bayes | Specifically, Gaussian Naïve Bayes is used.  It is an algorithm that assumes each feature to be independent of each other and that outputs a probability. [22] | * Can be used for datasets with high dimensionality as it does not try to find a relationship between the features. This works for my dataset which has many columns after encoding due to the many categorical columns. [22] * Simple as there is little hyper-parameter tuning required. | * Assumes all features have same weight. * Gaussian Naïve Bayes specifically assumes the distribution to be normal. This is the case for some, but not all columns, such as high\_fasting\_blood\_sugar. [22] |
| Random Forest | It adopts ensemble learning, using multiple decision trees trained on a random sample of the training data. These samples are sampled with replacement.  The predictions of all trees are then combined to produce a final prediction, which for classification is the class most predicted by all the trees. [23] | * Can handle outliers since not all trees will be affected, which is good for my final dataset that still has outliers present. * Less affected by noise as it uses the predictions of all trees, and not all will be affected by noise. [23] | * Vulnerable to overfitting, resulting in worse testing performance, especially if there are too many trees or the trees are too deep which may be the case for my high-dimensional dataset. * Time-consuming as each tree must make a prediction, which will be exacerbated by the current dataset which has 705 rows as it may result in more trees making predictions. [23] |
| Histogram-Based Gradient Boosting | It is a gradient boosting algorithm that builds trees sequentially, where the next tree corrects the errors of the previous tree. [25]  Although it supports missing values [24], some columns (colored\_vessels, thalassemia) had far too many missing values for decent prediction. | * Scales well to large datasets as it bins numerical values, which is good for my dataset of hundreds of rows. * Uses regularisation to reduce over-fitting. [24] | * Requires more hyper-parameter tuning, such as determining the number of bins, for optimal model performance. [24] |
| Support Vector Classification | It finds the hyperplane that best separates the classes. This finds the best decision boundary such that the model can decide more easily which class the instance belongs to. [26] | * Scales well to larger datasets as it determines the dimensionality of the hyperplane by the dimensionality of the data, more accurately modelling it. * Generalises well to new data using the optimal decision boundary found, which makes the classes more distinct. [26] | * Requires more careful hyper-parameter tuning as it uses kernel functions to transform the data, so the different types of kernel functions used can greatly affect performance. [26, 27] |

Performance Before Hyper-Parameter Tuning

The metrics for model performance evaluation used are accuracy, precision, recall, F1-score, along with a (testing) confusion matrix. Without hyperparameter tuning,

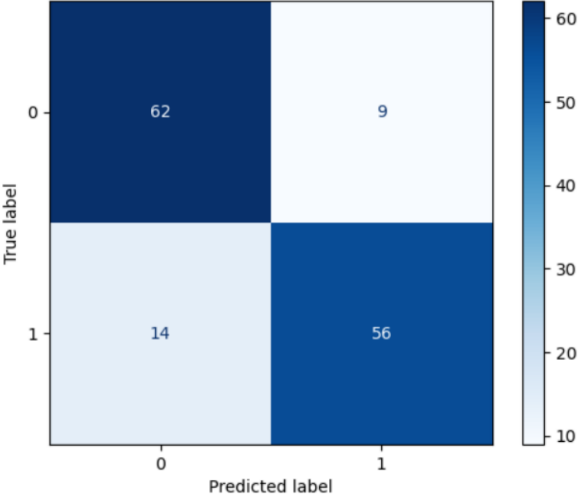
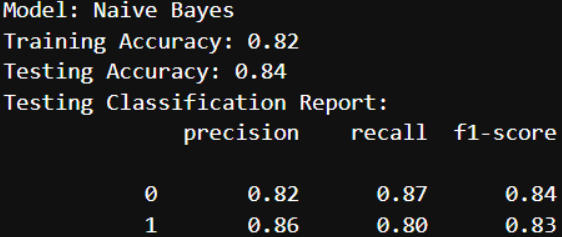
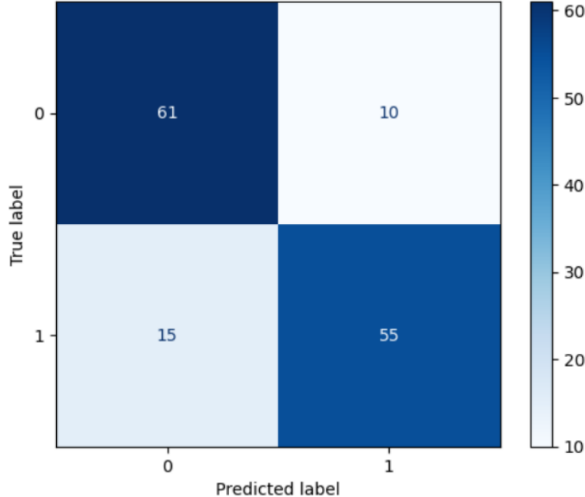
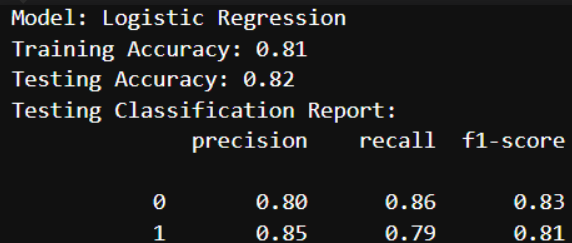


Figure 1. Logistic Regression (Before Tuning)

Figure 2. Naive Bayes

Figure 1A. Logistic Regression Confusion Matrix

Figure 2A. Naive Bayes Confusion Matrix

Figure 3. Random Forest (Before Tuning)

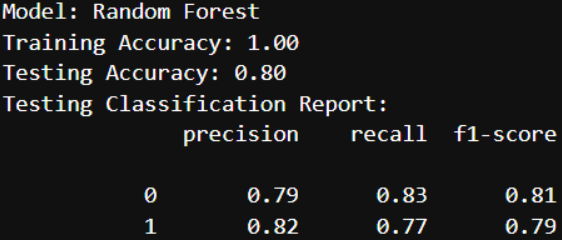
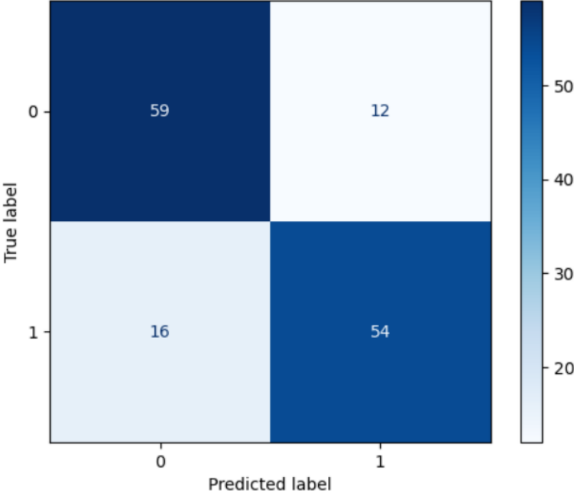
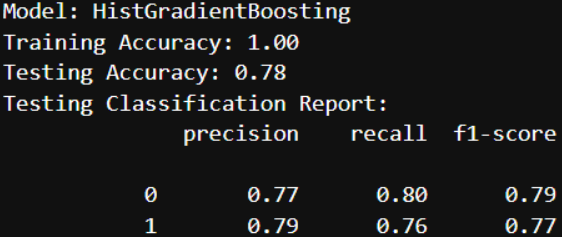


Figure 3A. Random Forest Confusion Matrix



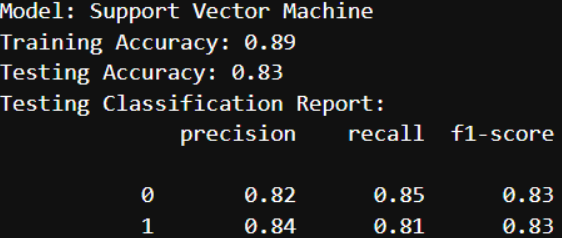
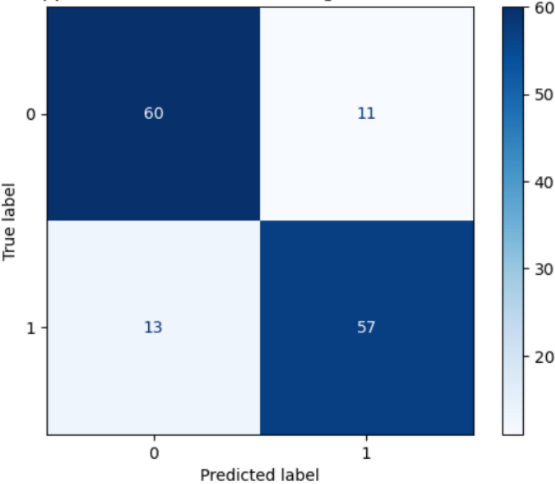
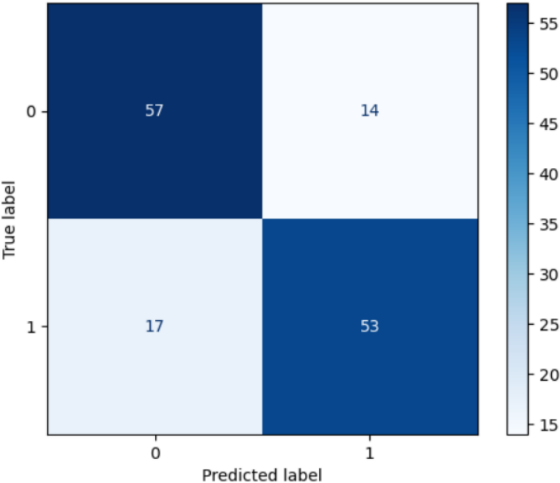


Figure 5. Support Vector Classifier (Before Tuning)

Figure 4. Histogram Gradient Boosting (Before Tuning)

Figure 4A. Histogram Gradient Boosting Confusion Matrix

Figure 5A. Support Vector Classifier Confusion Matrix



All algorithms except for Logistic Regression and Naïve Bayes displayed overfitting, likely because they use tree-based modelling and are complex in nature. Models seem to miss out patients with heart disease more often than they misdiagnose patients without as well. This suggests that the algorithms tend to be biased slightly to patients without heart disease.

For optimisation, hyperparameter turning via GridSearchCV is done to find the best possible combination for the model’s performance within the specified range below:

Logistic Regression

**C**: [0.01, 0.1, 0.5, 1] – Smaller values are specified up until the default value (1) to reduce overfitting by reducing regularisation strength.

**solver**: ['lbfgs', 'liblinear', 'newton-cholesky'] – lbfgs is kept as it is the default, whereas liblinear and newton-cholesky are used as they specialise in binary classification which this model is doing. newton-cholesky, especially, is better for datasets where there are more samples than features with one-hot encoded features that have minority categories, which are attributes my dataset has.

**max\_iter**: [100, 200, 300, 400, 500] – More iterations starting from the default (100) are specified in case the default number of iterations is insufficient to produce a satisfactory model.

[28]

Naïve Bayes

There are little hyper-parameters to set so there is no tuning. [29]

Random Forest

**n\_estimators**: [50, 100, 150, 200] – Lower number of trees along with default (100) is specified in case the model is overfitted, and more than default is specified in case more trees are needed to capture the data’s complex relationship.

**max\_depth**: [None, 4, 10, 20, 40] – Limits on depth specified to reduce overfitting, and default value (None) kept in case it is still the optimal choice.

**min\_samples\_split**: [2, 5, 8, 11, 15] – Higher values than default (2) also specified to prevent overfitting with more samples being required to qualify for a split.

**min\_samples\_leaf**: [1, 3, 5, 8, 10] – Higher values including default (1) specified to reduce overfitting as more samples will be required to be at the leaf node.

**max\_features**: ['sqrt', 'log2', 0.2, 0.3] – Values specified except for default value (sqrt) are to produce less complex trees with a smaller proportion of features considered for each split. This is to reduce overfitting.

**max\_leaf\_nodes**: [None, 20, 50, 100, 200] – Keep default value in case it is still the best option (None). Also specify limits on the number of leaf nodes in each tree to reduce overfitting by using less complex trees.

[30]

Histogram Gradient Boosting

**learning\_rate**: [0.01, 0.1, 0.2] – A lower than default (0.1) rate is specified to prevent missing the optimal weight, and a higher value is to reduce over-complication.

**max\_iter**: [50, 100, 150, 200, 300] – Same reason as in Logistic Regression, but a lower value than default (100), a shorter increment and lower range are specified to prevent overfitting if there’s too many trees.

**max\_leaf\_nodes**: [None, 31, 50, 100] – Keep default (31) should it still be the best option. ‘None’ and higher values if the model needs to be more complex to sufficiently capture relationships.

**max\_depth**: [None, 5, 10, 30, 50] – Same reasons as for Random Forest, but wider range in case the model requires more complex trees if the optimal bin count is low.

**min\_samples\_leaf**: [5, 10, 15, 20] – Lower values including default (20) also specified to ensure relationships--even after binning if the max\_bins is low--are captured.

**max\_features**: [0.1, 0.3, 0.6, 1.0] – Values specified except for default value (1.0) which are lower are used for the same reasons as for Random Forest.

**max\_bins**: [150, 200, 255] – Lower number of bins than default (255) also specified if there were too many bins, making final numeric data too complex.

[24]

Support Vector Classification

**C**: [0.1, 0.5, 1, 1.3] – Smaller values are specified up until the default value (1) to reduce overfitting by reducing regularisation strength. A larger than default value is also specified in case laxer regularisation is required.

**kernel**: ['linear', 'rbf'] – Besides the default ‘rbf’, a linear kernel is also used since it is commonly used for datasets with many features. [33]

**Gamma**: ['scale', 'auto'] – Use both possible values for wider coverage.

[32]

\* Naïve Bayes not shown as it had not undergone hyper-parameter tuning

Performance After Hyper-Parameter Tuning

The metrics used will be the same as before tuning. This time, the results also include the best hyperparameter combination from the ones earlier specified.

Figure 6. Logistic Regression (After Tuning)

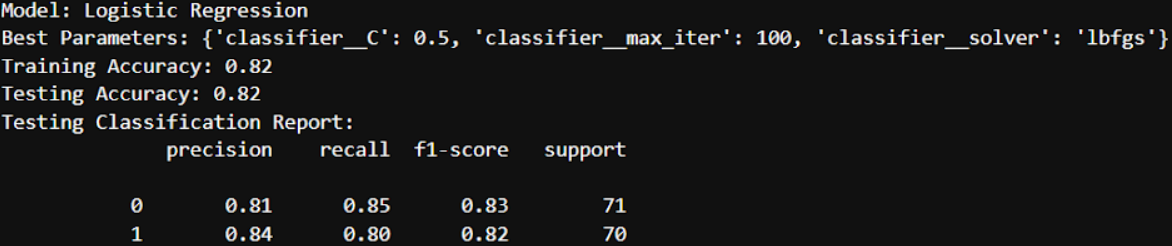
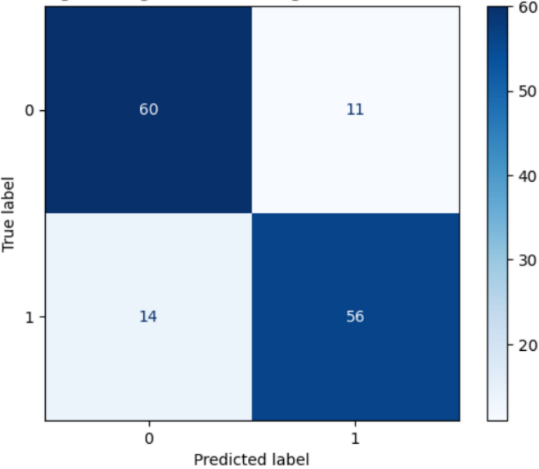


Figure 6A. Logistic Regression Confusion Matrix



Logistic Regression’s testing accuracy has not improved, but it is less biased to predicting no heart disease with less false negatives, albeit more false positives.

Figure 7. Random Forest (After Tuning)

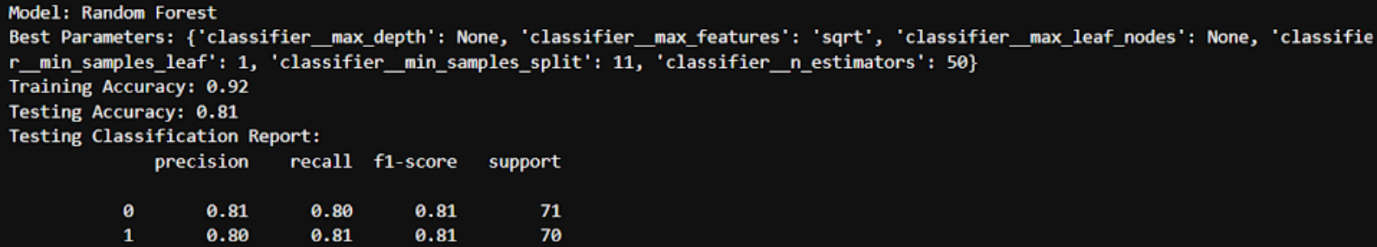
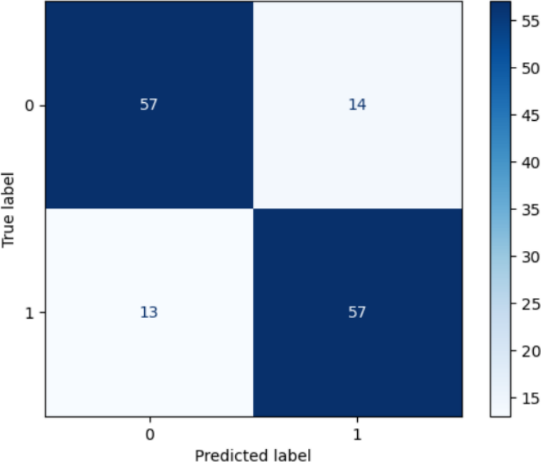


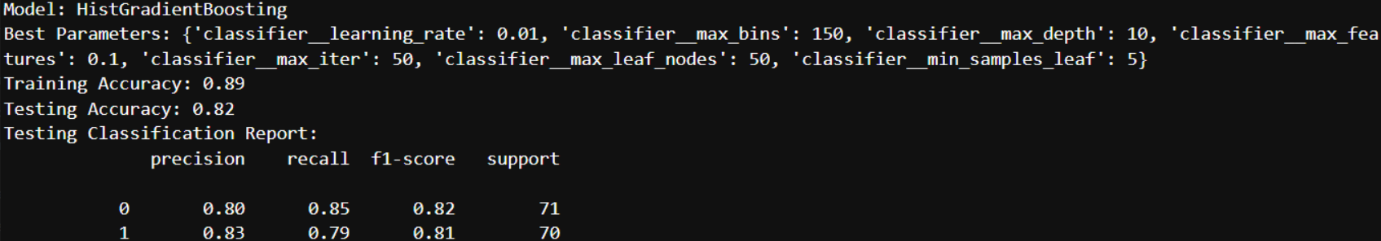
Figure 7A. Random Forest Confusion Matrix

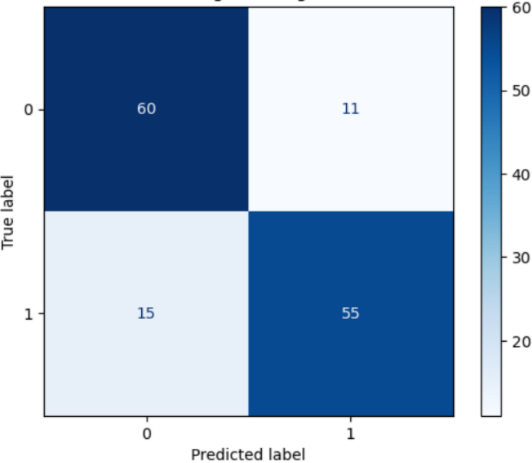


Accuracy increased slightly by 0.01, and overfitting is less with a difference of 0.11 between train and test performance compared with 0.2 before.

Like Logistic Regression, it tends towards predicting patients having heart disease now with more true and false positives as well as less true and false negatives.

Figure 8. Histogram Gradient Boosting (After Tuning)



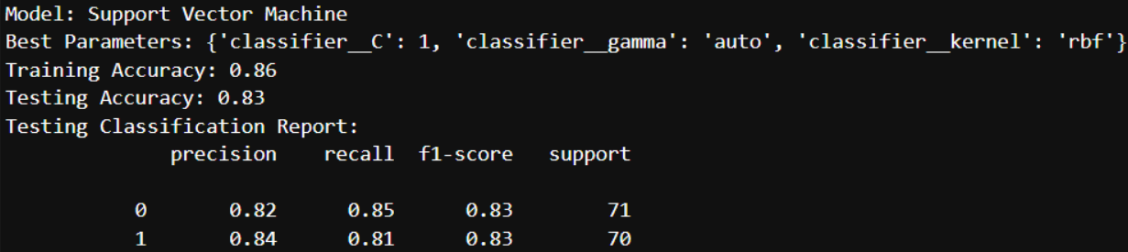


Decent accuracy increase (0.04) and less overfitting with a difference of 0.07 between performance on train and test set compared with 0.22 previously.

There are more true positives and true negatives compared to without tuning.

Figure 8A. Histogram Gradient Boosting Confusion Matrix

Figure 9. Support Vector Classifier (After Tuning)



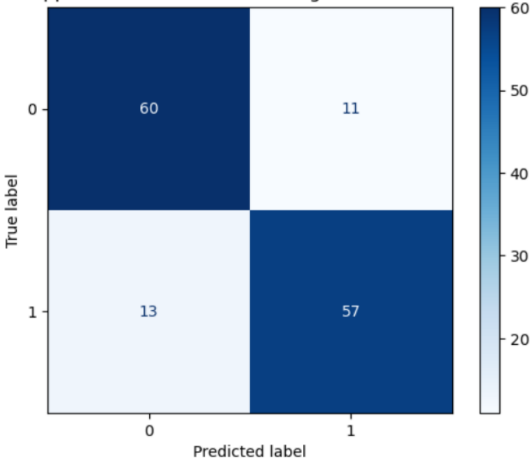
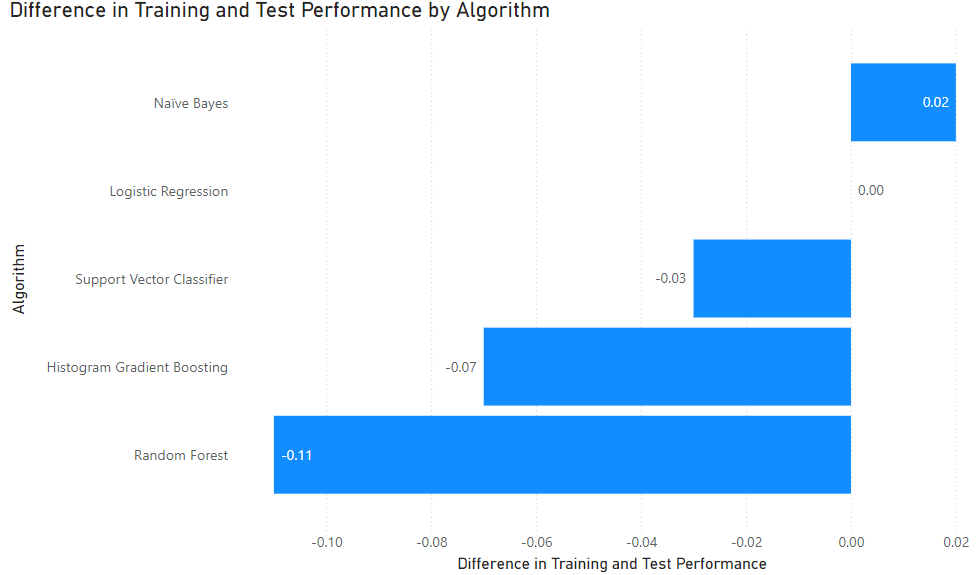


Figure 9A. Support Vector Classifier Confusion Matrix

Accuracy and other metrics have not changed, but the model generalises better now as there’s less overfitting, with the difference in training and test set performance decreasing from 0.06 without tuning to 0.03 after tuning.

Results and Analysis

The chart above of the difference in accuracy from the training set to the testing set reveals that Naïve Bayes generalises the best, as performance with the testing set improved due to increased accuracy. Meanwhile, there is no difference for Logistic Regression. For other models, they were overfit as testing accuracy is lower than training accuracy. Support Vector Classifier was the least overfit among them.

Below is a table compiling each algorithm’s test performance per metric, and revealing the best performing one for each metric:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 0 | | | 1 | | |
| Algorithm | Accuracy | Precision | Recall | F1-Score | Precision | Recall | F1-Score |
| Logistic Regression | 0.82 | 0.81 | 0.85 | 0.83 | 0.84 | 0.80 | 0.82 |
| Naïve Bayes | 0.84 | 0.82 | 0.87 | 0.84 | 0.86 | 0.80 | 0.83 |
| Random Forest | 0.81 | 0.81 | 0.80 | 0.81 | 0.80 | 0.81 | 0.81 |
| Histogram Gradient Boosting | 0.82 | 0.80 | 0.85 | 0.82 | 0.83 | 0.79 | 0.81 |
| Support Vector Classification | 0.83 | 0.82 | 0.85 | 0.83 | 0.84 | 0.81 | 0.83 |
| Best Algorithm(s) | Naïve Bayes | Naïve Bayes, Support Vector Classification | Naïve Bayes | Naïve Bayes | Naïve Bayes | Random Forest, Support Vector Classification | Naïve Bayes, Support Vector Classification |

All models perform relatively well, with accuracies and all other metrics 0.80 or above, besides recall for 1 class in histogram gradient boosting. This means that they arrive at the correct predictions around 80% of the time. They also tend to predict 0, with higher F1-Scores for class 0 than 1, apart from Random Forest and Support Vector Classification whose F1-Scores for both classes are equal.

Naïve Bayes and Support Vector Classification are the best performing algorithms, with Support Vector Classification lagging very slightly behind Naïve Bayes. In terms of accuracy, there is only a 0.01 difference.

Though Naïve Bayes is the best scorer for many metrics, Support Vector Classification was chosen for deployment. This is because Support Vector Classification has the most balanced F1-Scores for 0 and 1 classes, with both scores being the exact same (0.83). This makes Support Vector Classification the algorithm with the least bias as it performs equally well for both classes.

Furthermore, Support Vector Classification has a higher recall for class 1, meaning that there are less false negatives (less people with heart disease being predicted to not have the disease).

False negatives are more serious in the context of heart disease prediction, as it delays further tests and treatments for the patients since they are deemed healthy enough and dismissed, resulting in their conditions becoming more severe. For false positives, patients will likely undergo further tests instead, which will not affect the patient as badly.

The correlation matrices for Support Vector Classification and Naïve Bayes (Figures 9A and 2A respectively) corroborate these by showing that Support Vector Classification produces fewer false negatives (13) than Naïve Bayes (14). The predicted values are also more balanced. There were 11 false positives and 13 false negatives, as well as 57 true positives and 60 true negatives from Support Vector Classification. The values are close, showing how balanced the algorithm’s predictions are. For Naïve Bayes, there were 9 false positives and 14 false negatives, with 56 true positives and 62 true negatives. The differences are slightly more, indicating Naïve Bayes’ predictions are less balanced, and more inclined to predict negative, leading to more false negatives.

Conclusion

Heart disease is a broad and complex topic, as various ailments and defects fall under this umbrella, and because of the various factors that come into play. Thus, dimensionality reduction was difficult, and it would have been difficult for the models to find the relationships between the various columns (which was probably why Naïve Bayes performed so well) because many relationships are indirect. For example, certain types of slope would indicate Myocardial Ischemia, which can be related to heart disease but may not as well, likewise for high\_fasting\_blood\_sugar as it mainly indicates diabetes, but diabetes also puts someone at higher risk of heart disease.

Data handling for this dataset was also challenging as comprehensive, standardised clinical data is hard to obtain because certain patients may not undergo the required tests since they can be invasive, such as fluoroscopy hence the many missing values in colored\_vessels, or the patients have conditions preventing them from taking specific tests like the stress test, resulting in missing values for max\_heart\_rate and exercise\_induced\_angina. Furthermore, the data can be imbalanced and skewed because some conditions like thalassemia are rare, and patients being tested for heart disease are those already at risk such as older individuals. As some patients have rare conditions, this results in medical data having many outliers as well.

Nevertheless, using the pre-processing steps along with hyper-parameter tuning, decent models were built with the select algorithms. For Support Vector Classification which was chosen for deployment, an accuracy of 0.83 with fairly consistent and high results (above 0.8) in metrics for both classes, allowing for less biased predictions. The machine learning model which ultimately uses Support vector Classification can be used as a baseline to expedite diagnosis by flagging patients at potentially higher risk so these patients are prioritised for further tests and care, but its predictions should not be the final decision as it can still produce inaccurate results, and it cannot exactly understand the complex relationships of human biology and physiology.

References:

[1] Heart Disease Dataset - <https://www.kaggle.com/datasets/haiderrasoolqadri/heart-disease-dataset-uci>

[2] Heart Disease - <https://www.mayoclinic.org/diseases-conditions/heart-disease/symptoms-causes/syc-20353118#:~:text=Heart%20disease%20describes%20a%20range,born%20with%20(congenital%20heart%20defects)>

[3] Stress Test - <https://my.clevelandclinic.org/health/diagnostics/16984-exercise-stress-test>

[4] oldpeak (st\_depression) unit - <https://ecgwaves.com/st-segment-normal-abnormal-depression-elevation-causes/#:~:text=ST%20segment%20deviation%20(elevation%2C%20depression,conditions%2C%20particularly%20acute%20myocardial%20ischemia>.

[5] Cramer's V - <https://www.statology.org/cramers-v-in-python/>

[6] Contingency Table - <https://www.investopedia.com/terms/c/chi-square-statistic.asp>

[7] trestbps (top\_rest\_bps) - <https://www.betterhealth.vic.gov.au/health/conditionsandtreatments/blood-pressure#what-blood-pressure-numbers-mean>

[8] Importance of Cholesterol <https://www.hopkinsmedicine.org/health/conditions-and-diseases/high-cholesterol/cholesterol-in-the-blood#:~:text=Cholesterol%20is%20a%20fat%2Dlike,the%20cholesterol%20your%20body%20needs>.

[9] typical angina (stable angina), exang (exercise\_induced\_angina) - <https://www.mountsinai.org/health-library/diseases-conditions/stable-angina#:~:text=Stable%20angina%20is%20chest%20pain,%2C%20atria%2C%20arteries%20and%20veins>.

[10] atypical angina, exang (exercise\_induced\_angina) - <https://my.clevelandclinic.org/health/symptoms/24935-atypical-chest-pain>

[11] exang (exercise\_induced\_angina) - <https://www.mayoclinic.org/diseases-conditions/angina/symptoms-causes/syc-20369373>

[12] chol (cholesterol)- <https://www.medicalnewstoday.com/articles/321519#:~:text=Serum%20cholesterol%20can%20give%20an,a%20heart%20attack%20or%20stroke>.

[13] fbs (high\_fasting\_blood\_sugar) - <https://my.clevelandclinic.org/health/diagnostics/21952-fasting-blood-sugar>

[14] fbs (high\_fasting\_blood\_sugar) - <https://www.hopkinsmedicine.org/health/wellness-and-prevention/abcs-of-knowing-your-heart-risk#:~:text=Being%20overweight%20or%20obese%20is,increase%20the%20formation%20of%20plaque>.

[15] oldpeak (st\_depression)- <https://www.medicalnewstoday.com/articles/st-depression-on-ecg#what-it-is>

[16] Myocardial Ischemia - <https://my.clevelandclinic.org/health/diseases/17848-myocardial-ischemia>

[17] slope - <https://litfl.com/st-segment-ecg-library/>

[18] ca (colored\_vessels) - <https://www.urmc.rochester.edu/encyclopedia/content.aspx?contenttypeid=92&contentid=P07662>

[19] thal (thalassemia) - <https://www.mayoclinic.org/diseases-conditions/thalassemia/symptoms-causes/syc-20354995>

[20] Logistic Regression - <https://www.ibm.com/topics/logistic-regression>

[21] Logistic Regression - <https://www.geeksforgeeks.org/advantages-and-disadvantages-of-logistic-regression/>

[22] Naïve Bayes - <https://www.ibm.com/topics/naive-bayes#:~:text=Na%C3%AFve%20Bayes%20is%20part%20of,important%20to%20differentiate%20between%20classes>.

[23] Random Forest - <https://www.geeksforgeeks.org/what-are-the-advantages-and-disadvantages-of-random-forest/>

[24] Histogram-Based Gradient Boosting - <https://scikit-learn.org/stable/modules/ensemble.html#histogram-based-gradient-boosting>

[25] Histogram-Based Gradient Boosting - <https://machinelearningmastery.com/histogram-based-gradient-boosting-ensembles/>

[26] Support Vector Machine - <https://www.ibm.com/topics/support-vector-machine>

[27] Support Vector Machine - <https://scikit-learn.org/stable/modules/svm.html>

[28] Logistic Regression - <https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html>

[29] Naïve Bayes - <https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html#sklearn.naive_bayes.GaussianNB>

[30] Random Forest - <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html>

[31] Support Vector Classifier - <https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC>

[32] Support Vector Classifier – <https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC>

[33] Linear Kernel - <https://www.geeksforgeeks.org/creating-linear-kernel-svm-in-python/>