FIT3164 Final Report

HEART DISEASE PREDICTIVE MODEL

Luke Wilson (20643292), Utkarsh Patel (29143926),
and Joshua Fehring (28802691)

Word Count: 8841

Table of contents

1. Introduction
2. Background
    - 2.1. Feature Selection
    - 2.2. Predictive Models
    - 2.3. Recent Advances
3. Methodology
    - 3.1. Feature Selection
    - 3.2. Model Development
    - 3.3. User Interface Development
    - 3.4. Output Fuzzification
4. Project Management
    - 4.1. Approach
    - 4.2. Execution
    - 4.3. Management and Planning
    - 4.4. Limitations
    - 4.5. Critical Analysis
5. Outcomes
    - 5.1. Results
    - 5.2. Limitations
    - 5.3. Improvements and Future Directions
    - 5.4. Critical Analysis
6. Conclusion
7. References
8. Appendix

1. Introduction

Coronary Heart Disease (CHD) causes millions of deaths each year around the world according to the World Health Organisation (WHO). Part of the reason it has such a high mortality rate is that the best way to control it is with early diagnosis and treatment. Diagnosis is currently a technical and involved process however, and this limits the amount of people who can receive definitive diagnoses, and how early they can receive them. To combat this, this project aims to develop a program that allows input of a patient's data into a predictive model for diagnosing CHD to assess the likelihood of CHD being present in suspected sufferers. This program would be used as a tool for preliminary diagnosis of patients suspected of having CHD so that doctors can more accurately refer at-risk patients for further diagnosis. The software would be designed for use by a doctor, and feature a user interface (UI) that would allow input of patient data and return a prediction of the likelihood that the patient has CHD as the output. Both patients and doctors would benefit from such a program as it would allow doctors to make more informed decisions regarding diagnosis of CHD, and would allow patients to be diagnosed earlier and with greater confidence.

2. Background

CHD causes millions of deaths each year, making it the number one cause of death globally according to the WHO. This is in part because there is no cure for CHD, so once diagnosed, treatment must be undertaken to control symptoms. Because of this, early diagnosis is extremely important at slowing the disease onset and improving outcomes for patients (Kramer, Schlößler, Träger, & Donner-Banzhoff, 2012). Unfortunately, diagnosis of CDH can be invasive and complex, with the current leading method for diagnosis being an invasive and costly procedure called a coronary angiogram. Due to the technical nature of such a procedure, it must be performed by qualified technicians and can be quite costly, meaning that it cannot be used unless there is a significant suspicion of CHD, and in some instances may not be available to a patient at all due to financial status or because of a lack of available qualified technicians. The lack of accessible diagnostic tools undoubtedly contributes to the high rate of CHD fatalities worldwide, so development of such a tool is of great importance. Among the most promising methods of diagnosis that has emerged in this area is the use of predictive models developed through machine learning on large bodies of existing data on patient symptoms to determine the likelihood that a given patient is suffering from CHD (Krishnaiah, Narsimha, & Chandra, 2016).

Predictive models are an ideal solution for the lack of easy and affordable diagnosis of CHD, since a predictive model could be used to generate an accurate diagnosis very quickly and without any technical expertise beyond inputting the patient's data and interpreting the result, provided all the requisite patient data is available. The accessibility this affords, both in cost and convenience, to patients around the world could significantly improve CHD outcomes across the world since a diagnosis via predictive model could be employed far more liberally than a coronary angiogram, and so increased rates of diagnostic testing would lead to a higher proportion of CHD sufferers being diagnosed early and being able to manage the disease. Even using a predictive model as a preliminary test before sending the patient to confirm the diagnosis through a coronary angiogram would likely lead to a higher survival

rate. A predictive model would also be available to much more of the population since it lacks the possibly prohibitive cost and technical expertise, so it would be an invaluable tool in scenarios where a coronary angiogram is not a feasible option. No predictive model that is robust, effective, and accurate enough for widespread use has yet been developed, researchers have made significant headway developing them using numerous machine learning techniques (Dwivedi, 2018).

Some of the techniques that have been employed to predict the presence of CHD in a patient are development of Random Forest models (Jabbar et al., 2016), Artificial Neural Network (ANN) models (Alizadehsani et al., 2013; Dwivedi, 2018), and Support Vector Machine (SVM) models (Alizadehsani et al., 2013; Ciecholewski, 2013). This project aims to build on these methods in order to develop an improved CHD predictive model that can provide robust, accurate, and effective diagnosis of the presence of CHD in patients.

2.1. Feature Selection

Feature selection is an important tool in the development of accurate predictive models (Wieland, Kerkow, Früh, Kampen & Walther, 2017). Feature selection algorithms identify features in the data that are good predictors of the target variable, and exclude extraneous or unnecessary variables. Including a smaller number of features in the model reduces errors and noise in the model, and can help prevent overfitting. Excluding too many features however reduces the performance of the model, so finding the optimal number of features to include, and selecting those features, is an important part of model development.

There are many methods for feature selection, and each has advantages and disadvantages. Most feature selection methods are designed to be best suited for specific types of models or analysis, but generally they fall into three classes; Filter, Wrapper, and Embedded methods. Most feature selection methods fall into one of these classes or are a hybrid of them (Labani, Moradi, Ahmadizar, & Jalili, 2018).

Wrapper Methods

Wrapper methods use the interactions and effects of features on the specific predictive model they are being selected for to select a subset of features. Because wrapper methods take into account the effect of features on the model and model dynamics, they take into account interactions between features better than most other types of methods, and as such are often a good choice for finding the optimal feature set for a given dataset (Saeys, Inza, & Larrañaga, 2007). Being model specific does have some drawbacks however, and wrapper methods do have a higher risk of overfitting to a model than other methods, as well as being more complex and computationally intensive.

Filter Methods

Filter methods work by starting with all of the features in the dataset, or a chosen subset, then iteratively reducing the number of features by eliminating features based on relationships within the data, such as information gain, variance, or correlation of features. Filter methods also fall within two sub-categories: Univariate and multivariate filter methods.

Univariate filter methods assess the interaction between the data and each feature individually, while multivariate methods take into account multiple features at once. Univariate methods are usually quite simple and fail to take into account features that interact with each other, but have a very low computational cost. Multivariate methods however, although able to better account for interactions and correlations between variables and are simpler than wrapper methods, can scale badly with large datasets.

Embedded Methods

Embedded methods are feature selections that are built into the specific predictive model being developed. Tree-based models such as Random Forests often have embedded feature selection, and because they are integrated with the model they are embedded in, embedded methods can often take into account interactions between features and the model, making them effective feature selection methods within the models they are limited to (Jiang, Zhang, Yu, & Wang, 2019).

The features included in the model will have a large effect on its performance, and as such selecting the best feature selection for the developed model is of great importance.

2.2 Predictive models

Model selection is also extremely important in the development of a predictive model, since there are advantages and disadvantages to all models. Model selection must be done after considering the strengths and weaknesses of different models, and the desired properties of the model you are aiming to develop. For the purposes of this project, three main candidate models have been chosen based on the current literature on CHD prediction: Random Forest, Artificial Neural Network, and Support Vector Machine models.

Candidate Models

Random Forest (RF)
A Random Forest (RF) model is an ensemble model that creates multiple decision trees using different branches and features, then calculates predicted values using the votes of these trees using some measure of centrality. Since decision trees are used in generating the prediction in an RF model, it can capture nonlinear relationships in data, but is not particularly suitable for capturing linear relationships in data. RF models can be used for classification or regression, and can handle both continuous and classification data, making it a decent choice for many datasets (Cutler, Cutler, & Stevens, 2012). Major benefits of RF models include good accuracy when compared with other models for many different types of data, and an effective in-built feature selection methods in the form of the construction of multiple decision trees that can be used to identify important features in the data. They are also relatively resistant to missing values or outliers in the data due to their branching structure. The major disadvantages of RF models are their computational complexity and their lack of interpretability in the prediction process due to the creation and evaluation of many different trees over a centralised, interpretable tree or algorithm.

Artificial Neural Network (ANN)

Artificial Neural Networks (ANN) are a machine learning model with architecture based off of biological neural network architecture, hence the name “Artificial Neural Network”. ANNs generally consist of structured layers of neurons, with each neuron processing a continuous signal level dependent on the signals from surrounding axonal links to other neurons (Kim & Kang, 2017). The first layer of the ANN network consists of a number of neurons, each representing a single input variable normalised to a value between 0 and 1. Because ANNs are essentially blind to the nature of the input beyond the values of the input neurons, numerical data must be normalised to between 0 and 1, and categorical data must be made into a series of dummy variables for interpretation by the ANN. Between the input and output layers, layers of hidden neurons are trained to fit the model to the data. The greater the number of hidden neuron layers, the more the model will fit the data. Increased complexity does increase the risk of overfitting the model to the data as well as increasing computational load.

Support Vector Machine (SVM)
Support Vector Machines (SVMs) are a type of model that classifies objects by separating them into groups along n-dimensional hyperplanes, where n is equal to the number of predictive features in the model (Noble, 2006). The hyperplanes act as categorical boundaries for the data, so data points can be classified by which hyperplane partition it falls into. Selection of hyperplanes is critical to the performance of SVMs, and different methods of defining hyperplanes can yield very different results. SVMs are very good at handling complex data with a large number of features, but are limited to classification models and lack interpretability, making them potentially difficult to assess and optimise.

2.3 Recent Advances

Machine learning approaches to CHD diagnosis have been an active area of research for a number of years. Notably, 2020 a paper by Reddy *et al.* proposed the use of a feature selection algorithm that employs a genetic algorithm using fuzzy logic to find the optimal feature subset for prediction of heart disease. Fuzzy logic refers to the categorisation of data entries into three or more classes. In the aforementioned paper, “fuzzification” of numerical data into “high”, “medium”, and “low” classifications. This fuzzification of this data allows more refined feature selection due to the more well defined boundaries between values in the data. Fuzzification can also help with interpretability, as grouping continuous or ambiguous variables into easy to understand groups can help a user understand the output of a predictive model. Using genetic algorithms for feature selection is another approach that has been a popular one when tackling the problem of CHD prediction (Gokulnath & Shantharajah, 2019; Jasuja, 2020; Mafarja & Mirjalili, 2018). Genetic algorithms are a family of heuristic algorithms that work by iteratively creating subsets, then “breeding” the best performing subsets together by combining aspects of them together, mimicking the process of natural selection and creating increasingly better performing algorithms until an “optimal” selection is reached. Development of this type of algorithm is promising for future models of CHD diagnosis.

3. Methodology

During the planning phase of the project, we set forward a set of primary acceptability requirements for the deliverables. To quote:

> The HCP end user will be able to enter the minimal amount of (non-identifiable) data to derive a risk assessment of a patient's likelihood to present with coronary artery disease (CAD). The system does not and can not diagnose, but instead offers a probabilistic output for the HCP user to apply in conjunction with their own skills and expertise to make well-informed healthcare decisions in the interests of the patient.

From the Requirements Traceability Matrix (RTM):

1. Model having the highest sensitivity value
2. Model with interpretable outputs
3. User Interface should be simple and clean
4. Model should be able to perform well on unseen data, and potentially adjust the model if necessary

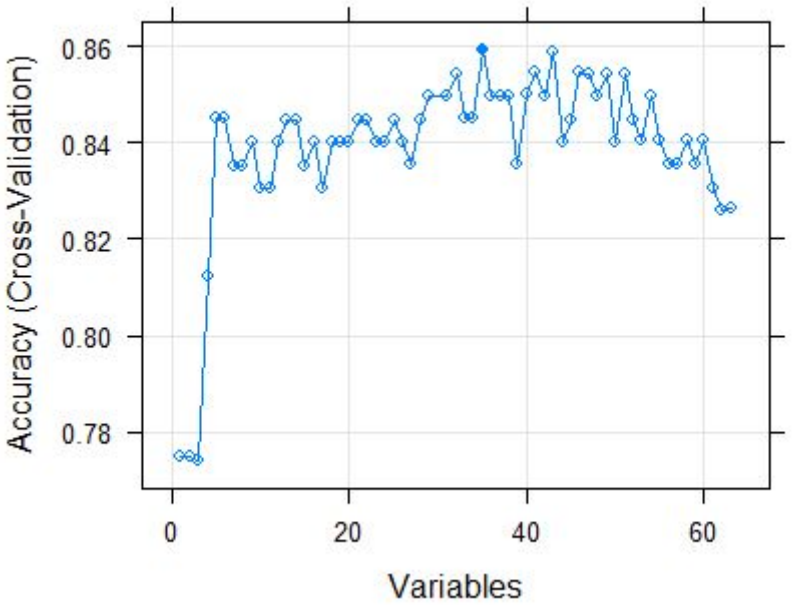
The methodology of the project was therefore designed to meet these requirements. This can be broken down into a short list:

1. Feature selection
2. Model development
3. User interface development
4. Output Fuzzification
5. Generalisability

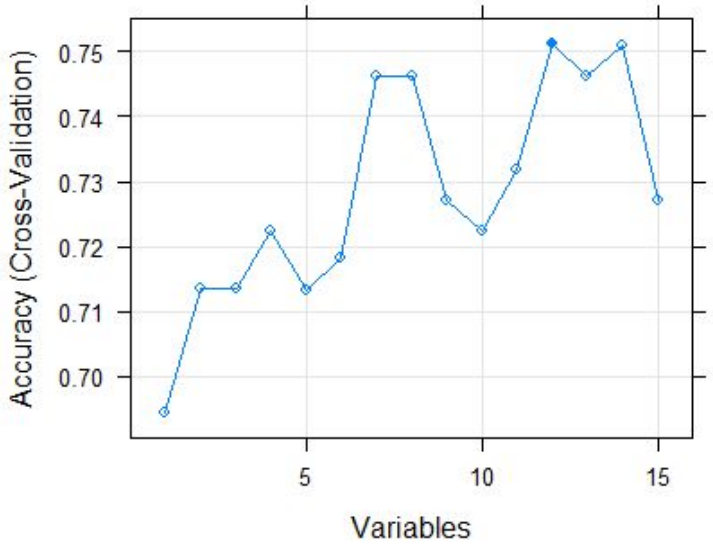
Our testing methodology was therefore designed to test whether these requirements had been met. Testing was performed manually, and thankfully due to the robust and well-tested caret and shiny R packages, testing was really only required for unit tests of local functions, integration of parts, and testing and validation of user input.

3.1. Feature selection

After processing of the dataset, feature analysis was performed in a number of stages. First, we began with the Random Forest model, as it conveniently produces a reduced feature set as a function of its training. However, the size of the ideal feature set needed to be determined firs; what is the number of features that would produce the best performance across all three models? Recursive Feature Elimination (RFE) is a method by which the optimal size of the subset of features is determined. Performing this with all available features gave us:



The accuracy of the model grew explosively for the first few subset sizes, but once reaching approximately 0.84 accuracy, it oscillated ± 0.02 seemingly randomly. Accuracy peaked at approx. 0.86 on subsets of 34 randomly selected features. For our purposes, a feature set this large was counterproductive, and we therefore determined to focus on the optimal reduced feature set size less that was less than 15, where the greatest gains in accuracy were seen in the RFE above.



From here, we were able to determine that the ideal reduced feature set size, at least with a Random Forest model, was 12. Which features to include in that reduced feature set was determined with all three models in mind.

Three different models were trained using the entire feature set (Random Forest, State Vector Machine, and Neural Net), and each model produces a scalable weight to each

feature which describes the relative strength of the feature within the model. From this, an aggregated average “score” for the “importance” of each feature was produced across the three models, and ordered, and the top twelve features in predicting the presence of Coronary Artery Disease (CAD) across all three models are listed below. As it turned out, this reduced feature set performed remarkably well on the other models as well.

	NN	RF	SVM	Score
Typical.Chest.Pain	5.599680618	4.94474207	0.8077545	1.0000000
Age	3.146284324	3.74826230	0.7145168	0.7348234
BMI	4.126219107	1.51998233	0.6415552	0.6128353
Atypical	1.626821588	2.76252027	0.7078840	0.5751863
TG	3.195934984	2.05840192	0.5818054	0.5690971
BP	1.894864380	2.52937636	0.6818917	0.5646994
EF.TTE	2.035789556	2.32253521	0.6820535	0.5592116
ESR	2.059159734	2.41803333	0.6273727	0.5444755
HTN	1.637539316	1.96902333	0.6831320	0.5121191
Nonanginal	2.958525817	1.31129805	0.5671915	0.4985706
PR	2.346687684	1.49819568	0.5993313	0.4880117
Tinversion	2.261236224	1.30252000	0.6383736	0.4858456

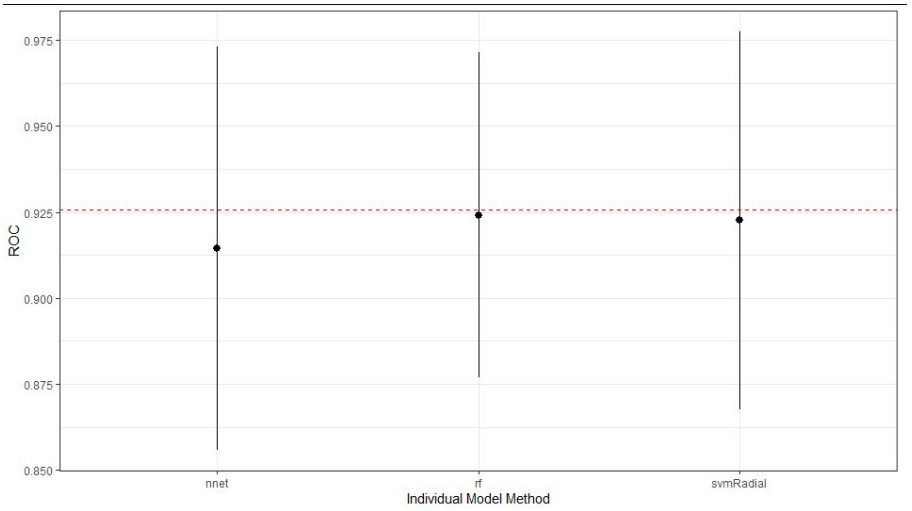
3.2. Model development

Models were then trained using the reduced feature sets, and assessed for their accuracy and sensitivity. The models performed similarly well, but the correlation between models below showed that an ensemble might be ideal as it could capture the strengths of each model while smoothing the variation and gaps between them.

```
                rf svmRadial      nnet
rf        1.0000000 0.7131334 0.3633429
svmRadial 0.7131334 1.0000000 0.5567156
nnet      0.3633429 0.5567156 1.0000000
```

Finally, an ensemble of the three models was produced using a greedy algorithm where the AUC of each model was the metric to be greedily optimised. The optimal weightings for the models in the greedy ensemble were:

```
The following models were ensembled: rf, svmRadial, nnet
They were weighted:
3.5594 -5.0522 0.1144 -2.5826
The resulting ROC is: 0.9256
The fit for each individual model on the ROC is:
    method       ROC      ROCSD
        rf 0.9242097 0.04718138
 svmRadial 0.9226290 0.05491660
      nnet 0.9145304 0.05866590
```



From the above diagrams, it can be seen that the ROC of each individual model was pretty similar, but the ROC for the ensemble was slightly higher.

3.3. User Interface Development

The user interface was designed to be as simple as possible. Specifically, input methods were designed to minimise the possibility of invalid or erroneous input. Checkboxes were used for binary inputs, radio buttons for categorical data, and text entry boxes only for real values that cannot be handled with any of the above.

3.4. Output fuzzification

When making a classification, our model does so by producing a probability that a given observation would present with coronary artery disease, given the past data that the model has seen. A simple fuzzification process is used to convert this raw value into something a little more intuitive for the end user. It should be noted that no members of this team are medically inclined, and so while we did our best to make a reasonable attempt at defining the ranges for the probability fields within our model and if this project were to be extended, it would

4. Project Management

All members in our team have used statistical and probabilistic analysis to develop the system known as user interface which can flag those individuals in specific risk categories. The main object of this UI is to help identify the high-risk individuals to be further investigated by the medical team for a potential diagnosis. Considering the consequences of the wrong prediction made just in case, by the system, our team has prioritized sensitivity over specificity value while selecting the model after testing and comparing a variety of models.

The developed user interface for the deployable model intends to communicate to the end-user to enter non-identifiable patient data for the same model to perform analysis and produce an output that is interpretable. The output is probabilistic, and it is interpretable into a range of categories. Also, the input data entered by the user needs to be in the format as expected by the model which can be found on the UI page. For any further details, the UI page can be accessed from the code report document.

4.1. Approach

It was a traditional approach which was used for most of the project cycle as the team had planned to implement. As the stages of the tasks were defined clearly, it made it easier for the members in our team to associate with each other and we met each deadline with collective approach as well as self-skills developed individually in the process. The COVID situation never allowed any of us to meet in-person which has been the biggest setback, but we managed to schedule meetings via video conferencing regularly as planned before. With not many changes taking place for the duration of the project, the group managed to run the project on-time. Hence, the traditional approach for the team worked out very well for the project cycle.

Scope:

The team member developed User Interface (UI) to show the risk and probability of patients with the provided features using CHD variables. The internet, hardware and software charges were met in the scope for the duration of the project.

Research collaboration was done in weekly meetings. Group members distributed with tasks and milestones according to work breakdown structure which allowed the members to get done with tasks individually before making decisions. It includes development, testing, deployments, algorithms etc.

Requirements Traceability Matrix: CAD Predictive Model

Project Name:	CAD Predictive Model
Project Manager Name:	FIT-3164 Staff
Project Description:	Aiming to develop a classification model that, given several observations and features, flags those observations that most confer the risk of angina and delivers risk assessment.

ID	***Requirements (Functional or Non-Functional)***	***Assumption(s) and/or Customer Need(s)***	***Category***	***Source***	**Status**
1	Laptops	To share information with team members	Technical	Personal	met
2	Internet	To communicate with team members	Technical	Personal	met
3	Cleaned data	To easily train the model	Technical	Data preprocessing	met
4.	Imputation	To allow missing values to be replaced when it's necessary	Technical	Data preprocessing	Not met and not necessary anymore with the nature of data
5	Model with Interpretable outputs	To explore the model further for testing	Non-Technical	Training algorithm	met
6	Model having the highest sensitivity value	To get the best model to be outputted	Technical	Testing algorithm	met
7	Web Page / UI thread	To make it available for doctors to testing	Technical	Output part	met

8	Visualisation	Models especially probabilistic outputs and performance evaluation may be more meaningful if they are visualised	Technical	Output	met
9	*Extensibility*	System should be able to perform well on unseen data and adjust the model if data changes in any ways	Technical	Processing	met
10	Google drive	To collaborate team documents	Non-Technical	University module	met
11	Required programming languages on device	All team members are expected to have programming languages such as R, python, HTML, CSS, MySQL, etc on their devices to work with models smoothly and GIT to handle version control	Non-Technical	Personal	Met but only R was required and additionally, Shiney package was used rather than CSS, HTML

4.2. Execution

Resources and tools required:

- Google Drive
    - Google drive doc management handled in proper manner
- GitHub: [To avoid losing individual work-loss risk]
    - Code committing into relevant repositories

- R-Studio and R-Shiny [UI development]
- Social media for regular discussions
- Zoom video conference for all meetings
- Asana project management
    - Used to set up the deadlines [to avoid the deadline meeting risk]

Resources management:

- Three team members, and one mentor.
    - Required research articles and materials regarding candidate feature selection algorithms, and classification models, for comparative purposes, and forewarning of model-specific
- Standard research and development of algorithms and models: minimum 8 hours per week, per team member.
- Project meeting: 2 hours per week
- Peer learning session in workshop: 2 hours per week
- Hardware: Require one workstation per team member, including microphone for digital
- meetings, and a webcam is desirable but not necessary.

4.3. Management and Planning

As outlined in the flow map cycle of the project, it was followed with the schedule of the tasks as defined in the project proposal document. The process was followed with the Asana project management tool. With the functionality of the software, the team was able to be in the loop of achieving those tasks.

By the end of the proposal document, the team had a Gantt chart which describes the timeline of the project with task description. However, that document required us to look back on the chart for the deadline reminder for the specific task. The team as a result decided to switch to the online system which can inform automatically on what day the task is due, what task it is and the assignee for the task. The team decided to adapt Asana project management tools which fulfils all the requirements suggested as above. By reminding the team member and assignee about the deadline on a regular basis, the system makes sure the team does not miss out on any submissions and that would help prevent the deadline risk of our team.

It is understandable that the communication risk for the team can be crucial for the entire project. Our team wanted to make sure that the team members do not suffer from miscommunication within the team environment and even if it does happen, our plan was to stay in touch with creative and justifiable discussions to resolve all queries. Therefore, the team created the social media platform for all members to keep in touch and to regularly discuss points and ideas whenever and wherever it was necessary. This strategy helped the team gain confidence towards the execution of the project as a team. The team members discussed their research and their ideas on social media informally before bringing it forward for the meeting. The social media platform has also helped us to schedule the meeting timings and quick discussion for any deadline. In short, having the team connected on social media has been a key decision to avoid the communication risk in most ways, if not all.

For all aspects of the project, there will be lots of files with report writing to collaborate your work with other team members. The number of submissions the team had for the entire duration of planning and execution of the project; the files always needed to be collaborative where each member can work on the same file. In case an individual loses the current working file, it can invite huge troubles as backups are not always available for all matters. Hence, To avoid individual's work losing risk, the team decided to store all files into google shared drive environment which not only allows to update the versions of the file, but also to create the folders categorically to locate files under relevant category.

Since this project is software based, the individuals were assigned tasks responsibility broadly at the planning stage. Therefore, the team members were handling coding documents individually to explore models and similar tasks followed by discussions with other members. As R script coding documents may get lost if not correctly handled, the team decided to follow the strategy of keeping weekly backups of the coding documents through GitHub repositories. The various repositories are created in the GitHub team branch to locate the files of specific tasks to correct paths. Again, losing the individual's data risk was avoided as the team kept uploading documents onto GitHub platform.

Since the COVID global pandemic has made it difficult for everyone to meet in-person, our team too suffered from the situation. As a result, the team ended up not meeting at all for the duration of project planning and execution. However, our team has made the best use of technology for meetings firstly, by social media for smaller discussions and zoom video conferences for weekly discussions of ideas generated. Basically, the team would have the points and ideas in the platform and the team makes an agenda to be followed for the minute. The minute-taker notes down the key discussions and decisions taken out of it into a minutes document which is available in the team management report. The meeting minutes helped us to reflect back on what approach and strategy the team was implementing which also allowed us to add checkpoints of our progress in the project.

More details on how team communication was maintained can be followed in the team management document.

This software project required us to do a certain amount of research before classifying the algorithm to put into practice. To better manage the process, the team distributed tasks among group members to give individuals enough time to come up with the suggested ideas weekly after doing research and exploration on concepts and algorithms. However, no one team member was solely responsible for the completion and workload of a single subsection of the project. The following leaders are assigned for the exploration part and for certain project functions, in that the function leader is the one responsible for leading the completion of a subsection of the project, including scheduling, communication, and documentation within that specified domain.

Actual workload within each function will be shared, but each member was designated leadership of specific functions. The designated functions and leaders we followed were as follows:

Joshua Fehring:	Luke Wilson:	Utkarsh Patel:
Documentation, research, and pre-processing	Feature selection, and model development	Testing, and evaluation

When the above classification model is successfully developed to a sufficient standard, then the team is to embark on the next stage of the project in planning and developing a user interface for the system.

As mentioned before, project responsibilities were taken in leadership mode where each member was in charge of making sure the specific goal was achieved. Joshua was in charge of resource collection for research and preprocessing of the data. To work with the data, the team looked for algorithms and techniques to transform the data into compatible for like factoring, cleaning, etc.

WBS schedule breakdown:

It is justifiable from the planning stage of the project that one process needs to be finished before the next one starts in this process. In the initial stage of the project, we found the datasets relevant to our project and modified the data into the form it is required to process further. This process was handled perfectly as we discovered the dataset at the planning stage of the project. As part of this process, the team communicated data using graphs available and did the wrangling process with handling null values, factoring features into correct forms etc. Here, all the steps were closely related to each other and they were

accomplished under Luke's leadership who made sure we have a dataset working in the correct format and are compatible with the model we would decide.

Within one week, the team started reviewing the literature review from the project proposal submission to get the insight on which model we should be going forward to explore with. During the meeting in early August, the decision was made about selection of models for the further exploration. The models decided were Neural Net, Random Forest, and Support Vector Machine (svm). All team members were allocated two weeks to work on these models to find out algorithms for feature selection and model development [Luke with Neural Network, Utkarsh with Random Forest, and Joshua with Support Vector Machine]. After two weeks, the focus was made on the feature selection process. It was decided that neural net by themselves do not work well for feature selection. However, SVM and RF stood with positive results. Then Joshua was allocated the responsibility to develop the algorithm to accommodate the feature selection process by ranking the features. In the meantime, Luke was allocated to find a new dataset to make sure the preprocessing and further modelling process works best for unseen datasets as well. Utkarsh's allocated task was to explore the interesting rfecontrol() method discovered to see if it can be used for implementation for the model development part. This process was stretched for two more weeks as it can be seen in meeting minutes that the discussions were happening in large amounts during those periods on how each model can complement each other to develop the feature selection as well as learning more about features in depth. At the start of mid-semester break, the algorithm was produced, and we had selected a number of features and a target on researching random forest was met. But Luke's task for the mid-semester was to explore the model development with the features extracted already and to get a rough idea on which model performs the best. The team make a decision at the start of the mid-semester break to finish exploring on the SVM model system and Joshua took responsibility for it followed by the algorithm to produce the model. The team also decided to make a sample UI such that it becomes easy to modify it later in the semester. Therefore, the pre-active step of making UI happened by the end of mid-sem which Utkarsh was allocated to. In the next week (week-8), the team took the decision to go ahead with the Random forest model which performed fairly better than other ones. Joshua took responsibility for developing the model, then Luke was in charge of testing them with other models which resulted in Random Forest being the best among all. Utkarsh was allocated to modify the UI to work better with random forest models and 12 selected features already. It was arranged well at the end of wee-9. In the next week, the code demonstration included the user script for coding methods and testing script for UI methods. Earlier in the same week, the team worked together to modify UI to add check buttons for categorical attributes and test cases which handles the error in case a user enters invalid or out of range inputs. At the end of week-11, the team finally succeeded in improving the accuracy of the model. Luke had taken responsibility for it and he made use of the ensemble method which makes use of all three models combined averaged. That way, the model performance got increased and Joshua was allocated to test the model performance in terms of accuracy, ROC curves and AUC values. Finally, Utkarsh updated the UI yet again with a better model than the previous one.

Execution: [Major milestones]

Feature selection algorithm: resources obtained from the caret and RFE methods and algorithm to rank feature in terms of importance

Model development and testing: Algorithms to explore and develop the models and [RF, ANN, Tree, SVM] testing them by accuracy and AUC value to make decision on model selection

UI: R-shiny applications information, the approach or steps followed to work with the application creation

4.4. Limitations

While the situation has stayed similar for both the planning and execution phase of the project, the team was habituated to work online with meetings since the planning stage of the project. However, the team was having difficulties with participation in classes for longer periods. The team was looking forward to more discussions with staff and a wide range of interaction with other teams which would have let us openly discuss the project with the resources point of view. However, due to the situation that innovation lacked a little. But team members managed their ways to connect online.

The team was concerned about the resource risk at the planning stage where not all of us had knowledge about topics and concepts to be used in the project. To handle these challenges of individual skills, team-members distributed tasks in-between and defined the temporary timeframe to learn about them. For example, team members got to the stage where they were lacking skills in support vector machine (SVM), how to deploy the predictive model onto a webpage, etc. To overcome this, the team took a decision to stay proactive and decided to pre-allocate time period and learn more about this content well in advance. RShiny() package team decided to work on having to be well understood and the team made use of the break period to learn and understand to implement better into the project to avoid resource risk.

The limitation was to find the dataset with research online. But it was the limitation of the project that the dataset could not be compatible with the project model component.

Team was doing great with managing the deadline risk. The main reason for being on-time was the reminders and allocation functionalities of the Asana tool. However, there were upgraded versions of the management tool available with upgrades of extra features such as additional security, rules, advanced search and reporting etc. Team did well with the handling of deadline risk but had the upgraded versions of features been available, the team could finish tasks well on time and with even better quality.

As there were no official agendas for each meeting, the team approached the following tasks to make sure members have enough to talk about during the meeting. Before each meeting, the group member would share an update on tasks achieved to see what to discuss for the meeting

4.5 Critical analysis

Team management of the project overall went vastly according to the plan. It was team members' contribution and teaching staff's feedback process which allowed us to work thoroughly and with dedication towards the project documents. The work breakdown structure we defined at the planning stage of the project helped us really well to follow along as we executed the project. It was a step by step process and the allocation to the tasks was one of the most challenging tasks as it is not always ideal to follow the computer system assigned tasks. Therefore, like the way we have been following the work breakdown diagram tasks, our target was to make sure who is working on what process at the given time. During each meeting we would have, our team might need to assign subtasks of any tasks early or later to specific team members depending on the situation. However, the timeline was strictly followed as the Gantt chart. So, as it is mentioned in the project management above, the team followed the waterfall traditional approach by sticking to the strict deadlines of project submission. But the subtasks of those submissions were the tasks which needed to be allocated to team members on specific time depending on the situation. However, the team did well with keeping up to the scheduled timelines of each stage of the project. Apart from the Gantt chart, the team decided, as per the suggestions by the staff members to use project management tools rather than sticking to the chart. It turned out to be a key plan since the team moved the schedule to Asana project management. That would remind us of the deadlines and let us allocate the tasks to group members which ended up performing well for us to always be reminded of. Therefore, the tool helped us manage better from deadline risk which was the positive move.

In the other aspects of the project, there were a number of challenges our team faced when it comes to managing the documents in the collaborative manner. The R-script coding document on GitHub and word documents on google drive had the messy folders which needed to be managed before and during the execution of the project. The result of this consumed so much time for team members to locate the files and documents rather retrieving and working on them. Finally, the team took decisions to make repository branches specifically for a variety of categories to control the versions of the coding file and create the folders in a classified way to locate documents conveniently. Therefore, the approach worked to overcome the junk of files issues. Communication was the most important element in our project. As we were moving from the planning stage to the execution phase, the team already had a social media page to discuss ideas and points on a regular basis. It allowed the team members to build confidence between each other at each stage of the process. Since we were used to zooming video conferences from the start of the planning stage of the project, it did not take too much effort to continue with the new meeting environment. However, it was necessarily important to keep track of our weekly discussed topics in case the team needs to retrieve the existing details. As part of the deliverables, the group started taking details of the key points from the first meeting of the execution stage of the project. It helps both us and staff to reflect upon what has happened in the past which makes sure all project stakeholders are on the same page. Hence, the challenge and risk of being off track for the project was better managed with the use of meeting minutes documents which took place twice a week. Additionally, the team was concerned about the resource risk at the planning stage where not all of us had knowledge about topics and concepts to be used in the project. The team hit such scenario on various stages of the project. To handle these challenges, team-members distributed tasks in-between and

defined the temporary timeframe to learn about them. For example, team got to the stage where they did not know about support vector machine (SVM) concepts which was the week before the mid-semester break. Here, the team made us of two weeks back to back weekend to learn about it and we came up together at the start of the mid-semester to present the learned model and its output. Here, we stayed pro-active and sensed that it could get even worse if no individuals know how to deploy the working model on a web server. Then the team allocated two weeks to learn the concept of RShiny() packages which allows to implement UI. Finally, team made decent use of break to get the project in the working flow and overcame the resource risk.

While there was success in producing the project, we had the failure in team management which was out of control. Due to the pandemic situation, the team could not do even a single meeting face to face as planned at the proposal stage which was subject to the situation. The big loss team suffered from when there was zero face to face interaction with the staff members. As the project is handled with deliverables, the staff played an important role by suggesting ideas and working around algorithms in the process. If situation had allowed and it could have been possible to meet face to face with staff members, the more discussions and wide range of interaction with other teams might have been enabled which would have let us openly discuss the project with the resources point of view.

5. Outcomes

5.1. Results

This project has successfully developed an accurate model for the prediction of CHD in a patient, as well as a program that allowed a prediction to be obtained from this model based on user input of patient information into a UI. This program could be used by physicians for the preliminary diagnosis of CHD to allow earlier, more accurate, and more confident referral to further confirmation of diagnosis, earlier treatment, and therefore better outcomes for patients suffering from CHD. To achieve this, a number of requirements had to be met for the model.

First of all, the model had to be developed and trained. Given the data used was already clean of missing or incorrect values, this only required the data to be preprocessed before model training could begin. To this end, an R script was written that factorised and normalised the data to ensure that the data was compatible with the various machine learning models being developed and to reduce variance between data features that could negatively affect the models. To determine which models would be used, we examined current literature in the field of CHD diagnosis prediction and decided that RF, ANN, and SVM models were the best candidates for our predictive model. Features were then selected by constructing a basic form of each model using all features in the dataset using the Caret package, then obtaining the ranking of feature importance from these models. The scores returned by the importance ranking were then normalised and averaged to obtain the ranking of features across all three models. This was done to mitigate biases of individual models in treating certain features as important, and create a more representative ranking of

features with predictive value. A Recursive Feature Elimination (RFE) was then used to find the optimal number of features, which was twelve for the final model. This led to us selecting our subset of the twelve features with the highest importance rankings. The chosen features were as follows:

Age
BMI
Blood Pressure
Typical Chest Pain
Atypical Chest Pain
Non-anginal Chest Pain
T-Inversion
Presence of Severe Vascular Heart Disease
Erythrocyte Sedimentation Rate (ESR)
Ejection Fraction
Hypertension
Triglycerides

Once features were selected, RF, ANN, and SVM models were trained, then combined into an ensemble model in order to reduce biases and overfitting to the data within single models. This model was then tested for accuracy via a number of metrics. Metrics used include constructing confusion matrices and ROC curves of the developed models. Further details of this testing can be found in the test report. This model was found to have the highest accuracy and sensitivity of all developed models, and so was chosen as the final model. Sensitivity was highly valued in this process given the disproportionate negative impact of a false negative compared with a false positive, with the former leading to a negative diagnosis for a patient with CHD, drastically reducing their probable lifespan, while the latter at worst will lead to an uncomfortable medical procedure for the patient with some out of pocket expense. This process led to the development of an accurate and sensitive model for the prediction of CHD based on a patient's symptoms.

Once the model was developed, we had to make predictions based on it accessible. The first step in this process was developing a UI that allowed the input of novel patient data into the model and would display a prediction of the patient's predicted likelihood of having CHD. This was accomplished by saving our trained model and loading it into a UI programmed by the Shiny package. The UI can take values for our twelve predictive features as input in each of twelve labelled fields, make a prediction based on the values using the loaded model, then output the results. Given the medically sensitive and technical health information required for the modol to make a prediction, this UI was designed to be used by qualified physicians to diagnose their patients, rather than for self diagnosis by patients. This meant that explanation of the information required for each field was minimal as a level of medical knowledge is expected by users of the program. Once the data was entered into the UI, the prediction had to be visualised in a readable, understandable manner. To achieve this, the output prediction probability was displayed along with a fuzzified low-extreme rating with five ratings based on the probability. A rating system accompanied by the raw probability was selected as the meaning of the probability could be ambiguous if displayed by itself, but having it displayed alongside an understandable rating would allow physicians more discretion when diagnosing patients when compared with only displaying a rating. A five tier

rating system was selected as simply providing a presence or absence prediction would be presumptuous given the nuanced factors involved in CHD diagnosis and our lack of medical expertise. On the other hand, a rating system with too many levels would lead to the same confusion about the meaning of each level as the raw output, making five levels the optimal number in our judgement. This rating and probability are displayed on the UI for interpretation by the physician and patient to allow more confident and nuanced diagnosis of CHD.

To test the ability of the UI to handle user input, we limited the possible entries into the input fields of the UI, converting all binary yes/no features into tick boxes and only allowing numeric and mathematical symbols to be input into the text fields. Erroneous inputs into the text fields were further handled by checking all required entries had been filled and were valid and outputting an error message if they had not been, and checking if the entered values were within the expected ranges of those values, and outputting a warning if they were not. Further details of the UI testing and error handling can be found in the test report.

The result of this development and testing was a robust program that allowed the input of patient data, and returned a prediction of that patient's probability of having CHD based on an accurate predictive model.

5.2. Limitations

While the outcomes of our project were very positive, there are limitations to the predictive model and program developed. The predictive model does have flaws, and while no predictive model can achieve perfect accuracy, there is still room for improvement. Since only one dataset was used in the training of the developed predictive model, overfitting is a risk, and the high accuracy and sensitivity measured may be buoyed by this. Overfitting to the training dataset means that the model will not be as reliable at predicting on new data, as predictions are skewed by biases in the source data. The predictive model also requires medically technical information to make a prediction. While this is necessary for accurate prediction, it does limit the model to professional medical use, since most of the required information is not easily obtainable by the general public, requiring medical testing such as an electrocardiogram. A limitation of the model development process was the lack of a fully integrated feature selection algorithm. Feature selection was done and a combination of wrapper and filter methods were used, however the process of feature selection used did not take into account interactions between features in the final model, which may have limited its ability to select the optimal dataset.

Some limitations of the program's UI include the lack of ability to save inputs and outputs for doctor's records, and a lack of information about the various inputs and outputs limits the interpretability of the output. Allowing a user to save inputs and outputs into the program would be useful for a doctor keeping records on patients, and allow them to keep track of a patient's diagnosis over time, possibly allowing more informed diagnoses to be made over time. There is also no clear instruction on how to use the UI, and while use testing has

shown it to be fairly easy to work out how to use and interpret the program, there is still some ambiguity that could cause problems.

5.3. Improvements and future directions

Future work on this program could improve on the limitations of the current version in many ways. Improving the model by reducing the possibility of overfitting by integrating more data sources into the training data for the model, as well as testing it against a wider variety of data, would be a good way of making the predictions made by the model more robust and applicable to new entries. Further development of the individual models that comprise the final ensemble, as well as further investigation into the development of alternative models, could also improve the accuracy of the model.

To improve the UI, adding the functionality to save inputs and outputs is also an achievable goal that could add to the effectiveness of the program. Additionally, more informative labelling of input fields and output labels could make outputs more understandable by users, increasing the ability of physicians using the program to interpret the output and make judgements on diagnosis.

5.4. Critical Analysis

Overall, development of this project went fairly smoothly. Group rapport and cohesion was good, making the collaboration and decision making a relatively smooth and simple process. The projected workflow that we had established in semester 1 was also executed well, with the program design and development workflow keeping pace with our checkpoints throughout the semester. Despite this, there were numerous challenges, both expected and unexpected problems we encountered and dealt with with varying degrees of success.

One of our major successes in overcoming challenges was the spreading of work between all three team members. To ensure that each member had a more holistic understanding of the project and a more enriched learning experience throughout, we attempted to have each team member have some hand in developing all aspects of the final program. This meant that we couldn't simply assign learning of UI development, model testing, or any other skill that none of us had experience with to a single person to learn. While this did come at a time cost, our approach of initially having a single group member learn any skills we collectively lacked, then using the results of their knowledge to teach the other two members paid off. Numerous times in the development of this program a team member was having problems with a part of the project they had oversight of, and another member was able to provide support, advice, and feedback on fixing the issue or improving the program. This approach ended up improving the quality of our final program, as multiple members were able to comment and contribute to all parts of the development, and no part was left up to a single individual. Another success that was critical to the development of our program was our pivot from using machine learning libraries we were familiar with, to a new library called Caret that offered greater flexibility in model development, but that none of our group had experience with. This switch occurred a few weeks into the project, and we had already created models using libraries we were more familiar with.After a brief discussion and investigation of the Caret library, we decided that a switch would be beneficial but challenging. Switching to the

Caret library, despite having a bit of a learning curb, did end up being beneficial as the greater flexibility and ease of assessing the models allowed us to develop a more sophisticated model than we had previously been capable of doing.

Not all challenges were handled as well as the ones discussed above however. During model development we did extensive training and testing of models to get an idea for how the individual models and libraries we were using functioned. Because we were still learning all the mechanics of the models and libraries, we constructed our testing code line by line, without developing any large scale functions. This was initially useful in learning how libraries and models worked, however once we had a better grasp on the programming aspect, we maintained the non-automated structure of the model development workflow to allow for easier tweaking and debugging. While this did make it easier for us to debug individual models, it did cost us time when we wanted to develop new models, and caused some problems with differences in model development workflow making the resulting models incompatible with the UI we had developed. Incompatibilities were fixed as they popped up, however a standardised automation of model development with tweaking of models and parameters, but standard forms of input and output would have saved time, effort, and likely resulted in more refined models being produced. Not establishing a standard workflow for model development was one of the major failings of our approach to this project, and we could have definitely improved by doing it. Another failing of our project was our inability to incorporate additional datasets into our model training data. We had intended to do this and even found other candidate datasets to incorporate. However, due to differences in data formats and insufficient time, we were unable to properly incorporate this data into our final model. This likely impacted the generalisability of the model and resulted in some overfitting to the single dataset that was used. Incorporating additional datasets had the potential to benefit the developed model and would have been a higher priority if the project was repeated.

6. Conclusion

In conclusion, this project has developed a program that allows prediction of a patient's probability to suffer from CHD based on the input of patient data. This prediction is generated by an ensemble of a Random Forest, an Artificial Neural Net, and a Support Vector Machine predictive model developed using machine learning techniques to create an accurate and sensitive model for the prediction of CHD. This program can be used by qualified physicians as a preliminary diagnostic tool for CHD in order to refer patients to further testing earlier in the onset of the disease with more confidence, improving their outcomes.

7. References

Alizadehsani, R., Habibi, J., Hosseini, M. J., Mashayekhi, H., Boghrati, R., Ghandeharioun, A., . . . Sani, Z. A. (2013). A data mining approach for diagnosis of coronary artery

disease. *Computer Methods and Programs in Biomedicine, 111*(1), 52-61. doi:https://doi.org/10.1016/j.cmpb.2013.03.004

Ciecholewski, M. (2013). Ischemic heart disease detection using selected machine learning methods. *International*

Journal of Computer Mathematics, 90(8), 1734-1759. doi:10.1080/00207160.2012.742189

Cutler, A., Cutler, D. R., & Stevens, J. R. (2012). Random Forests. In C. Zhang & Y. Ma (Eds.), *Ensemble Machine Learning: Methods and Applications* (pp. 157-175). Boston, MA: Springer US.

Dwivedi, A. K. (2018). Performance evaluation of different machine learning techniques for prediction of heart disease. *Neural Computing and Applications, 29*(10), 685-693. doi:10.1007/s00521-016-2604-1

Gokulnath, C. B., & Shantharajah, S. P. (2019). An optimized feature selection based on genetic approach and support vector machine for heart disease. *Cluster Computing, 22*(6), 14777-14787.doi:10.1007/s10586-018-2416-4

Jabbar, M. A., Deekshatulu, B. L., & Chandra, P. (2016, 2016//). *Prediction of Heart Disease Using Random Forest and Feature Subset Selection.* Paper presented at the Innovations in Bio-Inspired Computing and Applications, Cham.

Jasuja, A. (2020). Feature Selection Using Diploid Genetic Algorithm. *Annals of Data Science, 7*(1), 33-43. doi:10.1007/s40745-019-00232-5

Jiang, L., Zhang, L., Yu, L., & Wang, D. (2019). Class-specific attribute weighted naive Bayes. *Pattern Recognition, 88*, 321-330. doi:https://doi.org/10.1016/j.patcog.2018.11.032

Kim, J., & Kang, S. (2017). Neural Network-Based Coronary Heart Disease Risk Prediction Using Feature Correlation Analysis. *Journal Of Healthcare Engineering*, *2017*, 1-13. doi: 10.1155/2017/2780501Kramer, L., Schlößler, K., Träger, S., & Donner-Banzhoff, N. (2012). Qualitative evaluation of a local coronary heart disease treatment pathway: practical implications and theoretical framework. *BMC Family Practice, 13*(1), 36. doi:10.1186/1471-2296-13-36

Krishnaiah, V., Narsimha, G., & Chandra, N. S. (2016). Heart disease prediction system using data mining techniques and intelligent fuzzy approach: a review. *International Journal of Computer Applications, 136*(2), 43-51.

Labani, M., Moradi, P., Ahmadizar, F., & Jalili, M. (2018). A novel multivariate filter method for feature selection in text classification problems. *Engineering Applications of Artificial Intelligence, 70*, 25-37. doi:https://doi.org/10.1016/j.engappai.2017.12.014

Mafarja, M., & Mirjalili, S. (2018). Whale optimization approaches for wrapper feature selection. *Applied Soft Computing, 62*, 441-453. doi:https://doi.org/10.1016/j.asoc.2017.11.006

Noble, W. S. (2006). What is a support vector machine? *Nature Biotechnology, 24*(12), 1565-1567. doi:10.1038/nbt1206-1565

Reddy, G. T., Reddy, M. P. K., Lakshmanna, K., Rajput, D. S., Kaluri, R., & Srivastava, G. (2020). Hybrid genetic algorithm and a fuzzy logic classifier for heart disease diagnosis. *Evolutionary Intelligence, 13*(2), 185-196. doi:10.1007/s12065-019-00327-1

Saeys, Y., Inza, I., & Larrañaga, P. (2007). A review of feature selection techniques in bioinformatics. *Bioinformatics, 23*(19), 2507-2517. doi:10.1093/bioinformatics/btm344

Wieland, R., Kerkow, A., Früh, L., Kampen, H., & Walther, D. (2017). Automated feature

selection for a machine learning approach toward modeling a mosquito distribution. *Ecological Modelling*, *352*, 108-112. doi: 10.1016/j.ecolmodel.2017.02.029

8. Appendix

Appendix 1. Risk Register

Appendix 1: Risk Register

Heart Disease Angina Predictive Model Risk Register

Prepared by Joshua Fehring, Utkarsh Patel, Luke Wilson **Date:** 9/6/2020

No.	Rank	Risk	Description	Category	Root Cause	Triggers	Potential Responses	Risk Owner	Probability	Impact	Risk Score	Status
1	1	Lack of communication between team members	If there is a lack of effective communication between members of the team, It may lead to the progress not managed properly and project may fall behind.	Communication	Poor communication management	Not proper coordination within the team on how to make decisions on certain things, no team work	Schedule regular weekly meetings and checkpoints to ensure all group members are aware of any issues or concerns of other members	Project team members	7	8	56	High (open)
2	4	Technical/security Risk in the website	Software risk can occur for uncertain events. It could end up with data leakage as privacy might be the issue.	Software	Executing website programming	Not enough coordination with functional management and underskilled indiviidual for the website development	It's important to start early and learn in advance how to extract output of the model into UI component	Team members as a group	6	7	42	Medium (open)
3	3	Project not complete by deadline	Barely to complete on time	Timeline Risk	Planning	Not enough monitoring group members on how the tasks have been handled resulting they lose track of time	Set regular achievable milestones to ensure that we stay on track for completion	Individual	5	9	45	Medium (open)
4	2	Model selected earlier than expected	Early completion of the model selection process	Positive Risks	Planning	Too much dedication at the planning stage and while developing	Model can be further refined and extended upon	Project team members	6	9	54	Medium (open)
5	6	Losing individual's work	If not managed properly, individual can lose their workstation proving costly for the team since it would end up losing a large amount of progress	Lost work Risk	Processing	Getting carried away with the work and not too worried about the backup	Team members need to make sure they are working under version controlled software or make sure they upload or save their work freuently to avoid this risk	Project team members	4	9	36	Medium (open)

6	7	Risk of demand	The model used in the output may be impressively accurate (with high sensitive value) such that it positively risks of being in demand	Positive Risk	Model Implementation	Model risks of having too much load after it passes tests on testing datasets and become successful further	Model can be further put into practice after appropriate testing procedures.	Project team members	4	7	28	Medium (open)
7	5	Individual skills risk	Having insufficient skills to work with models and UI platforms to end up spending too much time learning and risking deadlines	Resource Risk	Planning	Not having enough knowledge about the specific programming languages, libraries, models, etc.	The team needs to make sure beforehand about tools and techniques each member is familiar with and learn accordingly well in advanve if need be	Project team members	5	8	40	Medium (open)
8	8	Domain Specific Risk	The implemented model not being able to handle new dataset to adapt to the changes in model	Domain	pre-processing	Not enough variety into original dataset and model properties	While implementing models, make sure to exlplore data and model in thoroughly	Project team members	3	8	24	Low (open)
9	9	Hosting models on online organization platforms	Since the model is hosted on organization websites, it could risk of being overused by the users	Positive Risk	Outputting the tested model	Since model is put on the organisation site in the output process, it can become too popular	The UI or website part of the process can be revisited further to make it smooth and easy for users to access it	group of organization and team members	3	6	18	Low (open)
10	10	Model misdiagnoses people suffering from coronary heart disease as healthy	If the model is not sensitive enough to reliably diagnose patients with coronary heart disease, patients who erquire treatment could be overlooked with serious consequences for their health	User Risk	Insufficient refinement and testing of model	Diagnosis of patients who have been cleared by the model as suffering from heart disease	Avoid this risk by implementing rigorous testing of the model before implementing it in order to verify its suitability for use	Project team	5	8	40	Medium (Open)
11	11	Model misdiagnoses healthy people as having coronary heart disease	If the model cannot distinguish well enough between healthy and ill patients, healthy patients may be diagnosed as having heart disease and referred for uneccessary further testing	User Risk	Insufficient refinement and testing of model	Diagnosis of patients who have been diagnosed as having heart disease as healthy in later tests	Avoid this risk by implementing rigorous testing of the model before implementing it in order to verify its accuracy	Project team	3	6	18	Low (Open)