Heart Disease Classification

ECS 171 Machine Learning

Group 24 Project Report

Group Members:

Russell Umboh, Konsing Ham Lopez, Aarush Agrawal, Grace Loh, Chen Li (Charlie)

GitHub Repository:

https://github.com/LeUmbono/ECS171-Group24-Project

Introduction and background

As the leading cause of death worldwide, claiming 17.9 million lives annually, cardiovascular disease (CVD), more commonly known as heart disease, is a clear and present concern for the healthcare industry at large. Around 80% of CVD fatalities occur as a result of heart attacks and strokes, and a third of these deaths are from people under 70 years of age. Although some risk factors for CVD are genetic in nature, many others are directly correlated to one's lifestyle, most notably high blood pressure, high cholesterol and smoking. The ability to identify heart disease accurately, therefore, may also highlight exactly which factors are most likely to have caused the occurrence of the disease in those affected, serving as motivation for others to alter their personal habits to reduce their risk of CVD.

Those with pre-existing conditions (such as hypertension, diabetes, hyperlipidaemia, etc.) which may aggravate the risk of cardiovascular disease deserve a comprehensive and accessible method of detecting and managing heart disease; this can most definitely be achieved with a well-trained machine learning model.

If a concrete and reliable classification model for classifying heart disease can be derived, it would present a clear benefit in the following fields:

- Clinical purposes
  - Early and efficient detection of heart disease would ensure that proper treatment of the condition can be delivered in a timely fashion to the affected patient. This would possibly reduce the lethality of CVD, given that patients would be aware of their condition and seek treatment at an earlier point in the disease. Statistics regarding CVD could also be better recorded, allowing medical practitioners greater insight into the symptoms and consequences of the disease, and thus more effective treatment.
- Economic benefits
  - By improving the detection of heart disease through machine learning, people affected or at risk of CVD would take greater precaution in monitoring their lifestyles and conducting medical check-ups to keep track of their health. This would prevent them from being as indisposed due to the disease and therefore increase their productivity at work. Companies can also use the detection model to screen their employees or potential recruits for CVD risk and thus maintain a healthier workforce through better healthcare coverage, etc.
- Public awareness of cardiovascular disease
  - Effective detection of heart disease as a result of machine learning may result in more accurate statistics about the occurrence of the disease. This may incentivize politicians or medical agencies to promote educating the public about the disease in an attempt to ameliorate its effects.

Literature review

Machine learning offers robust tools for medical prognostic models, particularly in predicting heart disease, an area where early detection can significantly impact patient outcomes. This review briefly examines the application of three computational models in heart disease prediction.

In their research, Ambrish G. et al. used logistic regression to predict cardiovascular diseases. They utilized a heart disease dataset from the UCI ML repository, containing 13 features and 303 records. Their approach achieved a prediction accuracy of 87.10% in heart disease.

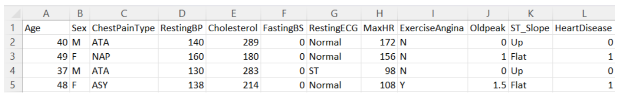
In another study, Patel, Tanvi S., et al. explored the use of Support Vector Machine (SVM) and Naive Bayes algorithms for predicting heart disease and survivability. Their model employed a clinically validated dataset with 16 attributes from the University of California, Irvine's Centre for Machine Learning and Intelligent Systems. The SVM model reached an accuracy of 88% in heart survivability prediction.

Finally, Sharma, V., et al. conducted a study using Deep Neural Networks (DNN) for heart disease prediction. They used the Cleveland Heart Disease Dataset from the UCI ML repository, consisting of 303 instances and 14 features. Their research achieved an accuracy of 81.9% in predicting heart disease.

By analyzing these studies, we gain insights into the application and effectiveness of logistic regression, SVM, and DNN in the field of heart disease prediction, providing valuable benchmarks for our own model.

Dataset description and exploratory data analysis of the dataset

The dataset we have acquired contains 918 observations and 12 attributes: age, sex, chest pain type, resting blood pressure, cholesterol, fasting blood pressure, resting ECG, max heart rate, exercise angina, old peak, ST slope, and heart disease. Because of the nature of our dataset, we had to divide it into 3 different groups that separate the categorical, numerical, and binary features. The categorical data consists of features that produce varying data categories. An example would be the chest pain type, which includes typical angina (TA), atypical angina (ATA), non-anginal pain (NAP), and asymptomatic (ASY). The numerical data consists of features that yield numeric values, and the binary data are features that produce bivariable outcomes.



	A	B	C	D	E	F	G	H	I	J	K	L
1	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease
2	40	M	ATA	140	289	0	Normal	172	N	0	Up	0
3	49	F	NAP	160	180	0	Normal	156	N	1	Flat	1
4	37	M	ATA	130	283	0	ST	98	N	0	Up	0
5	48	F	ASY	138	214	0	Normal	108	Y	1.5	Flat	1

Figure 1: A subsection of our dataset.

After dividing our data into different segments, we aimed to assess any potential data elimination by using the chi-square method to determine the relationship between each of the categorical attributes and heart disease. All of the results came out to have a small p-value and large chi-square value so no eliminations were needed for this segment of the data. After that, we were able to use one-hot encoding on the categorical data. For the numerical data, we used a box plot and calculation of IQR to determine if there are any outliers. We then convert the outliers into the median value of each data attribute to make the data more consistent.

Once data processing was completed, we constructed heat maps utilizing numerical, categorical features, and the heart disease attribute. Our analysis revealed an absence of linear correlations among the columns. Consequently, we conducted feature selection by segregating the data into three subsets based on correlations at approximately ± 0.15, ± 0.25, and ± 0.35, respectively. The ± 0.25 range was specifically utilized for a distribution plot due to its stronger correlation with heart disease. Examining this plot, we observed a predominance of males among those with heart disease, with a female-to-male ratio of 2:7. Notably, there was a significant peak in the 'oldpeak' attribute. Regarding chest pain types, the plot highlighted that approximately half of the individuals displayed no symptoms, while the remaining half were distributed across other chest pain types. Additionally, we generated a pair plot to explore correlations between features, ultimately concluding an absence of linear correlation between any two features.

Proposed methodology

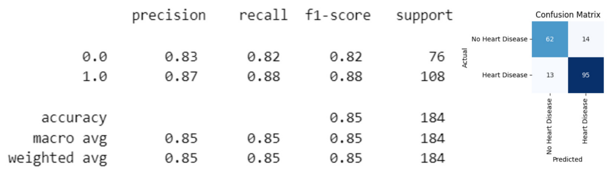
The problem of classifying heart disease is a binary classification problem. This is because a person can either have heart disease, or they do not have heart disease. Based on this, we picked a couple of models that can solve classification problems. The models we picked were logistic regression, support vector machine (SVM), and a neural network (ANN). We felt that trying these models would give us a great chance at accurately predicting heart disease given the popularity of the models and our familiarity with them.

Our dataset was taken from Kaggle, and had a very high dataset score, which meant that the data was relatively straightforward to use. Because of this, there was little to no time spent on data cleaning. The only data preprocessing we ended up doing was using a one-hot encoding for categorical features, normalizing the numerical features, and changing the outliers in the data to their median values. We had to use one-hot encoding because the types of models we ended up using needed numerical values, so the only way to represent categorical variables in a numerical sense would be to use one-hot encoding. Additionally, some duplicate features such as Sex_Female and Sex_Male in the dataset were removed because there is no point in having both features.

After the initial preprocessing, we needed to decide what features to use in the model. We wanted to make sure that the features that we were using had sizeable correlation to heart disease to give us the best chance of accurately predicting future cases. To do this, we used a correlation matrix to tell us information about how important each feature is when predicting heart disease. We decided to have three different groups of features based on the correlation matrix, features with a correlation greater than 0.15, 0.25, and 0.35. Then, we displayed the classification report for the training and testing data (to see if there was overfitting), and the confusion matrix for each model. We also did 10-fold cross validation in order to see if there was any overfitting of the initial model. Finally, we used hyperparameter tuning on the neural networks before displaying the model's accuracy. Based on the accuracy results of the models, we then chose a model to be the best he one that would make the actual predictions on our webpage.

Experimental results

Logistic regression:

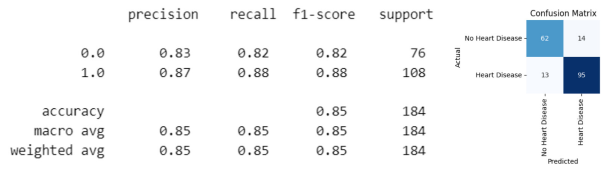


```
              precision    recall  f1-score   support

         0.0       0.83      0.82      0.82        76
         1.0       0.87      0.88      0.88       108

    accuracy                           0.85       184
   macro avg       0.85      0.85      0.85       184
weighted avg       0.85      0.85      0.85       184
```

Figure 2: Testing result of Logistic Regression Model for 0.15-threshold data.

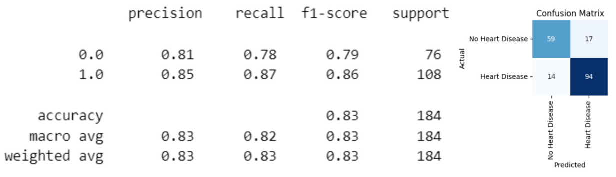


```
              precision    recall  f1-score   support

         0.0       0.83      0.82      0.82        76
         1.0       0.87      0.88      0.88       108

    accuracy                           0.85       184
   macro avg       0.85      0.85      0.85       184
weighted avg       0.85      0.85      0.85       184
```

Figure 3: Testing result of Logistic Regression Model for 0.25-threshold data.



```
              precision    recall  f1-score   support

         0.0       0.81      0.78      0.79        76
         1.0       0.85      0.87      0.86       108

    accuracy                           0.83       184
   macro avg       0.83      0.82      0.83       184
weighted avg       0.83      0.83      0.83       184
```

Figure 4: Testing result of Logistic Regression Model for 0.35-threshold data.

For logistic regression, we built the models based on scikit-learn's LogisticRegression classifier, using the default parameters. With this approach, L2 regularization is automatically performed on the model to obtain the best weights and biases through the L-BFGS optimization algorithm. This reduces overfitting by limiting the influence of any one single feature of the data. A logistic regression model was built for each threshold dataset obtained, specifically containing features that had an absolute correlation value of 0.15, 0.25 and 0.35 with the target feature (HeartDisease) respectively.

Based on the classification reports, confusion matrices and 10-cross validation for each logistic regression model, we can see that the best models appear to be the models for 0.15-threshold dataset and the 0.25-threshold dataset with an average accuracy of 84.51% after cross validation was performed. The classification reports for these models also report a higher accuracy of 85% vs 83% for the 0.35-threshold model when run on a single instance of testing data. Similarly, higher precision, recall and f1-scores were reported for the former models as well. This may be explained by the fact that the limited number of features captured by the 0.35-threshold model was not as effective as the other models in reflecting the true nature of the target variable despite the higher correlation values of its features, thereby resulting in a higher number of incorrect classifications.

SVM:

```
              precision    recall  f1-score   support

         0.0       0.90      0.83      0.86        76
         1.0       0.89      0.94      0.91       108

    accuracy                           0.89       184
   macro avg       0.89      0.88      0.89       184
weighted avg       0.89      0.89      0.89       184
```

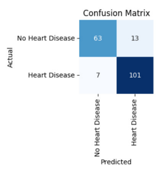


Figure 5: Testing result of SVM Model for 0.15-threshold data.

```
              precision    recall  f1-score   support

         0.0       0.90      0.83      0.86        76
         1.0       0.89      0.94      0.91       108

    accuracy                           0.89       184
   macro avg       0.89      0.88      0.89       184
weighted avg       0.89      0.89      0.89       184
```

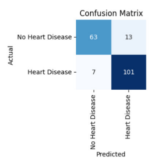


Figure 6: Testing result of SVM Model for 0.25-threshold data.

```
              precision    recall  f1-score   support

         0.0       0.85      0.76      0.81        76
         1.0       0.84      0.91      0.88       108

    accuracy                           0.85       184
   macro avg       0.85      0.84      0.84       184
weighted avg       0.85      0.85      0.85       184
```

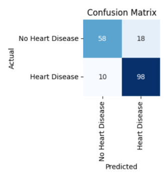


Figure 7: Testing result of SVM Model for 0.35-threshold data.

For SVM, we built the models based on scikit-learn's SVC classifier, using the radial basis function (RBF) kernel. This ensured that we could find Support Vector Machines for our datasets in infinite dimensions, ensuring that the most accurate models are obtained. All other parameters were set to their default values. An SVM model was built for each threshold dataset obtained, specifically containing features that had an absolute correlation value of 0.15, 0.25 and 0.35 with the target feature (HeartDisease) respectively.

Based on the classification reports, confusion matrices and 10-cross validation for each logistic regression model, we can see that the best model appears to be the model for the 0.25-threshold dataset with an average accuracy of 84.19% after cross validation was performed. In this case, this may be the result of the 0.15-threshold data containing too many poorly-correlated features to accurately train the SVM model to detect heart disease whereas the 0.35-threshold, much like with the logistic regression models, lacked enough relevant features to train the model well.

Compared to the average accuracies obtained by the corresponding logistic regression models for each threshold, we see that they are lower for the SVM models than the Logistic Regression models. This is interesting because we would expect these values to be higher for SVM models as they are more complex than the relatively simple Logistic Regression model. In fact, the classification metrics obtained from the SVM models operating on the same splits of training vs. testing data do indeed show that accuracy for these models are higher than the Logistic Regression models. Thus, the lower average accuracy for SVM obtained from cross validation on the SVM models may indicate a higher degree of overfitting compared to the Logistic regression models, given their poorer performance on different splits of data.

ANN:

```
              precision    recall  f1-score   support

         0.0       0.88      0.83      0.85        76
         1.0       0.88      0.92      0.90       108

    accuracy                           0.88       184
   macro avg       0.88      0.87      0.88       184
weighted avg       0.88      0.88      0.88       184
```

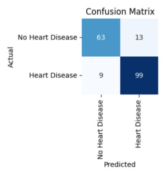
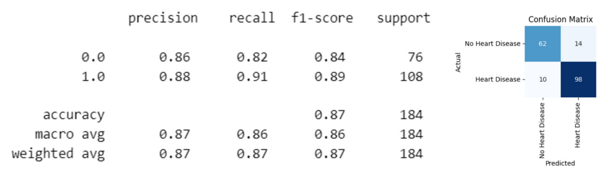


Figure 8: Testing result of ANN Model for 0.15-threshold data.

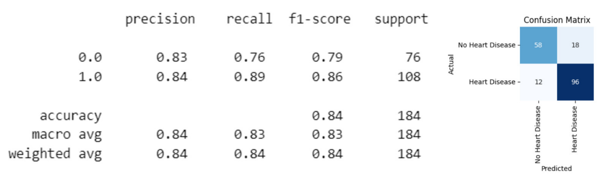


```
              precision    recall  f1-score   support

         0.0       0.86      0.82      0.84        76
         1.0       0.88      0.91      0.89       108

    accuracy                           0.87       184
   macro avg       0.87      0.86      0.86       184
weighted avg       0.87      0.87      0.87       184
```

Figure 9: Testing result of ANN Model for 0.25-threshold data.



```
              precision    recall  f1-score   support

         0.0       0.83      0.76      0.79        76
         1.0       0.84      0.89      0.86       108

    accuracy                           0.84       184
   macro avg       0.84      0.83      0.83       184
weighted avg       0.84      0.84      0.84       184
```

Figure 10: Testing result of ANN Model for 0.35-threshold data.

For our neural networks, we initially started out with scikit-learn's MLPClassifier function, with our solver as stochastic gradient descent, activation function as sigmoid (for binary classification), an initial learning rate of 0.3, batch size of 100, maximum iterations of 500 and two hidden layers consisting of 7 neurons each. Using hyperparameter tuning via Grid Search, we found the optimal hyperparameters for maximum iterations, learning rates and hidden layer neurons. We tried varying the hidden layers based on the number of layers and the size of each layer, so as to get a comprehensive grasp over which factors worked best for our models.

Looking at our results, the best neural network model appears to be the model for the 0.15-threshold dataset, with an average accuracy of 85.3% after cross validation. Based on hyperparameter tuning, the optimal values for the model's hyperparameters are as follows: hidden layer sizes of (7, 7), an initial learning rate of 0.45 and max iterations of 500. Something to note is that although the classification reports of the SVM models report higher accuracies for the same training-test splits, their lower average accuracies further strengthens the idea that our SVM models suffer from overfitting. On the other hand, the lower accuracies (both in the classification reports and the average accuracy after cross validation) of our logistic regression models are as expected; the neural network is a much more complex model and as such is expected to generate a more accurate classification for heart disease.

Software implementation:

The heart of the application lies in its ability to predict heart disease using user-input health data. It is constructed using Flask, a Python web framework, which makes it suitable for handling web requests and delivering dynamic web content. The website offers a form where users can input various health parameters such as age, sex, chest pain type, resting blood pressure, cholesterol levels, fasting blood sugar, and more. This data is then used to predict the likelihood of heart disease.

The front-end of the website, defined in the index.html file, is structured using HTML, styled with CSS, and made interactive with JavaScript. The layout is straightforward, presenting a form to input health data, alongside buttons for submitting the data, generating random test data, clearing the form, and an option to automate multiple predictions. This design ensures ease of use, catering to users with different levels of technical proficiency.

The back-end, managed by app.py, handles data processing, model predictions, and server-side logic. The application converts user input into a format compatible with the machine learning model. Categorical variables are encoded, and data columns are reordered as per the model's requirements, ensuring accurate data interpretation by the model. The model itself, loaded and used in model.py, is a pre-trained machine learning model, responsible for making predictions based on the processed input data.

The integration of machine learning is a critical aspect of this application. The Python script model.py uses the joblib library to load the pre-trained model and pandas for data manipulation. When a prediction request is made, the script processes the input data and feeds it to the model, which then returns a prediction about the presence of heart disease.

In conclusion, this web application demonstrates an effective use of web technologies and machine learning to address a critical health concern. Its integration of front-end and back-end technologies, along with a machine learning model, provides a seamless and interactive experience for users seeking to assess their risk of heart disease.

Conclusion and discussion

Overall, our models performed relatively well when compared to the model's performance in literature reviews. The ANN that we had slightly outperformed the NN used in Sharma, Rasool, and Hajela (85.3% to 81.9%), but the SVM that we had compared to the SVM in Patel et. al was slightly worse (84.19% vs 88%) as well as the logistic regression model in Ganesh et al. (84.51% vs 87.10%). Regardless, the models were decently accurate enough to predict heart disease when compared to the literature.

Something that we could have done differently is to try using all the features in the model regardless of their correlation to heart disease. When training the model, we ignored using features that had a low correlation to the output, but this could have resulted in a lower accuracy because we might have generalized too much and ignored valuable information that could have helped. Ultimately, something that we can do next is keep experimenting with different groups of features.

Project Roadmap

Week 2: Project Theme Decision and Dataset Selection

- **Tasks:** Search for heart disease-related datasets on Kaggle to determine the feasibility of the project theme. Given that cardiovascular diseases (CVDs) are the leading cause of death globally, the focus will be on finding a high-quality dataset with rich features and a substantial sample size.
- **Activities:**
  - Conducted a search on Kaggle for datasets related to heart disease.
  - Evaluated the quality of the Heart Failure Prediction Dataset found on Kaggle, a comprehensive dataset combining multiple independent datasets (like Cleveland, Hungarian, Switzerland, etc.) totaling 918 observations.
  - Analyzed the dataset's attribute information, such as age, sex, chest pain type, blood pressure, etc., and how they correlated with heart disease prediction.
- **Outcome**: Selected the Heart Failure Prediction Dataset as the project dataset and decided on heart disease classification as the project theme based on the dataset's characteristics. Prepare a report detailing the chosen dataset's source, features, and suitability for the project.

Week 3: Literature Review

- **Tasks:** Conduct an in-depth study of relevant academic papers and research, focusing on studies by Ambrish G. et al., Patel, Tanvi S. et al., and Sharma, V. et al.
- **Outcome**: A comprehensive literature review report summarizing existing research and methodologies.

Week 4-5: Dataset Understanding and Exploratory Data Analysis

- **Tasks:** Acquire and process the dataset and perform preliminary data exploration, including data cleaning, categorization, and fundamental statistical analysis.
- **Outcome**: Detailed description and exploratory analysis results of the dataset.

Week 6-7: Developing Accurate Prediction Models

- **Tasks:** Build and train models using logistic regression, SVM, and ANN.

- **Outcome**: Initial model construction and performance evaluation.

Week 8: Model Evaluation and Performance Testing

- **Tasks:** Conduct in-depth evaluation and performance testing of the models, including cross-validation and hyperparameter tuning.
- **Outcome**: Model evaluation report and suggestions for improvement.

Week 9-10: Webpage Front-End Development and Project Summary

- **Tasks (First Half):** Design and implement the web front-end, integrating the model.
- **Tasks (Second Half):** Compile the project summary report and prepare for the presentation.
- **Outcome**: Initial version of the web application and the project summary report.

References

Dataset:

https://www.kaggle.com/datasets/fedesoriano/heart-failure-prediction

Literature Review:

Ambrish G., Bharathi Ganesh, Anitha Ganesh, Chetana Srinivas, Dhanraj, Kiran Mensinkal. ”Logistic regression technique for prediction of cardiovascular disease.” Global Transitions Proceedings, vol. 3, no. 1, 2022, pp. 127-130. ISSN 2666-285X, doi:10.1016/j.gltp.2022.04.008.

Patel, Tanvi S., Daxesh P. Patel, Mallika Sanyal, and Pranav S. Shrivastav. ”Prediction of Heart Disease and Survivability using Support Vector Machine and Naive Bayes Algorithm.” bioRxiv, 2023, doi:10.1101/2023.06.09.543776.

Sharma, V., A. Rasool, and G. Hajela. ”Prediction of Heart Disease Using DNN.” 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA), Coimbatore, India, 2020, pp. 554-562, doi: 10.1109/ICIRCA48905.2020.9182991.