Using Machine Learning to Predict Cardiovascular Disease

Steven Crowther and Thaddeus Bartz

Abstract—The concept approached and tested in this paper was the idea of predicting heart disease using a machine learning model. Our work was based on an earlier project, which had used fewer attributes to achieve an 83 percent accuracy rating on the data set. A preliminary analysis of works conducted before ours found many issues, one being the focus on a relatively small number of attributes. In addition, their quantity of data was lacking, leading to issues when calculating their actual accuracy. As such, we focused on increasing the number of used attributes, as the algorithm would be able to counteract outlying variables and increase the accuracy accordingly. Our work focused on training two models initially used by the program, that being a categorical and binary model. The binary one had more initial success, as predicting the severity of the heart disease at play was far more difficult than determining if a given row was at risk. By visualizing the training data, our ability to interpret where the model would be most effective was used by stopping it before over fitting the training data. However, our work was unsuccessful in increasing the accuracy of the initial program. An evaluation of the confusion matrices showed that both categorical and binary models shared similar success in classification based on the increase in used attributes. Further work should and will be conducted regarding the accuracy and effectiveness of different data sets, as well as the accuracy of multiple models.

Index Terms—overfitting, underfitting, features, neural networks, features

1 Introduction

Cardiovascular disease (CVD) is composed of a group of disorders concerning the heart and circulatory system. These diseases can take many forms, such as effecting the blood vessels in various areas of the body, while others can directly be caused by the blood itself. All of these, however, can lead to some serious medical conditions such as heart attacks and strokes. Common causes of these is a build up of fat on the inner walls of blood vessels. This can lead to serious blood flow issues, where the blood clots in the veins, blocking that flow and killing the surrounding cells. In contrast, strokes are caused by internal bleeding in the brain, or by clots finding their way into the blood vessels in the brain.

The most common causes of cardiovascular disease include an unhealthy diet, tobacco use, as well as numerous other contributing factors. As these risk factors are also related to other diseases, the difficulty in determining if an individual has the disease is inherently large. In addition, while major factors are the main determinant of the disease, other factors like a history of CVD in the family and the economic status of that family contribute. While there are treatments available, the easiest way to reduce these factors is to first identify that there is an issue, and its underlying causes. Following this, lifestyle changes are needed if the individual wants to combat CVD, such as altering their diet, becoming more physically active, and avoiding certain foods and alcohol.

The symptoms of cardiovascular disease can manifest in ways that make it difficult to identify quickly unless the individual is already aware of the condition. In certain cases, these symptoms can manifest briefly, making it unlikely that an effected individual will seek medical treatment. Symptoms of a heart-related attack, while symptoms of a stroke include numbness, difficulty seeing, and loss of balance. While the more serious symptoms would incline the individuals to seek treatment, the underlying risk factors at play here mean even when getting treated for conditions related to CVD, the underlying causes are still at play.

Cardiovascular disease is one of the leading causes of death in the United States, responsible for a death every 36 seconds. CVD effects well over 18 million Americans over age 20. Treatment for this disease is costly, totaling over 350 billion dollars in 2017, and the cost is only going to increase. Therefore, identifying important potential risk factors for this condition and detecting it early are paramount in the survival of a patient. Additionally, diagnosing this condition early and prescribing both medicinal and lifestyle changes can help slow or stop the ongoing disease. As the quantity of data that goes into the medical system is well and above a human's ability to collate and form trends, the next step is applying machine learning to this problem. Machine learning techniques and data selection have become much simpler to implement, and so using these solutions will allow us to accurately predict heart disease using a few important factors. Implementing a machine learning system to accurately diagnose heart disease also opens up the possibilities of initially screening patients for the condition before they see a medical professional. Additionally, by only focusing on relevant factors related to CVD such as BMI and blood pressure, the cost of diagnosis can be lowered.

2 Related works

The first work examined was performed by Bulent Siyah. His initial work was focused on determining and optimizing

S. Crowther and T.Bartz are with the College of Computing and Software Engineering, Kennesaw State University, Marietta, GA, 30060 USA. Emails: scrowthe@students.kennesaw.edu and tbartz@students.kennesaw.edu

a machine learning method by which heart disease could be diagnosed. The data set used for this initial implementation originated from UCI, and in line with their analysis, focused on a specific set of 14 columns to train the algorithm. Additionally, before using the provided data set a number of data cleaning techniques were used to streamline the algorithm training process. Firstly, normalization was applied to the data set to streamline the training process, as the data is instead distributed via a mean of the values and their standard deviation. Secondly, dropout was applied to the learning algorithm to overall increase the weights of relevant features. This in turn leads to a more sensitive program. Thirdly, weight regularization is applied to the program in turn, altering the trained weights to be less polarized, therefore allowing all the weights to be important when creating an end result. These training alterations were applied both to the data set and to the resultant weights. However, these were also complicated by the inclusion of two different models that were tested. One is a categorical model, allowing the algorithm to detect the severity of heart disease given the data set's points. This algorithm was consistently worse than a binary classification model given the same data. The binary model focuses instead on if a given point has or does not have CVD. Here the initial work focuses on improving the accuracy of the model, arriving at around 80 percent accuracy with the categorical model and 83 percent with the binary model. These scores were only achieved after implementing the data and weight altering methods above, as well as limiting the number of testing epochs the algorithm was trained for.

While this training method was successful in creating an 83 percent accuracy rate, the problem is less focused on the methods, but rather the focused columns that Mr. Siyah used. The data set he used is a trimmed down version that UCI also studied for predicting heart disease severity. While using the same columns, several data points that actually effect Heart disease severity were not included, such as Body Mass Index, or BMI. The exclusion of these points represents an issue with the predictive model, that being excluding information related to the severity of the disease. Removing/ignoring these values means the prediction is not an actual representation of the factors at play when talking about CVD. Therefore, the prediction can miss important causes of CVD, therefore effecting the individuals using this method of detection.

The second work examined was created by Nisha Choondassery. This work examines an ongoing data set sourced from a study still being performed in the town of Framingham, Massachusetts. The data consists of over four thousand records and 15 recorded attributes related to coronary heart disease. Following the initial import, this data was subjected to some prepossessing techniques. The objective with these was to remove data that did not heavily correlate with the outcome, therefore tuning the algorithm to be more sensitive regarding the attributes still included in the data set. The first data cleaning technique was to remove null values from the data, which constituted around twelve percent of the total collected data. Next, the data was processed to determine which attributes had the highest correlation values relating to coronary heart disease. This process cut eight attributes out of the data entirely due to their low Pvalues. The Pvalues were obtained via an initial run on the data set using a probability matrix, then using that matrix to remove non-statistically significant factors from the data. Following these cleaning techniques, the data was then split and fed into a logistic regression model, achieving a score of 88 percent accuracy. Using this model as the initial run, a confusion matrix was then used to calculate the number of type one and two errors. Calculating these revealed the regression to favor false negatives, and considering the subject matter, false negatives are very dangerous when detecting treatable conditions. The final step they took was to lower the sensitivity of the program, and while this did drop the accuracy to 73 percent, the drop in accuracy was weighed against the possibility of false negatives.

This program addressed some of the issues related to long term CVD, and how the risk factors effect the likely hood of the condition occurring and progressing. However, the issues presented with this data are similar in scope to the first analyzed method. By removing attributes with low correlation, the objective was to increase the predictions accuracy. But in this instance, the programmer managed to catch the issues associated with their tuning of the data. In this case, the false positive rate represents people who have the condition, yet the algorithm has determined otherwise. They mention this later, and the idea to increase the program's sensitivity with the data available was done. However, we would have suggested a different approach, that being using some of the other excluded columns of data for the model. In this manner, while they might have a low correlation to the subject manner at hand, its likely that the excluded data would allow for a more accurate model to be created.

The third work examined was a visualization and prediction of heart failure created by Sanchita Karmakar. The work begins with a highly visual analysis of the data set, including multiple types of graphs and charts. One type that was favored was the plot of the attributes correlation to the death event. Selecting 3 features based on their correlation values here, these values are split into the according testing and training values, whereupon the data was fed into multiple different models to determine the most effective classifier. The process of feeding the data into the variety of models here resulted in an average accuracy of 91 percent, with the only method scoring below a 90 being Logistic regression.

While this approach reached a very high accuracy score, there are several issues with the base data fed into the system. Firstly, the data used was quite small, hence the high accuracy scores. Secondly, the provided confusion matrices for these approaches show very little in the way of improvement concerning the false negative rate, hovering at around 13-16 for this data set. Another note to be made here, but the data actually used by the program doesn't include several statistically significant factors logged as attributes initially, such as platelet count and age of the individual. Not including this data decreases the likely hood of a correct outcome being predicted by the algorithm. Additionally, even if the elements in question may cause an overall loss in accuracy, the decrease in false negative should be weighed against overall prediction accuracy. One last thing that would be welcome is a much larger data set, as the number of rows totals 299. More data would likely improve

the algorithms accuracy if used in a larger scale.

The fourth work explored was authored by Nelakurthi Sudheer, and focused on predicting heart disease. The initial data set mentioned consisted of a wide range of attributes, totaling 76, but the focus of the program was on a much smaller set of 13. After running a quick heat map to ascertain which features scale significantly with the output column, the statistically significant values made up five of the total attributes. However, this information was not used when feeding the data into the used algorithms and models. The models accuracy was around a 72 percent accuracy, with the best model being the Naive Bayes Classifier scoring an 80 percent.

The earlier work performed to determine which attributes fit best to the problem was not used later on in the project. Additionally, no provided confusion matrix makes the acquired results much less truthful considering the earlier works discussed in this section. Finally, the data set used only contained around 300 entries, making it much more likely to have a higher accuracy considering its small size. If reworking this problem, the focus should be on using the information that was acquired earlier on in the program to tune out some of the non-correlating variables. Finally, using the confusion matrix for the program should have been provided, seeing as were this to be used in the real world, false negatives would likely prove costly, if not fatal. Tuning the algorithms with these points would have likely decreased the accuracy, but the overall false negative rate should be lower as a result.

The last work investigated in this section was authored by Nayan Sakhiya, focusing on predicting heart failure and modeling the potential data. The idea behind the visualisation of the data is that trends can be easily seen with the eye, and subsequently used to filter and trim data that is fed into a given algorithm. There is also a heatmap used to assist in the finding of significant factors relating to the output event. By using the high correlation figures here, they avoid tuning the algorithm just using the model score as its evaluation. From there, the data is inserted into several models, achieving at the highest point a 93 percent accuracy rate.

When evaluating the data, an initial inquiry should be made regarding the data's integrity, as well as the spread of its attributes and values. Poor data can either lead to models with extremely high accuracy, or models with very poor output results. In this case, Nayan Sakhiya trimmed down his model to focus on just 3 features. In that case, the algorithms high accuracy rate is also down to the relatively small size of the available data set. The accuracy achieved here should also be tempered by the fact that when dealing with medical data, mistakes or tuning methods could quite easily overrun the idea of wide diagnosis.

In conclusion, the works studied here make use of a variety of machine learning tactics, including data prepossessing and feature selection. However, when using these tactics on medical data, its important to remember that accuracy of the program should not be the first priority. The idea of these, even if not implemented in the system, is to diagnose conditions using preexisting data. Thusly, it becomes doubly important to understand what techniques are used when creating a prediction algorithm, and even more so, to know where to draw the line for accuracy vs correctness.

3 Proposed method

As we have discussed in the related works section, there have been many attempts at predicting heart disease. After reading and analyzing previous implementations, we decided to implement and test our own method for predicting heart disease. Furthermore, we decided to implement our heart disease prediction program with a drastically different approach than most other methods, while expanding on one specific area. Most previously implemented methods, mainly centered around the same approach. This approach was to create some sort of prediction program that requires the use of 14 specific features. These 14 features were chosen from a set of 76 total features. The use of these features are mainly due to a paper written in 1989 and researching the relationship between each feature and heart disease. While the paper and research shows everyone the value of these features, the medical world's understanding of heart disease has shown us that we need to consider more factors for predicting heart disease. Therefore, we decided to modify an existing heart disease prediction program, to utilize more features.

3.1 Baseline Method

As we previously stated, there have been numerous attempts at creating a program that can predict heart disease. However, none of them have reliably predicted heart disease. Many of the results seem to range anywhere from 80 to 88 percent. A notable case of this is Kaggle. Looking and comparing the results of the highest voted implementations on Kaggle, we found similar implementations of an algorithm, but the implementations seem to have different results. As a result of finding similar ideas with conflicting results, the only thing we were able to conclude was that there was no true baseline to work with. Despite this reality, we decided to take a machine learning approach for predicting heart disease. We decided to modify an existing person's program.

This person, who goes by Bulent Siyah, decided to tackle the task of predicting heart disease by exploring the use of neural networks. Even though there is a good amount of uncertainty, Siyah's implementation seems to be more reliable, so we decided to use Siyah's best model accuracy of 83 percent as a baseline. We chose Siyah's program, because it did not seem to have any problems, as seen with others. Siyah's implementation consisted of creating and training two separate classification based neural network models and the use of the 14 features. Siyah implements a categorical classification based neural network. Siyah then trains and tests the neural network using the data from the csv file. Afterwards, Siyah implements a binary classification based neural network. Once again, Siyah trains and tests the neural network using a modified copy of the previously extracted data. According to Siyah's code, Siyah chose to implement both neural networks with three layers. These layers were implemented with an input layer, a one hidden layer, and the output layer. Siyah decided to use a rectified

linear unit for both hidden layers. For the input layers, Siyah used a 13 dimension input, normal kernel initializer, L2 kernel regularization, and rectified hidden layers, for both hidden layers. Siyah's output layer, for the categorical model, uses a softmax for the activation function, and a sigmoid activation function, for the binary model. Lastly Siayh uses Adam for optimizing both neural networks.

3.2 Proposed Model

For our method, we decided to use the same models used by Siyah. As we stated in our Baseline Method, Siyah decided to implement a categorical and binary model. We decided to use these models due to the nature of each model and their use in the field of medicine. A categorical model takes a sample set of data and will decide where to place a sample set, given a set of classes. In our project, we need to decide if a sample set of data should be placed into the class of heart disease or not heart disease. Hence, it is worth exploring the effectiveness of this model. A binary model will take a sample set and decide if the value is a one or zero. Since we are dealing with heart disease or no heart disease, the problem can easily be broken down to a binary value. As we can see, either one could be well suited for our given task. The reason why we decided to use both was to see which one would yield a better result. Of course, there are other neural network models we could have tried.

Some of the most popular ones are a MLP, CNN and RNN. A multi-layer perceptron, or MLP, is a network network with one than one hidden layer. A convolutional neural network, or CNN, is a neural network that pools information from a sample set, typically an image, and evaluates the pool with fully connected layers, as explained in viso. A recurrent neural network, or RNN, is designed to deal with the sequential feeding of data. We didn't use a MLP, because we wanted to take the simpler approach of just increasing the amount of parameters we would use and not deal with several hidden layers. While viso has shown CNNs have a history of having high accuracy, a research paper by Aizubaidi et al, has pointed out the fact that CNNs need a lot of data to properly function, however the ideal amount of data, for our task, does not exist. We will explain this in more detail within our Dataset section. Since we are not dealing with sequential data, it would make no reasonable sense to use an RNN. To reinforce our earlier statement, we wanted to use a categorical and binary model to their nature and functional use to our case. In essence, it boils down to the principle of Occam's Razor. For those unaware, this notion states that the simple solution is often the best solution. In our case, reason would suggest that the most relatable models would be the simple solution, therefore the best solution.

3.3 Implementation Details

For our implementation for heart disease prediction, we utilized two datasets and a collection of libraries for data preprocessing, creating neural networks, offering visual aids for representing our chosen features, as well as training the networks. The beginning of the programs shows all of the features being used as parameters for both neural networks, which are extracted from one of the datasets. Afterwards we show a subset of 20 rows to offer an idea of what data we have to work with. Due to the chance some of the data is likely to have missing information we have to remove certain rows contains columns that have either incorrect or no information. The main reason for doing this data removal in the notebook to make sure we don't include non-essential data. We offer a visual presentation of the most common values for each feature with histogram. Showing the most common values could be useful in modifying our program to better pick on patterns. The bar graph of frequencies for ages that yield have or do not have heart disease offer the same purpose as the histogram. The heatmap offers a 2d representation of the severity level for how each features affect each other. Afterwards we have to remove the target column, since this is used as reference point for determining the accuracy for each model. Next we use sklearn for generate both our training and testing set. Next we initialize and run both our models. We decided to have the categorical model's hidden layer use a rectified linear unit for the activation function and l2 regularization, and uses softmax for the final layer. A softmax is best for multiclass classification, which should work best for a categorical model. Next we feed the dataset into the categorical model. Afterwards we used matplotlib to show a graph of the model loss and accuracy over time. The binary model required the data being pre-converted into training and testing set of ones and zeros. From here, we initialized the model's hidden layer with a rectified linear unit as an activation function and l2 for regularization. The binary model's last layer used the sigmoid function, as is better at handling values from 0 to 1. Next we once again do a graph showing model loss and accuracy, but with the binary function. Lastly, we use sklearn to measure and show the accuracy and precision of both models.

4 Experimental Setup

As we have mentioned, we had hypothesized that 14 features are not enough to accurately predict heart disease. Our other goal was to answer a series of questions about our experiment. In order to test our hypothesis, we needed to create a dataset that would allow us to test more features on our neural networks. This led us to search the internet for more data. After finding more data, we then took on the task of cleaning the data. Finally, to measure the reliability of our data and our model we used a collection of common libraries.

4.1 Research Questions

For our project we want to answer several questions:

- What other parameters do we need?
- Would too many features cause our neural networks to experience overfitting?
- Should we consider inferring some of the data?
- If our neural networks do not produce satisfactory results, what might be causing this?

Since we believe that more features would be needed to improve the chosen baseline method's accuracy the first question was meant to help decide what the necessary features are. The next two questions is are common problems

when implementing any kind of neural network. The last question stem from the problem when we were searching for more data.

4.2 Dataset and Preprocessing Techniques

Not only did we need more data, we also needed data that contained our desired features. This led us to search for the origin of the source of the 14 features. Siyah's implementation was part of a larger Kaggle community called Heart Disease UCI. After visiting UCI's heart disease database, we discovered a total of four dataset files of heart disease samples. One dataset, called Cleveland, appeared to be the original source for the Heart Disease UCI community. As happy as we were, we did need to clean each dataset to make them usable. This relied on us using regular expressions and the python library pandas to remove any rows containing corrupt data, as well as convert the files into csv files. After we had cleaned and created our csv file, we discovered there were less than 303 samples. This was a problem. To train our neural network, we needed to have at least 1000 samples. Ideally, we would need to have 10,000-100,000 or more instances of our features. We looked at each dataset to find out why there were so few samples. Each of the datasets contained some negative values in some of the columns, for some of the rows. There was no indication of why this was the case. Despite this obstacle, we proceeded to use whatever useful data we could find, which was about 202 samples. We did consider inferring missing data, using well-known inference tool called Epsilon Machine Inference. However, due the fact that many rows were mostly corrupted data, we decided to stick with the non-corrupted data. Besides the challenge of cleaning and finding useful data, the next problem faced was choosing the correct features for training and test our neural networks.

In order to figure out which parameters we needed, we spent a great deal of time researching the effects of many of the parameters on heart disease. This search led to a second problem. Among the list of 76 features, a good portion of them used abbreviated names that were too ambiguous. Luckily, we were able to find more features that were clearly overlooked. We then encountered another problem regarding the features we could understand. Among the features we understood, two of them, being hypertension and digitalis, had information that implied they might not need to be features of heart disease. The use of unrelated features could have led to both neural networks overfitting. One study of hypertension, done by G T McInnes, showed that hypertension was not a direct cause of coronary heart disease. An article on digitalis explained that digitalis would not treat all types of heart conditions. The reason why we were hesitate on including these features was due the possibility of overfitting both neural networks. Overfitting occurs when neural network trains and test with too many parameters, which might have led to lower accuracy. To test whether these features would cause overfitting, we decided to create two separate datasets, as we mentioned in the implementation details. We then tested each dataset on our neural networks.

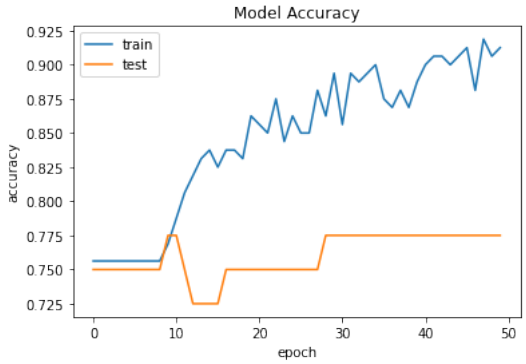

Fig. 1: Caption

4.3 Evaluation Metrics

For our implemented models and features, we decided to use a collection of metrics to measure their performance and reliability. These metrics were accuracy, loss, precision, data shape, and a heatmap. The use of accuracy, loss, precision, were a way to gauge the performance and outcome of each neural network, using the python libraries sklearn and matplotlib. We used the data shape and heatmap to give us an idea of how our chosen features would perform, using the python libraries matplotlib and seaborn. We gauged each neural network's accuracy by measuring accuracy over many epochs. An epoch represents a batch or specific size of data samples. In order to properly demonstrate the accuracy of our neural networks, we needed to understand how accurately they would predict heart disease by training and testing them on a number of batch sizes. Besides using epochs as an accuracy metric, we also wanted to see the highest level of accuracy that each neural network would be able to produce. We would then measure the loss over many epochs to better express any problems or difficulties the neural networks were facing. Our last metric for the neural network would be the use of percent of precision to show how likely the neural networks would be at reproducing their highest level of accuracy. Regarding the data shape, we used a series of histograms to help us understand how the data distribution for all the samples would look. Lastly, the heatmap helps to show how the value of one feature will change when its neighbor feature's values are also changed.

5 Results and Discussion

5.1 Model Evaluation

As we have previously discussed we decided to create and test two separate dataset on our neural networks. The first dataset consists of 20 features, while the second dataset consists of 22 features. In order to determine which dataset would be more reliable, we ran our code on both sets to gauge each dataset's neural network accuracy, precision, and loss. Initially, we had supposed that including more features would most likely cause both neural networks to lose accuracy. After we ran the code using the dataset containing 20 features, we were able to see the categorical model's accuracy per epochs, as seen in Figure 1.

From what we could see in the Figure 1 graph, the categorical model's training accuracy seemed to increase, as

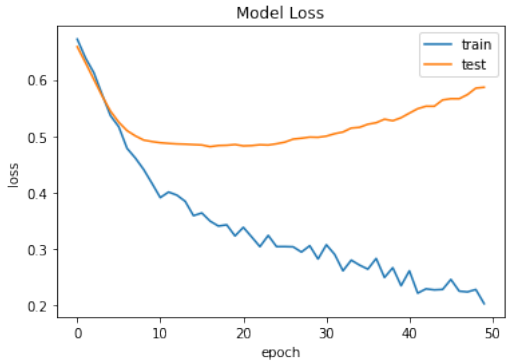

Fig. 2: Caption

Results for Categorical Model with 20 features
0.775

	precision	recall	f1-score	support
0	0.82	0.90	0.86	30
1	0.57	0.40	0.47	10
accuracy			0.78	40
macro avg	0.69	0.65	0.66	40
weighted avg	0.76	0.78	0.76	40

Fig. 3

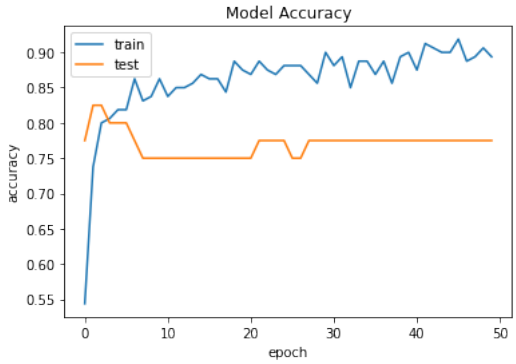

Fig. 4

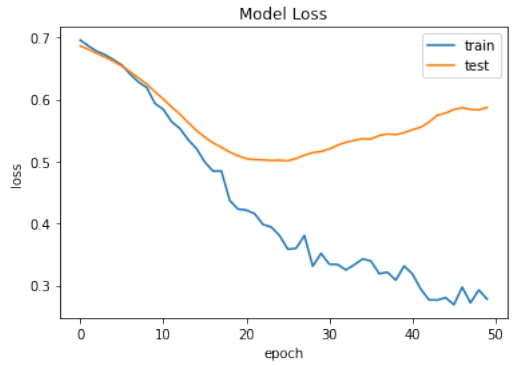

Fig. 5

the number of epochs increased, despite the increase being sporadic. The graph showed us that the categorical model's testing accuracy seemed to experience mainly steady accuracy, with a plateaued change in the accuracy. As we can see from the model's training and testing accuracy, there was clearly an immense difference between the training and testing accuracy. From this graph's results and the size of the training set vs the testing set, there seemed to be a fairly large amount of overfitting. The result was likely due to the small dataset size and selection of data. The next graph we saw was the loss per epoch.

Looking at Figure 2, we saw that the categorical model's training loss was decreasing as the epochs increased, even though the decrease was somewhat sporadic. We noticed this was far less sporadic than the training accuracy. As for the testing set, we saw that there was an increase in the loss as the epochs increased. This increase was fairly steady, as opposed to the training set. After contrasting both the training and testing set, we saw that there was a significant difference, similar to the accuracy per epochs. Similar to the accuracy, we determined that the immense difference between the training and test loss is due the small dataset being used, as well as the testing set consisting of 20 percent of the whole dataset. The results were not too surprising. The more interesting part was the gradual increase in the testing set's loss overtime. As we mentioned before, the model's training accuracy was increasing but in an irregular manner. This irregular manner slowly increased per epoch. Therefore, we deduced the testing loss's gradual increase was due to the training accuracy's behavior. Now in figure 3, we saw a general overview of the categorical model.

Here, in figure 3, we saw the model's highest accuracy, as well as the precision. The highest degree of accuracy that the categorical model could achieve was 78 percent. As for the precision, we noted that the categorical model only achieved a precision of 82 percent. This is of course lower than the baseline accuracy of 83 percent. From here, we moved to the binary model's accuracy in figure 4.

From the accuracy, of figure 4, we noticed that the binary model's training accuracy took an enormous leap up, then sporadically increased with the epochs. It was not as sporadic as the categorical model's accuracy. We noticed that the model's testing accuracy had a similar behavior, but with more plateaued degrees of accuracy. Contrasting them showed us there was less of a difference between the training and testing accuracy, as opposed to the categorical model. This seemed to suggest that the binary model would be a more reliable model, then the categorical model. As we saw with the categorical model the plateauing nature of the testing accuracy was likely due to the small sample size. From here we observed the model loss, in figure 5.

As we observed the model loss, in figure 5, we saw that training loss was decreasing as epochs increased, with minor fluctuation. From the testing loss, we saw that the model was indeed decreasing. The training loss did not show us any serious surprises. However, we noticed that around halfway through there was a shift from decreasing loss to increasing loss. As surprising as this seemed, this was similarly seen with the categorical model's loss. Despite the similarity, there was still a difference in the rate of loss. The difference between the two models testing loss seemed to reinforce the previous idea that the binary model is better than categorical model.

Using figure 6, we had hoped to get a better idea of the general performance of the binary model. The overview

Results for Binary Model with 20 features
0.775

	precision	recall	f1-score	support
0	0.82	0.90	0.86	30
1	0.57	0.40	0.47	10
accuracy			0.78	40
macro avg	0.69	0.65	0.66	40
weighted avg	0.76	0.78	0.76	40

Fig. 6

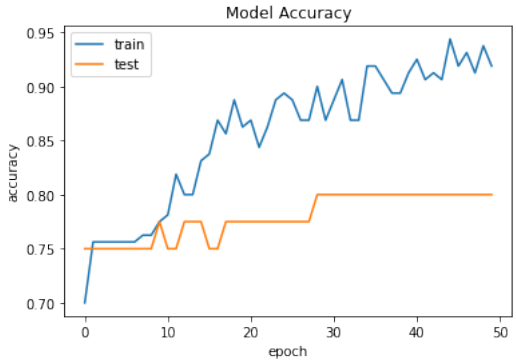

Fig. 7

shows us the same topic as with the categorical model, being highest accuracy and precision. The binary model appeared to have a highest accuracy of 78, and a precision of 82. The accuracy seemed to differ a lot from the binary model accuracy's highest accuracy of about 83.

After we had ran and observed all aspects of both models, we came to some conclusions. It noticed that the model appeared to have similar performances. Despite the shared characteristics, the binary model was able to perform slightly better. It seems the binary model was likely the better algorithm for predicting heart disease with 20 features. To be sure though, we then used the 22 feature dataset to test each model.

When we ran the categorical model on the 22 features we got the accuracy shown in figure 7. This graph showed the changes in the categorical model's accuracy. Here we saw the training accuracy had improved, as it increased, but was less sporadic. Furthermore, we saw the testing accuracy had steadily increasing plateaus, with minor decreases. As we had stated before, the difference between training and testing would likely be due to a small dataset. The minor increase in accuracy seemed to suggest a favoring of more features. We then moved to the model's loss, in figure 8.

When we ran the categorical model on the 22 features we got the accuracy shown in figure 7. This graph showed the changes in the categorical model's accuracy. Here we saw the training accuracy had improved, as it increased, but was less sporadic. Furthermore, we saw the testing accuracy had steadily increasing plateaus, with minor decreases. As we had stated before, the difference between training and testing would likely be due to a small dataset. The minor increase in accuracy seemed to suggest a favoring of more features. We then moved to the model's loss, in figure 8.

In figure 8, we saw that the training loss was steadily decreasing per epochs. The testing loss showed us a decrease in loss then a gradual increase in loss. Both the training and testing loss graphs were nearly identical to the loss graph from before. This differed from what was expected. The model accuracy shows a change, despite the model loss showing no significant change. We had imagined at least some degree of change from the loss, but seemed to find no real change. This indicated that the change in accuracy was only a minor change. Next we studied the overall behavior of the categorical network, with figure 9.

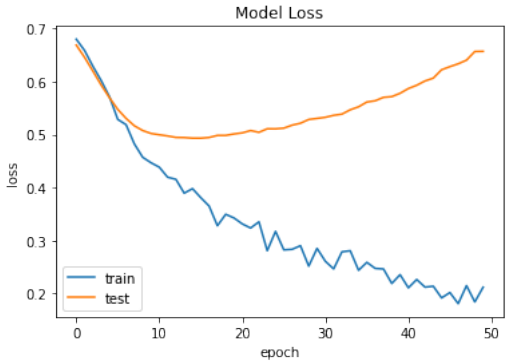

Fig. 8

Results for Categorical Model with 22 features
0.8

	precision	recall	f1-score	support
0	0.82	0.93	0.87	30
1	0.67	0.40	0.50	10
accuracy			0.80	40
macro avg	0.75	0.67	0.69	40
weighted avg	0.78	0.80	0.78	40

Fig. 9

Figure 9 gave us a better idea on the performance of the model. Here, it seems, the model's accuracy did indeed improve. The accuracy had an increase in accuracy from 78 to 80. This indicated the accuracy was having a minor improvement as a result of the extra features. After looking at the progress of the categorical model, we moved to the binary model, with figure 10.

The binary model's training and testing accuracy, in figure 10, showed us a minor improvement from before.

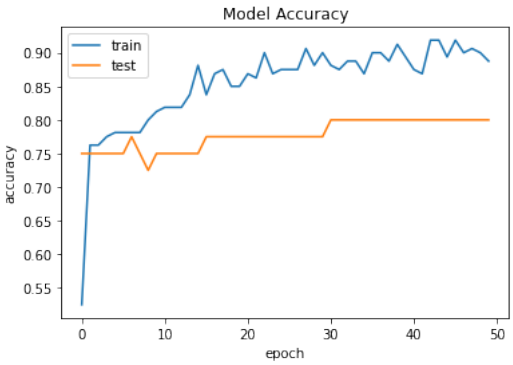

Fig. 10

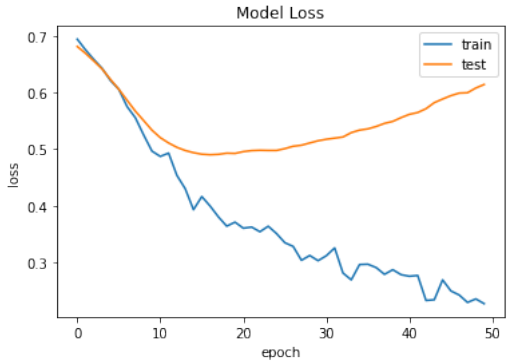

Fig. 11

Results for Binary Model with 22 features
0.8

	precision	recall	f1-score	support
0	0.82	0.93	0.87	30
1	0.67	0.40	0.50	10
accuracy			0.80	40
macro avg	0.75	0.67	0.69	40
weighted avg	0.78	0.80	0.78	40

Fig. 12

The training set didn't seem to have any substantial improvement from before. However, the testing model did show improvements. This time the model appeared to be able to steadily improve its accuracy, despite being a minor improvement. While this did not show a huge improvement compared to the other model, it represents a potential increase due the use of more features. Next, we took a look at the model's loss.

Here, in figure 11, we saw the binary model's loss with respect to the training and testing sets. The training did not seem to show any clear signs of a change in the loss. The testing set, however, appeared to have had a slight increase in the loss. Altogether, there seemed to only be a slight change in the loss. This was somewhat surprising, but mainly expected as there was not a significant change in the accuracy. From this close up view of performance, we decided to see the overview of the binary model, as we had before.

After viewing the binary model's accuracy and precision, using figure 12, we noticed two things. We first noticed the accuracy had indeed improved. We then noticed that the result was nearly the same as the categorical model. The accuracy had seemed to increase from 78 to 80.

Despite increasing the amount of features our models had trained with, we did not see a significant change in their performance. However, this did not mean there was no change. We had discussed that there was an improvement in each model's accuracy, after increasing the available features. This suggested that our idea for increasing the features to improve accuracy is not invalid. At this point the only way to see a significant change would be more data samples, or even deep learning.

6 Conclusion

As we have discussed in our introduction, Cardiovascular Disease is one of the leading causes of death in the United States. This has lead to a lot of time, effort, and money being devoted to identifying the risk factors, so that we are able to predict and stop heart disease. Unfortunately there has yet to be an accurate and precise method for predicting heart disease. For this reason our focus was on determining a computer-based system for predicting heart disease based on parameters. As such, the initial approach of enlarging the amount of data fed into the system resulted in a significant drop in prediction accuracy. However, this drop was accompanied by a curious development, that being that both categorical and binary models had comparable accuracy with the twenty-element data set.

In terms of future endeavors, an effort should be made to expand the number of utilized algorithms to determine which has the highest rate of success. In addition, while the data set we used was quite large, more would likely be necessary for adoption into the healthcare system. As such, gathering more data to feed into the models, as well as increasing the number of models would result in higher accuracy rates. One suggested method of investigation would be applying the data to a more comprehensive model, or using an ensemble method of training with the same data.

References

https://www.ijert.org/breast-cancer-detection-using-machine-learning-techniques
https://www.nature.com/articles/s41598-020-73060-w https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds)
https://www.heart.org/en/health-topics/consumer-healthcare/what-is-cardiovascular-disease
https://www.kaggle.com/bulentsiyah/heart-disease-prediction-using-neural-networks
https://www.kaggle.com/neisha/heart-disease-prediction-using-logistic-regression
https://www.kaggle.com/sanchitakarmakar/heart-failure-prediction-visualization
https://www.kaggle.com/nelakurthisudheer/eda-preciction-of-heart-disease https://www.kaggle.com/nayansakhiya/he
fail-analysis-and-quick-prediction
https://www.baeldung.com/cs/epoch-neural-networks
https://pubmed.ncbi.nlm.nih.gov/8576788/ (hypertension) https://www.fda.gov/tobacco-products/health-effects-tobacco-use/how-smoking-affects-heart-health
https://pubmed.ncbi.nlm.nih.gov/3886752/ (dig)
https://medlineplus.gov/ency/article/000165.htm (problems with dig) https://pubmed.ncbi.nlm.nih.gov/31402328/ (prop) https://pubmed.ncbi.nlm.nih.gov/7848018/ (nitrates) https://www.mayoclinic.org/diseases-conditions/high-blood-pressure/in-depth/calcium-channel-blockers/art-20047605 (pro)
https://medlineplus.gov/ency/patientinstructions/000112.htm (diuretic) https://towardsdatascience.com/l1-and-l2-regularization-methods-ce25e7fc831c (L2 regularization)
https://www.researchgate.net/publication/350527503$_{R}eview_{o}f_{d}eep_{l}e$
$//www.mayoclinic.org/diseases-conditions/heart-$

$disease/symptoms - causes/syc - 20353118https : //viso.ai/deep - learning/deep - neural - network - three - popular - types/https : //www.kaggle.com/ronitf/heart - disease - uci/code?datasetId = 33180sortBy = voteCounthttps : //www.kaggle.com/ronitf/heart - disease - uci$