CS1390: Project - Predicting Heart Failure using supervised Machine Learning Methods

Avani Maroo and Veda D

- Introduction :

Our project aims to perform supervised learning methods to predict heart failure and allow us to pick out which features largely can indicate heart failure. We also wanted to develop an understanding of what sort of dataset would be required for such forms of supervised machine learning methods, providing us with a better insight into further projects we do.

Predicting Heart Failure is extremely vital with it being one of the most serious killers across the world. As mentioned in Researcher David Chicco's paper cardiovascular diseases kill approximately 17 million people globally every year, and they mainly exhibit myocardial infarctions and heart failures. To begin we must define what heart failure is: Heart Failure is when the heart fails to pump enough blood to meet the needs of the human body.

However, predicting heart failure is extremely hard as there are many parameters to monitor and it also largely depends on parameters such as the stage of heart failure, the presence of diabetes, high blood pressure etc. Trying to understand which would be ideal markers is key. However, one of the main problems that occur with these datasets is a lack of balance in our patient's records. More often than not, there is a higher number of patients who survive compared to those who do not. Most research papers overcome this by performing a Stratified K-Fold to create a more balanced training and test set. We shall elaborate more on this further.

- Literary Survey:

The initial paper that used the following dataset performed by researchers Asif Newaz, Nadim Ahmed and Farhan Shahriyar Haq performed a 5 Fold Stratified Split on the dataset and trained it via Balanced Random Forest and further performed a Chi-Square Test and Recursive Feature Selection to identify features that were highly related to heart failure. We would like to elaborate more on the above-used methods when we discuss our methods and implementations. The results (Fig 2.1) were further compared with SVM, Logistic Regression, ADA boosting, KNN Classification and Random Forest Classifier.

	SVM	**KNN**	**LR**	**AdaBoost**	**RF**
Sensitivity	56.42	33.47	65.63	58.42	59.53
Specificity	69.5	73.46	71.98	73.93	79.34
G-mean	61.98	48.24	68.61	65.32	68.41
Accuracy	65.21	60.54	69.92	68.91	72.93
MCC	25	7.25	35.97	32.01	38.68
ROC-AUC	62.96	53.47	68.8	66.17	69.43

Fig 2.1

The results (Fig 2.2) for the Balanced Random Forest classifier were as the following:

	Balanced Random Forest (BRF) Classifier
Sensitivity	71.9
Specificity	73.45
G-mean	72.67
Accuracy	72.93
MCC	43.39
ROC-AUC	72.67

Fig 2.2

The second paper which further worked on the above dataset was by researcher David Chicco who tried to conclusively prove that machine learning can be used to predict heart failure by monitoring Serum Creatinine (a waste product present in the blood that is generated by the muscles) and Ejection Fraction (measures the amount of blood pumped out one's ventricles) alone.

The paper changed some of the features of the original dataset and perform 10 different machine learning methods: three Tree-based methods: Random Forest, One Rule and decision tree, a linear regression model, one artificial neural network using perceptron, two Support Vector Machines using linear and Gaussian RBF, one instance-based classifier(KNN Classification), one probabilistic classifier method (Naive Bayes) and an ensemble boosting method (Gradient Boosting). They further performed a feature ranking using traditional univariate biostatistics techniques and compared results before and after. The biostatistics techniques employed were Chi-Square Test, Mann-Whitney U Test and Pearson Correlation Coefficient. This allowed them to highlight the two features that seemed to have the highest relation with heart failure. The following features were selected: Serum Creatinine and Ejection Fraction.

The results (Fig 2.3) for all the machine learning methods on all the features (mean of 100 executions) are as follows:

Method	MCC	F_1 score	Accuracy	TP rate	TN rate	PR AUC	ROC AUC
Random forests	blue**+0.384***	0.547	blue0.740*	0.491	0.864	0.657	blue0.800*
Decision tree	**+0.376**	blue0.554*	0.737	blue0.532*	0.831	0.506	0.681
Gradient boosting	**+0.367**	0.527	0.738	0.477	0.860	0.594	0.754
Linear regression	**+0.332**	0.475	0.730	0.394	0.892	0.495	0.643
One rule	**+0.319**	0.465	0.729	0.383	0.892	0.482	0.637
Artificial neural network	**+0.262**	0.483	0.680	0.428	0.815	blue0.750*	0.559
Naïve bayes	**+0.224**	0.364	0.696	0.279	0.898	0.437	0.589
SVM radial	**+0.159**	0.182	0.690	0.122	0.967	0.587	0.749
SVM linear	**+0.107**	0.115	0.684	0.072	blue0.981*	0.594	0.754
k-nearest neighbors	**-0.025**	0.148	0.624	0.121	0.866	0.323	0.493

Fig 2.3

The results (Fig 2.4) for SVM(radial), Random Forest and Gradient Boosting on Serum Creatinine and Ejection Fraction (mean of 100 executions) are as follows:

Method	MCC	F_1 score	Accuracy	TP rate	TN rate	PR AUC	ROC AUC
Random forests	blue**+0.418***	blue0.754*	blue0.585*	0.541	blue0.855*	0.541	0.698
Gradient boosting	**+0.414**	0.750	blue0.585*	blue0.550*	0.845	blue0.673*	blue0.792*
SVM radial	**+0.348**	0.720	0.543	0.519	0.816	0.494	0.667

Fig 2.4

-

- Project Description:

Our project aims to predict heart failure from the above dataset using various forms of supervised machine learning methods. We have also tried to understand how an imbalanced (medical) dataset must be treated and what methods help in providing a more accurate model, especially in the case of supervised machine learning methods. We also hoped to determine and revalidate the results obtained from the previous papers in proving which factors were highly related to heart failure and would be necessary parameters that might need to be monitored in patients with Heart Failure of stages 3 and 4. We also hoped to show the linear separability once we have identified the two parameters that seemed to be highly related to heart failure.

We initially began struggling to even comprehend what we could perform any form of machine learning on. We both had a keen interest in trying to perform some form of supervised learning to get a better grasp on the understanding of the methods in practicality that professor mentioned in class. We tried complicating our project but after a conversation with the professor, he mentioned that a simple project does not reduce the merit of the project. We decided to pursue a project that had focused on our interests: something in medical health. Combined with our interest in understanding how data sets needed to be treated, this seemed perfect keeping in mind that most medical datasets are extremely imbalanced in each other.

- Dataset Specification:

  The dataset contained 299 medical records with 13 parameters. The data collection was originally performed in Faisalabad, Pakistan. The current dataset we are using is an improved version used by Researcher David Chico and was obtained from the UCI Repository. All the patients had either stage 3 or 4 of heart failure as defined by the New York Heart Association. The first 11 parameters are various features such as age, gender and more. The 12th feature is time, which indicates the date of the follow-up to check the condition of the patient. The 13th parameter is the target feature of the dataset which has been referred to as "DEATH_EVENT". This parameter lets us know if the patient survived in the follow-up period, another parameter.

  Here is the list (Fig 4.1) of parameters present in the dataset :

Feature	Explanation	Measurement	Range
Age	Age of the patient	Years	[40,..., 95]
Anaemia	Decrease of red blood cells or hemoglobin	Boolean	0, 1
High blood pressure	If a patient has hypertension	Boolean	0, 1
Creatinine phosphokinase	Level of the CPK enzyme in the blood	mcg/L	[23,..., 7861]
(CPK)			
Diabetes	If the patient has diabetes	Boolean	0, 1
Ejection fraction	Percentage of blood leaving	Percentage	[14,..., 80]
	the heart at each contraction		
Sex	Woman or man	Binary	0, 1
Platelets	Platelets in the blood	kiloplatelets/mL	[25.01,..., 850.00]
Serum creatinine	Level of creatinine in the blood	mg/dL	[0.50,..., 9.40]
Serum sodium	Level of sodium in the blood	mEq/L	[114,..., 148]
Smoking	If the patient smokes	Boolean	0, 1
Time	Follow-up period	Days	[4,...,285]
(target) death event	If the patient died during the follow-up period	Boolean	0, 1

Fig 4.1

One of the main limitations of the dataset, we have an extremely imbalanced dataset. We have 203 patients who survived out of 299 while only 96 perished (Fig 4.2). This makes the dataset extremely imbalanced. It leads to highly inaccurate models especially those that try to maximize accuracy such as the many supervised machine learning methods.

```
1 df.DEATH_EVENT.value_counts()
```

```
0    203
1     96
Name: DEATH_EVENT, dtype: int64
```

Fig 4.2

Another limitation of the dataset is that the survival of the patient has been only through the first follow-up period. This way we don't know if the patient died within the period when the data was being collected post-follow-up or the time they died after the data collection as a whole. This may be an incorrect representation of the data at hand and their conditions.

There was not much cleaning of data that was required from our end due to it being a relatively cleaned set provided by previous researchers. However, some basic checks were run to ensure it was 100% clean such as checks for duplicates and incomplete values in the table. We further went on to divide the variables into categorical and numerical variables due to the way they were classified. The continuous variables have been classified as numerical variables and those with a boolean value as categorical variables. We have highlighted (Fig 4.3) those that have been classified as numerical variables out of the 12 parameters in our cleaning:

```
age                         0
anaemia                     0
creatinine_phosphokinase    0
diabetes                    0
ejection_fraction           0
high_blood_pressure         0
platelets                   0
serum_creatinine            0
serum_sodium                0
sex                         0
smoking                     0
time                        0
```

Fig 4.3

While trying to see if a dimension reduction was a possibility to avoid overfitting, the procedure was begun by running a correlation function on the data and was plotted

using SNS Heatmaps. The graph produced is attached below (Fig 4.4). From the graph, we can see that it is hard to identify if any variables are dependent on each other. The relatively low scores rather highlight that they may largely be independent.

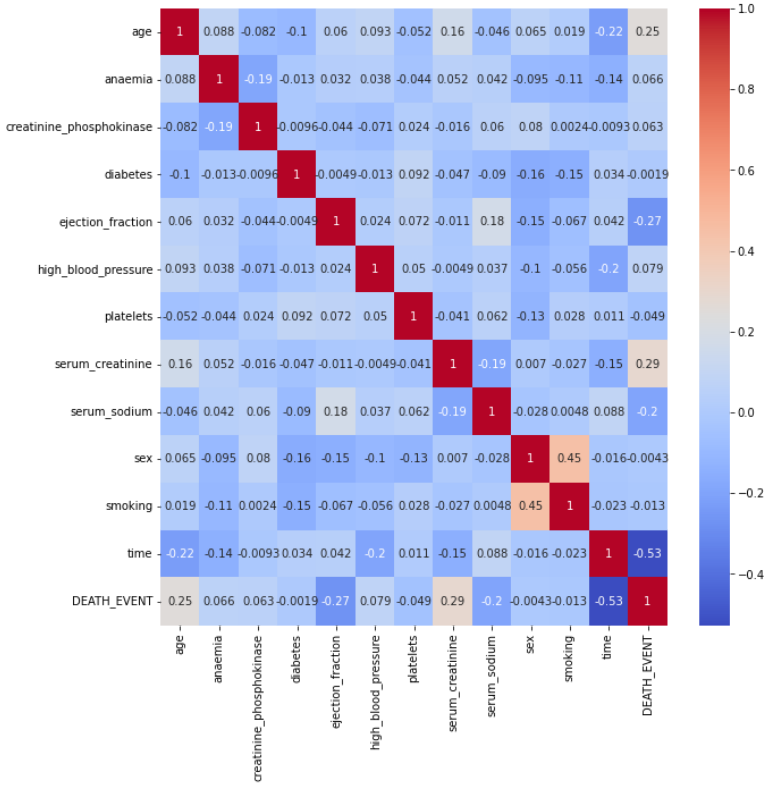


Fig 4.4

Performing a dimensional analysis and gaining insights on reduction from there on seemed more apt.

Due to the mix of variables, one will be unable to use something such as Principle Component Analysis which applies only numerical variables. When applied to such a combination of variables, the below would be the result while trying to understand the required components from the explained variance. The resulting graph (Fig 4.5) is produced:

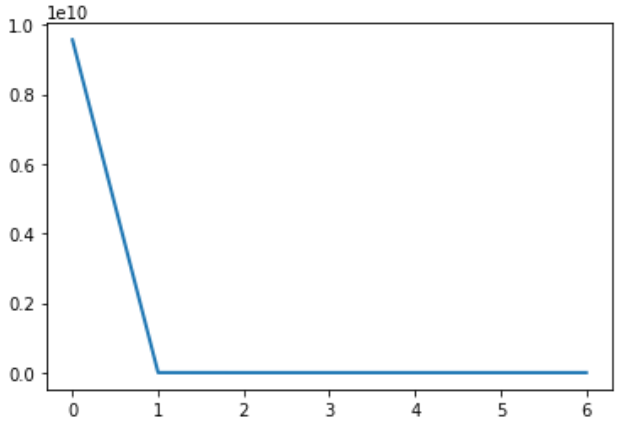
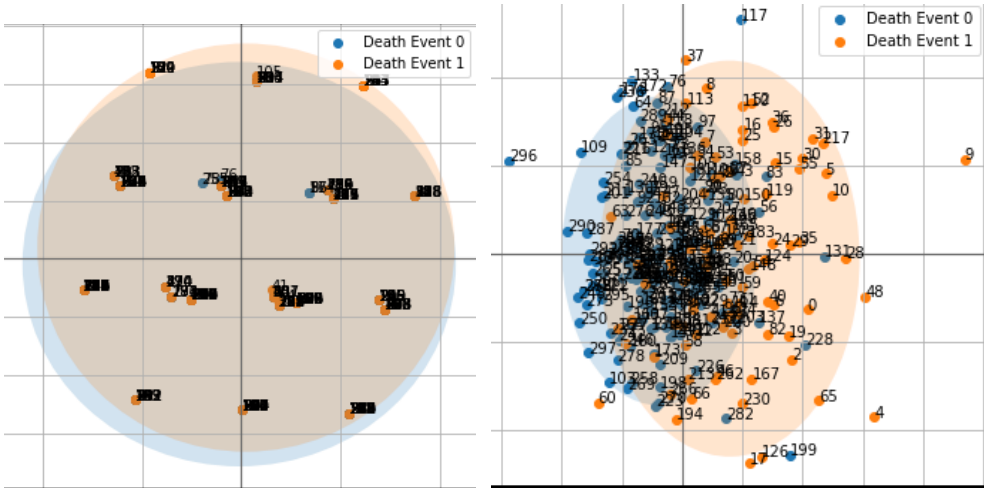


Fig 4.5

This graph indicates that only one component is required which seemed incorrect to us. While handling a mix of variables, such as the above one needs to perform a FAMD (Factor Analysis of Mixed Data), however, due to the way the data encoded in our dataset, we needed to look into the underlying program of FAMD to perform the attempt of dimension reduction.

FAMD inherently splits the variables into two categories and the theme performs a Multiple Factor Analysis, which tries to determine the correlation between the various parameters but they must be of the same category (i.e Numerical, categorical etc) and identify if the parameters are linearly separable. However, when the program was run on both the categorical and numerical variables, the results (Fig 4.6a - Categorical and Fig 4.6b - Numerical) did not indicate any sign regarding the correlation and which factor could be chosen to be eliminated. Therefore, we chose not to perform any form of dimension reduction.



(Fig 4.6a) (Fig 4.6b)

Further data exploration was produced to understand the data better and perhaps aid in perceiving limitations and to further the understanding and implications of the parameter on the final result.

Analysis was performed to look at the age and gender spread of the dataset. The results (Fig 4.7) show that there were significantly more males and well spread across all ages whereas there was a small number of females and very few above the age of 75. This may lead to some bias in the learning. Another limitation of the dataset is the significant number of more males in a disease in which men have a higher genetic risk of heart failure.

These two points are huge drawbacks for the dataset.

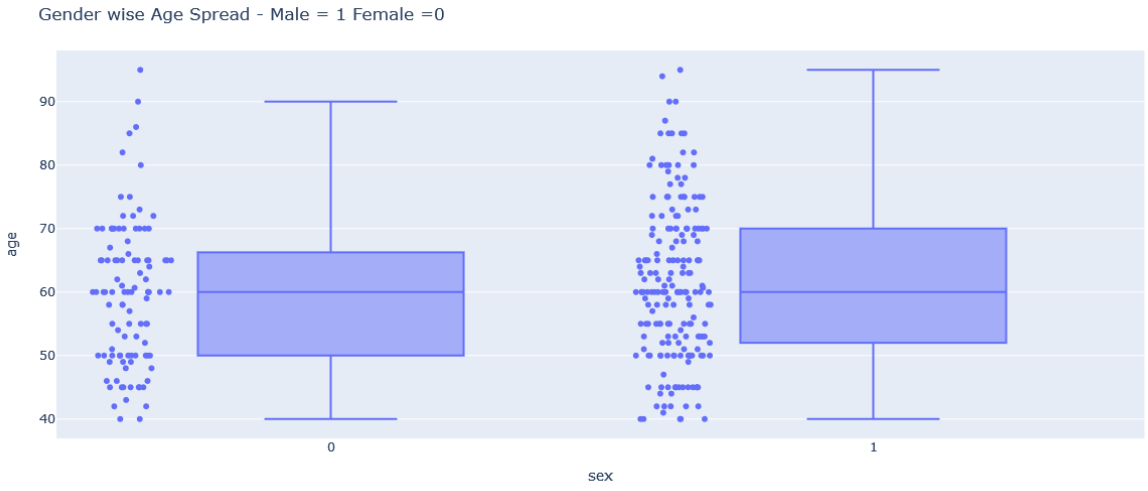


Fig 4.7

Drawing a distribution of the age through a histogram (Fig 4.8) also provided insights regarding the patients' survival. Those who were around the age of 40 always seemed to survive during the follow-up period whereas the older patients of the samples always seemed to not survive.

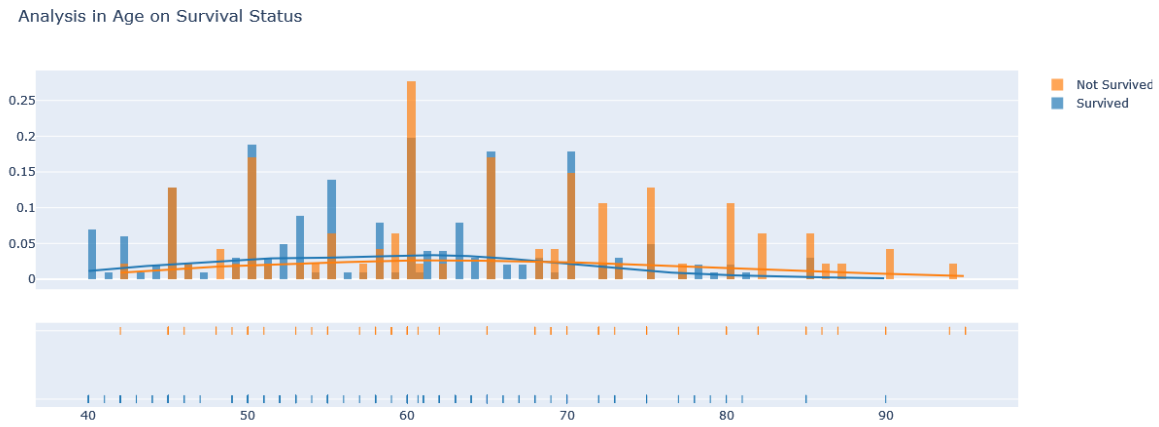


Fig 4.8

Performing analysis on the patients in tandem with the smoking parameter (Fig 4.9a) produced another result that would lead to incorrect learning. The equal number of patients who had survived who smoked and who did not smoke. Therefore the training set may further suggest smoking does not have an impact on heart failure. However, this is a false impression.

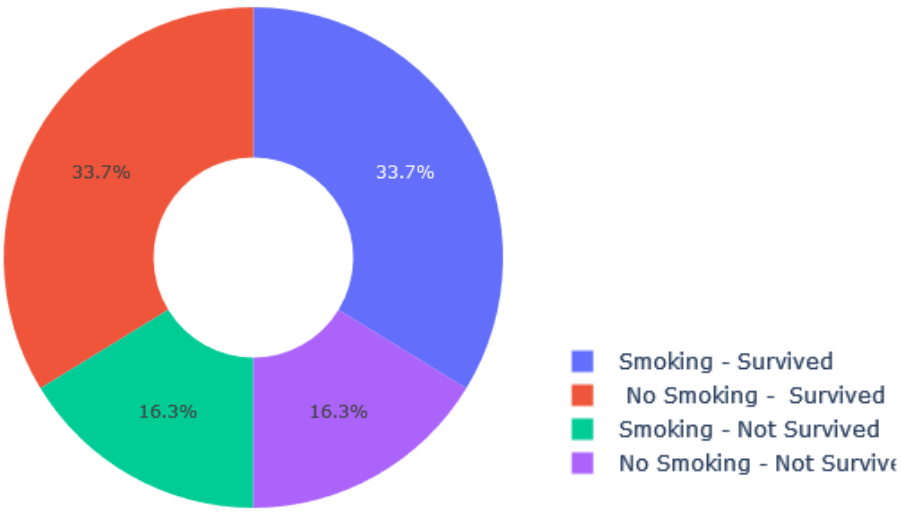


Fig 4.9a

Other comparisons of survival are based on a patient having high blood pressure, diabetes and anaemia (Fig 4.9b, Fig4.9c and Fig4.9d respectively).

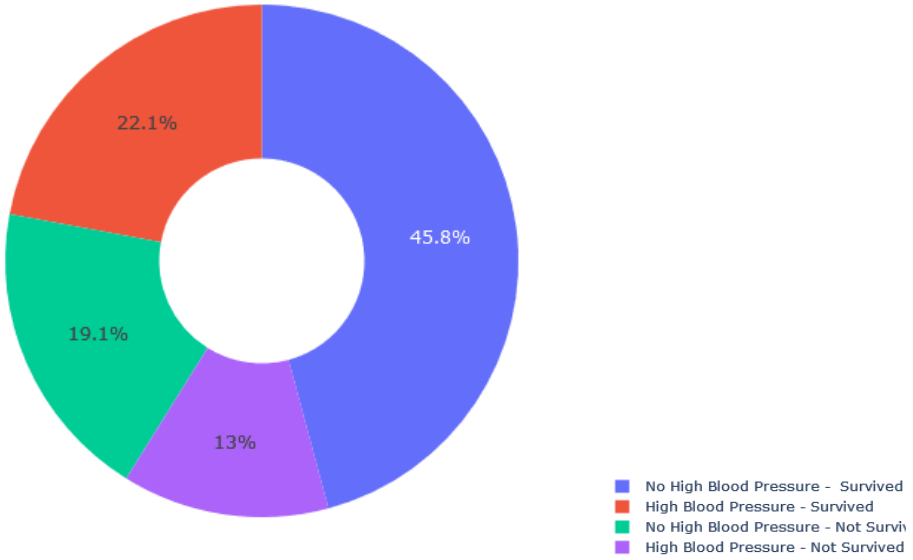


Fig 4.9b

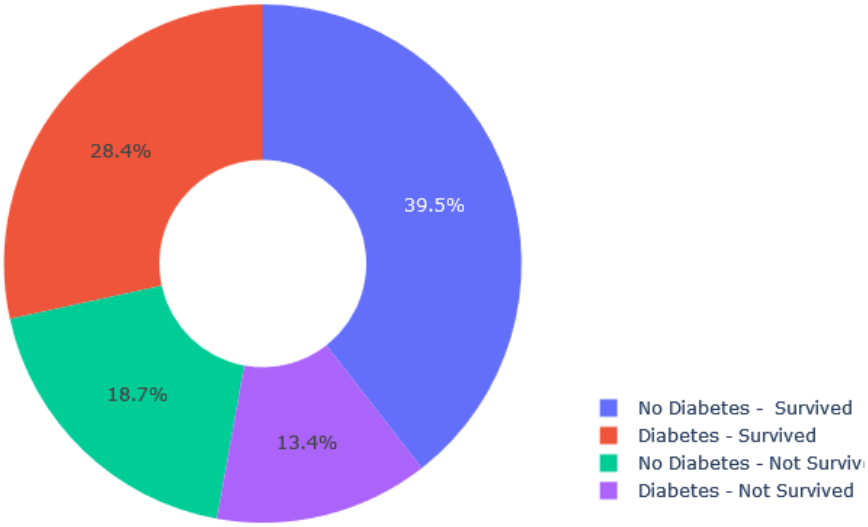


Fig 4.9c

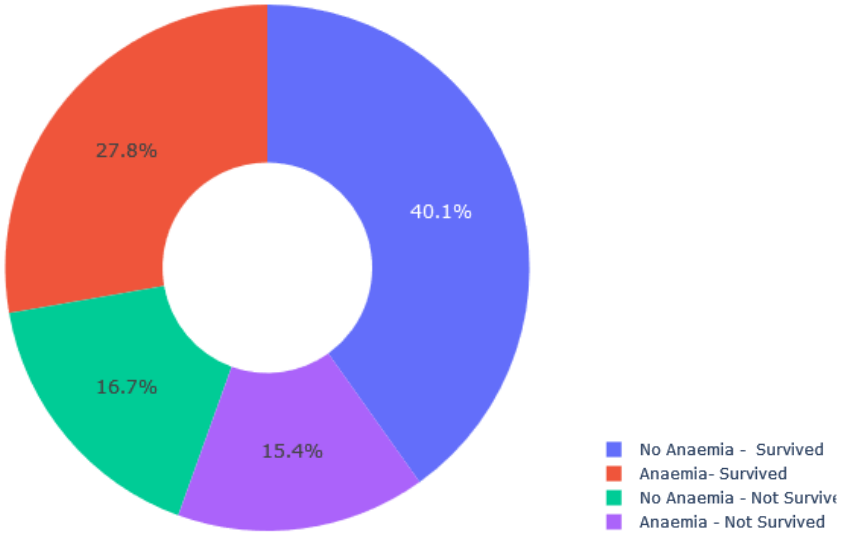


Fig 4.9d

The figure 4.9d in tandem with Anaemia also raises doubts as to the percentage of those who had and did not survive which are relatively close which may again influence the machine learning algorithms.

Another part of data exploration was to compare the patient's survival with continuous features such as Creatine Phosphokinase, Serum creatinine, Serum Sodium, Ejection fraction and Platelets counts (Fig4.10a,b,c,d and e respectively).

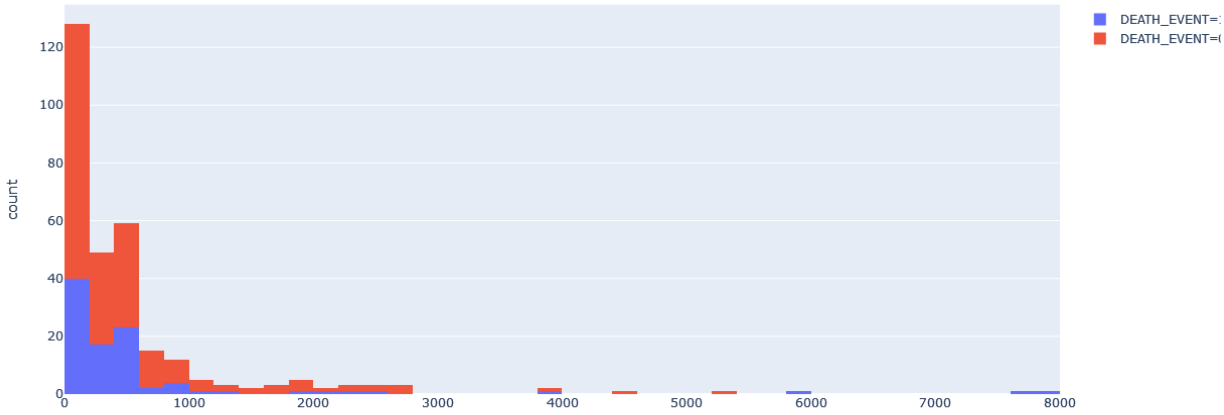


Fig 4.10a

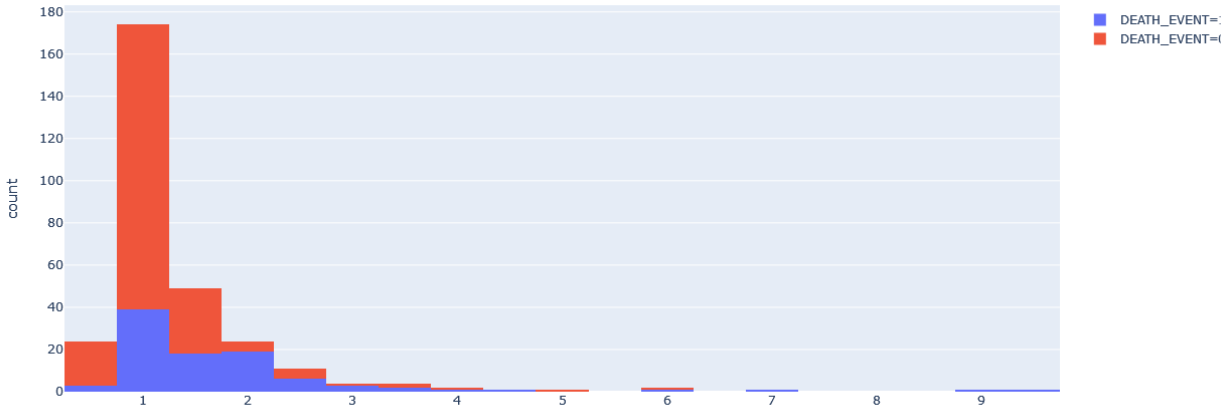


Fig 4.10b

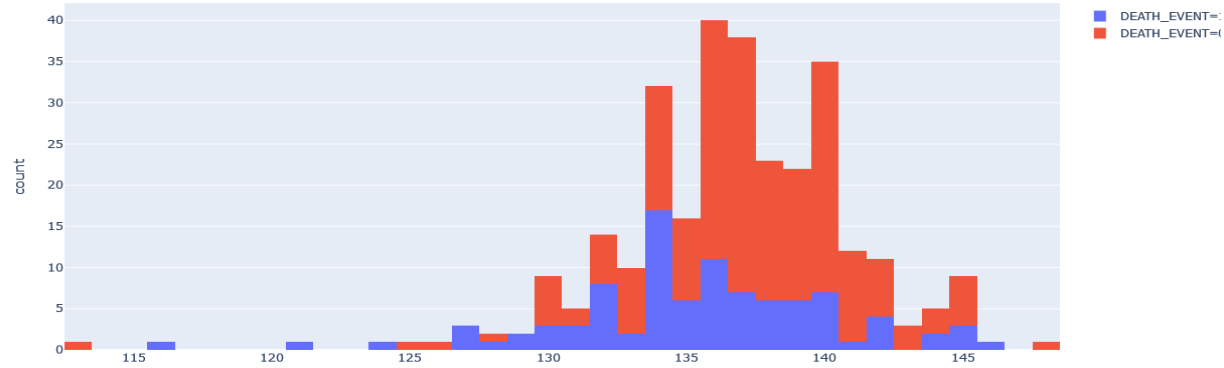


Fig 4.10c

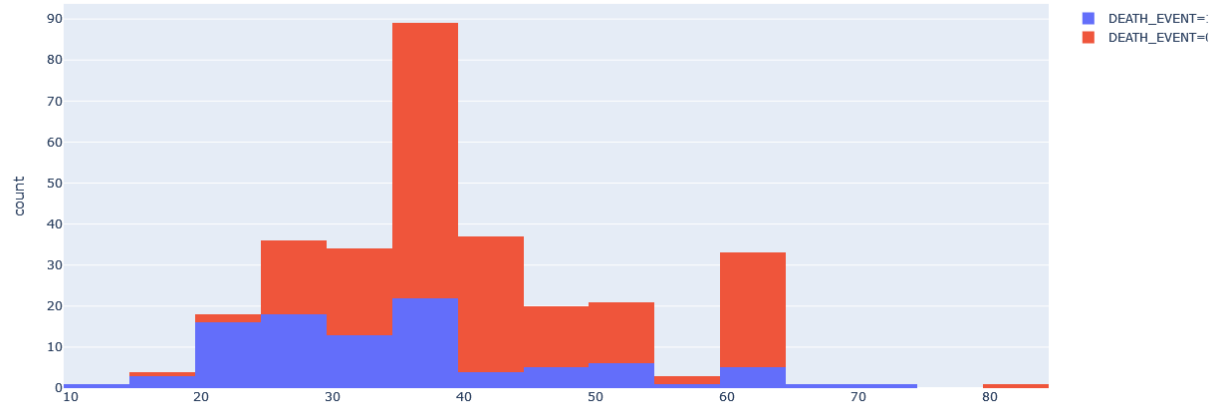


Fig 4.10d

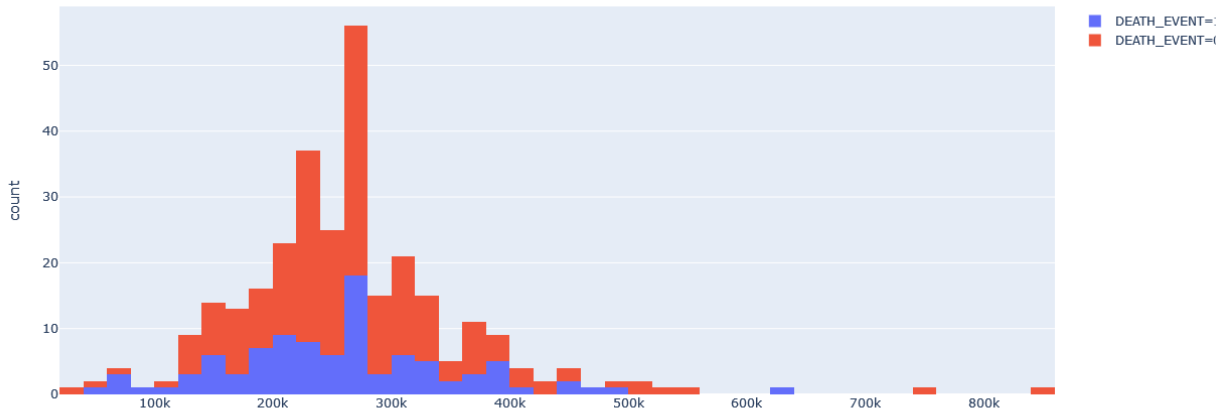


Fig 4.10e

The above graphs did not add more insight into how far each contributed just via a cursory glance.

- Methods & Implementation

Pre-processing Data:

- Feature Scaling - Normalization:

The next step was scaling our features. As discussed in the previous section on Dataset Specification, there are 12 features of varying degrees of magnitude, range, and units.
It can be noticed that the “age” feature uses “years” for measurement, whereas other specific medical parameters use different units of measurement. Machine Learning algorithms can be very sensitive to these features. So how can these features be used in the first place when they vary so vastly in terms of their measurements and ranges? This is where the concept of feature scaling comes in and becomes crucial while preprocessing the data. It is a way by which we can get our features to lie in a common range of values so that the model isn’t biased towards anyone becauseAdaBoost has a higher value.

For our study, we have chosen to use the “Normalization” technique for feature scaling. In this technique, values are shifted and rescaled so that they end up ranging between 0 and 1. It is also popularly known as Min-Max scaling. The formula (Fig 5.1) for normalization is as follows:

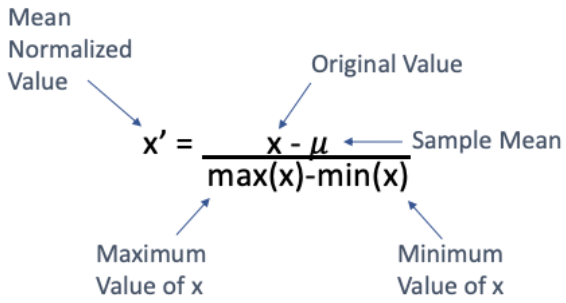


Fig 5.1

When the value of x is the minimum value in the column, the numerator will be 0, and hence x’ is 0. When the value of x is the maximum value in the column, the numerator is equal to the denominator and thus the value of x’ is 1.
If the value of x is between the minimum and the maximum value, then the value of x’ is between 0 and 1.

Data Resampling:

- Why resampling?

After normalizing, we could directly go on to splitting our data into training and test sets. However, this can lead to some misleading predictions. This is because the chosen data set is imbalanced. The target event i.e. the "Death Event" shows that 203 people out of 299 people survived during the follow-up period, whereas 96 people did not. One class is in the majority and one class is in the minority. Furthermore, the chosen dataset is very small for the task at hand. Thus, when standard classification algorithms are run without resampling, they tend to have a bias towards the majority class. As a result, the prediction gets biased towards only the majority class and major misclassification can occur in the minority class. This is because most of these algorithms have an accuracy oriented design. As long as the total number of wrong predictions are low, the algorithm thinks that it is predicting well. So, there is a need to resample the data.

- Stratified k - Fold Cross-Validation - The chosen method of resampling for our study:

There are many ways of resampling data. One very popular technique though is cross-validation as it is helpful especially in cases where the amount of data is limited. In this technique, a fixed number of folds (or partitions) of the data are made. Then, analysis is done on each and then the average overall error is estimated. Again, there are many methods of cross-validation, however, for our study, we have chosen "Stratified k-Fold Cross-Validation". This technique is essential "k -Fold Cross-Validation" with an additional benefit.

In k- Fold Cross-Validation, the data set is first randomly into k subsets (or folds). For each fold, the model is built on $k - 1$ folds of the dataset. The model is then tested to check the effectiveness of the k^{th} fold. This is repeated until each of the k folds has been the test set. Each time, the accuracy is recorded and the average of the k accuracy's serves as the main performance metric. This method (Fig 5.2) is beneficial because it means that every point of the dataset has appeared in both: the training and test set resulting in less bias.

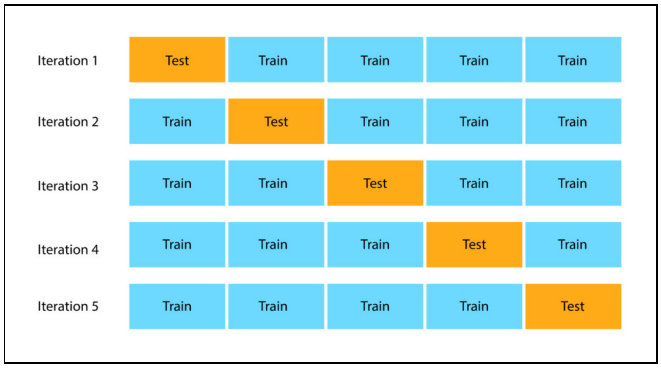


Fig 5.2

However, the disadvantage is that the k folds are made randomly. This means that there can be a chance of imbalance occurring in the folds which defeats the purpose of sampling in the first place. This is where stratified sampling comes into the picture. Stratification ensures that each fold is a good representative of the data by rearranging the data.

1. <u>Classification:</u>

After having pre-processed our data, it is now time to perform classification on the resampled dataset. We run our algorithm through the following classification algorithms:

a. SVM:

SVM (Support Vector Machine) is a very popular algorithm used in machine learning, especially when it comes to classification. In this algorithm, each data item is plotted as a point in n-dimensional space (where n is the number of features so in our case $n = 12$) with the value of a particular coordinate being the value of each of the n features. Then, classification is performed by finding a hyperplane that differentiates (Fig 5.3) the two classes.

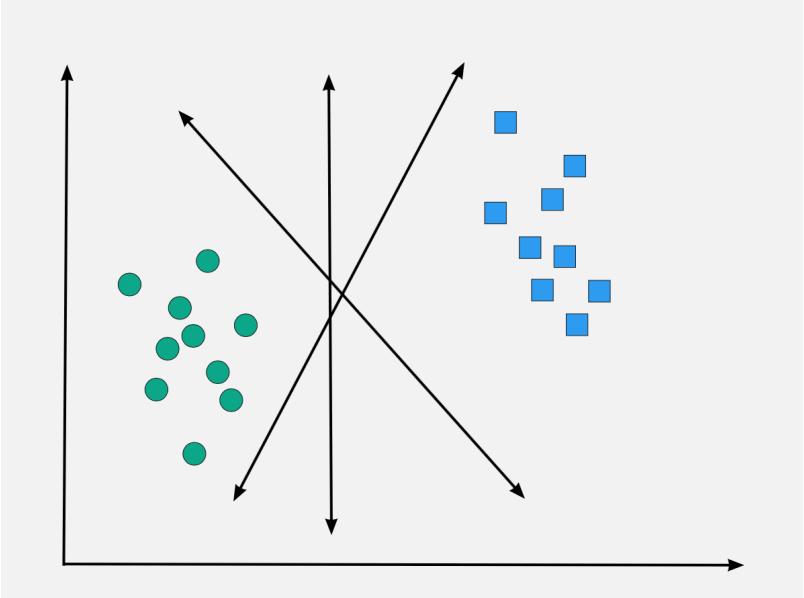


Fig 5.3

It can be seen that many different hyperplanes (Fig 5.4) can successfully separate the two classes. The goal, however, is to find a plane that has the maximum margin, i.e the maximum distance between data points of both classes:

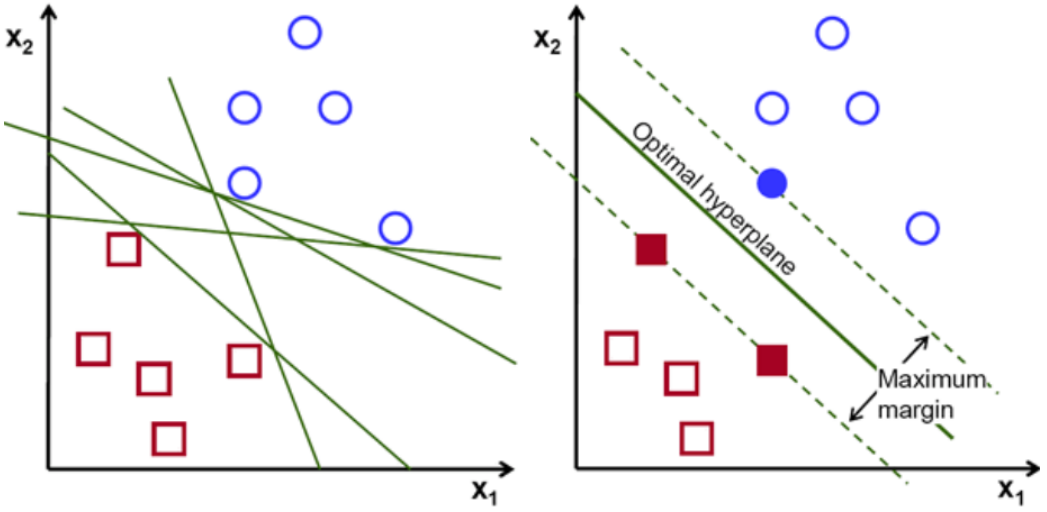


Fig 5.4

Maximizing the margin distance is necessary so that future data points can be classified with much more confidence.

SVM works very well without any modifications for linearly separable data, that is, any data that can be plotted in a graph and can be separated into classes using a straight line.

However, linearly separable data is rarely found in the real world. Thus, we use “kernelized SVM” for non-linearly separable data. In kernelized SVM, the data is mapped to a higher dimension to make it linearly separable. A kernel is a measure of similarity between data points. The kernel function in a kernelized SVM shows that given two data points, what is the similarity between the newly transformed space. There are different kernel functions present, but for hose three popular kernels: The Gaussian kernel, the linear kernel and the polynomial kernel:

- Gaussian or RBF (Radial Basis Function) kernel:

This is the most generalized form of kernels and is the default kernel in SVM due to its similarity between the gaussian distribution. It is mathematically represented (Fig 5.6) as:

$$K(X_1, X_2) = exp(-\frac{||X_1 - X_2||^2}{2\sigma^2})$$ Fig 5.6

where,
1. ‘σ’ is the variance, hyperparameter
2. $||X_1 - X_2||$ is the Euclidean Distance between two points X_1 and X_2

The maximum value of the RBF kernel is 1. This happens when $||X_1 - X_2|| = 0$, i.e. $X_1 = X_2$. When the points are the same, there is no distance between them. Thus, they are very similar.

However, when points are far apart and the distance between them is large, then the kernel value is less than 1 and close to 0 thus concluding that the points are dissimilar.

- Linear kernel:

This is also another popular and basic kernel. It is used as a normal dot product of any two given observations. It is generally used when there are a lot of features. This kernel is employed usually when the data is linearly separable.

- Polynomial kernel:

This is a generalized version of the linear kernel. Mathematically (Fig 5.7), it is defined as:

$$K(X_1, X_2) = (a + X_1^T X_2)^b$$

Fig 5.7

a. Logistic Regression

Logistic Regression is a popular algorithm to use in cases of Binary Classification. The nature of the target or deepening the dent variable is dichotomous, which means there would be only two possible classes.

b. K Nearest Neighbors

K-Nearest Neighbors is another algorithm that is commonly used in classification algorithms. Given a new data point, it assumes similarity between the new data point and the available data points. It then puts the new data point into the category that is similar to the available categories.

c. Adaptive Boosting

AdaBoost or Adaptive Boosting is a technique of iterative ensemble machine learning. Ensemble learning is a technique in which a combination of weak learners gives a strong learner. This algorithm sets weights to the classifiers and trains a sample of data in each iteration to ensure accurate predictions.

1. <u>Feature Selection:</u>

After having run classification algorithms, we were very curious to know which were the features that were having the most influence on our results. We wanted to do some sort of feature selection and then run the classification algorithms only using these features. Thus, we decided to go for Univariate feature selection. Univariate feature selection is a method to examine each
feature individually. This is done mainly to determine the strength of the relationship of the feature with our target variable and in general to gain a better understanding of the dataset. It is necessary however to optimize the collection of features. We chose to opt for scikit-learn's “SelectKBest” function to select the three features which have the highest scores. This method ranks features with the most “importance” or with the “highest scores” concerning the target variable. It does this by using a score function. Many score functions can be used, however, we used the “f_classif” which calculates the ANOVA F-value between the features.

Observations

Resampling:

We checked the difference in the accuracy of the algorithm when we did re-sampling vs when we did not. And here's a comparison between the two (on the SVM algorithm) (Fig 6.1):

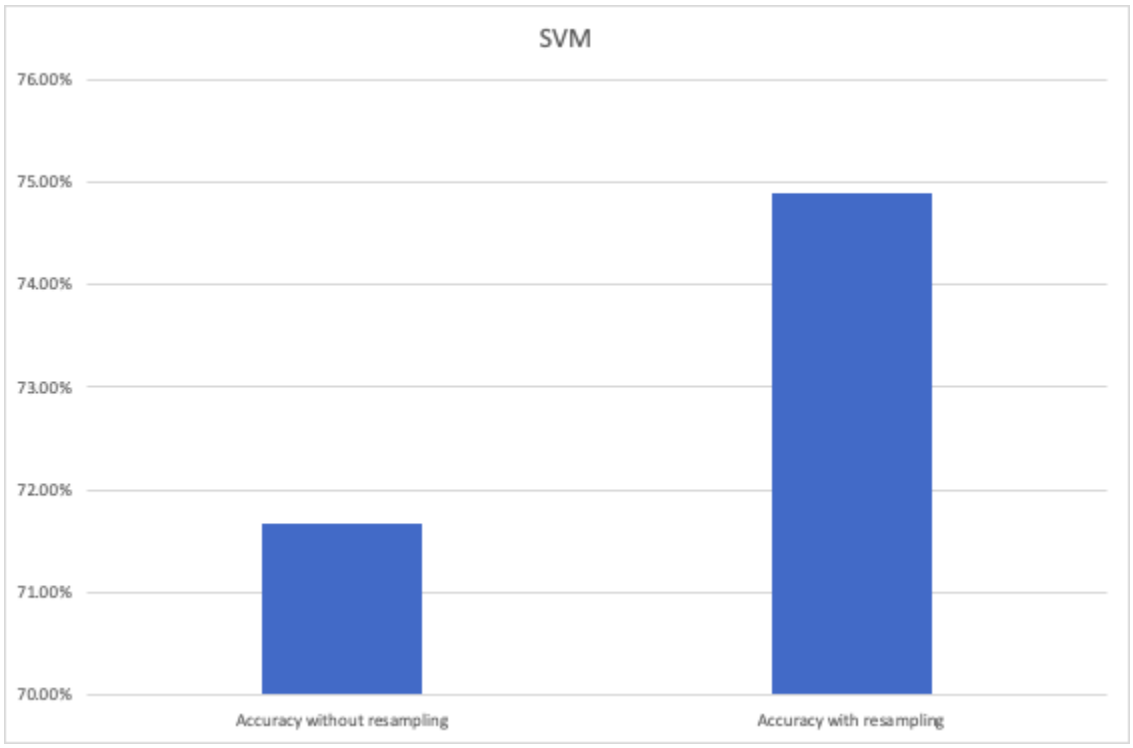


Fig 6.1

The accuracy with re-sampling is better than the accuracy without resampling. This is because resampling helped remove the inherent bias in the dataset towards the majority class. The Stratified *k* fold cross validation ensured that there is a balance of data points from both classes in each fold that it created. Thus, the accuracy with which the algorithm performed on test data was better than when no resampling was done.

Classification:

- Accuracy

Here is the accuracy of all six algorithms (Fig 6.2):

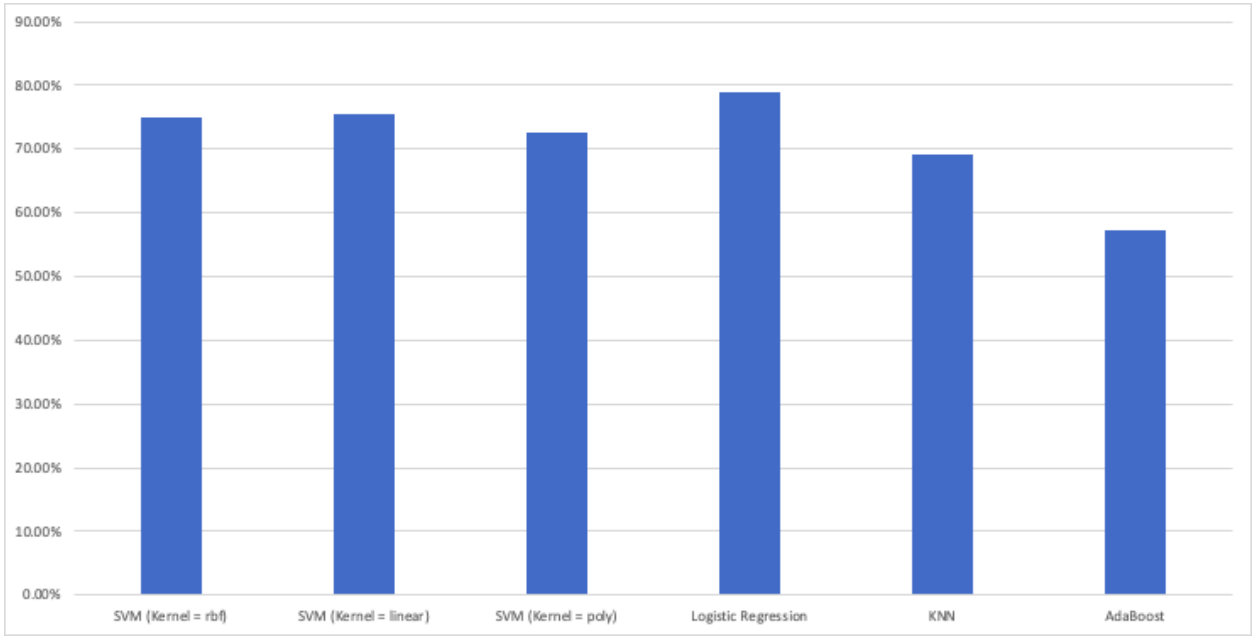

90.00%
80.00%
70.00%
60.00%
50.00%
40.00%
30.00%
20.00%
10.00%
0.00%
SVM (Kernel = rbf)
SVM (Kernel = linear)
SVM (Kernel = poly)
Logistic Regression
KNN
AdaBoost

Fig 6.2

- Confusion Matrices

Since we have an imbalance dataset, a good evaluation technique is to use confusion matrices.
A confusion matrix is an *N x N* matrix, where *N* is the number of target classes. The matrix (Fig 6.3 and 6.4)) compares the actual target values with those predicted by the machine learning model.

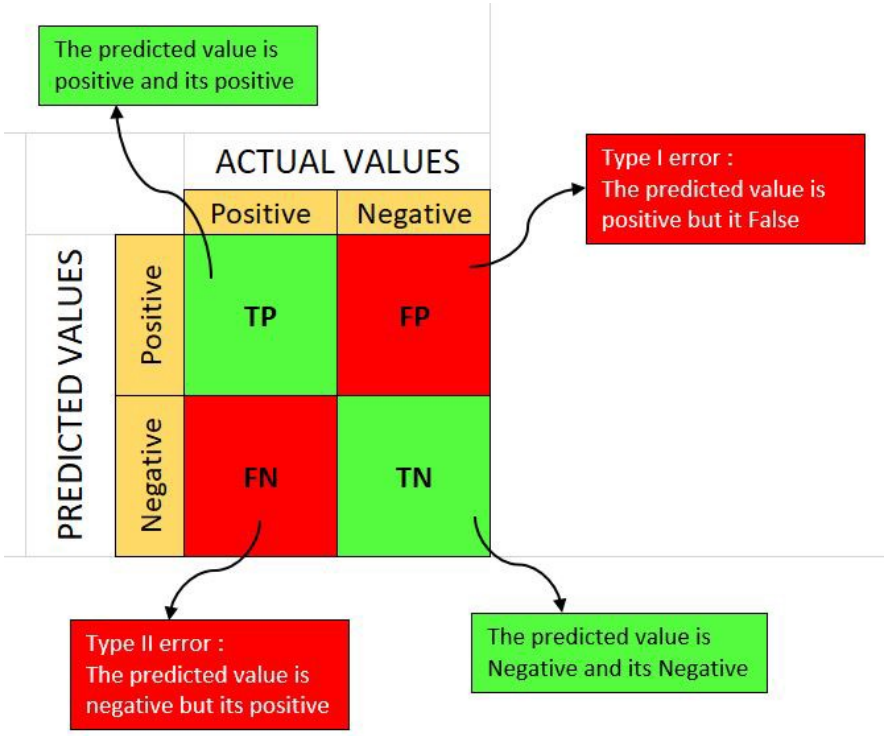
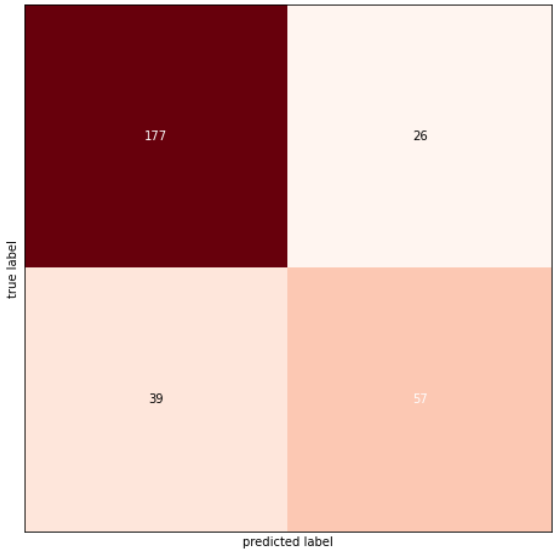


Fig 6.3

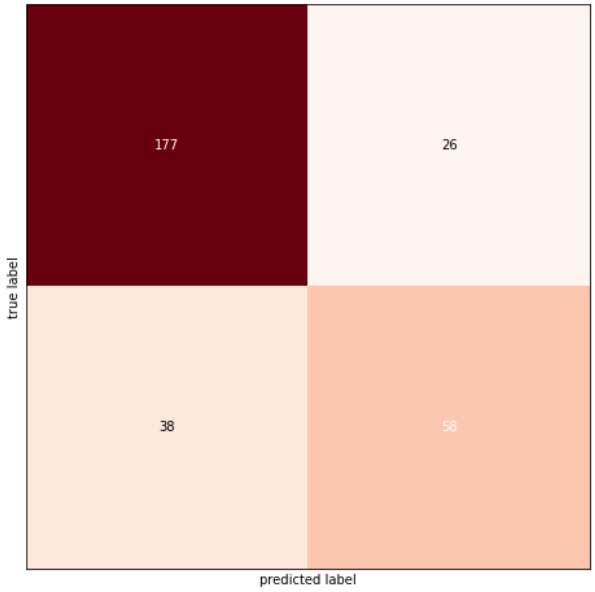
		Predicted	
		Negative **(N)** -	Positive **(P)** +
Actual	Negative -	True Negative **(TN)**	**False Positive (FP) Type I Error**
	Positive +	**False Negative (FN) Type II Error**	True Positive **(TP)**

Fig 6.4

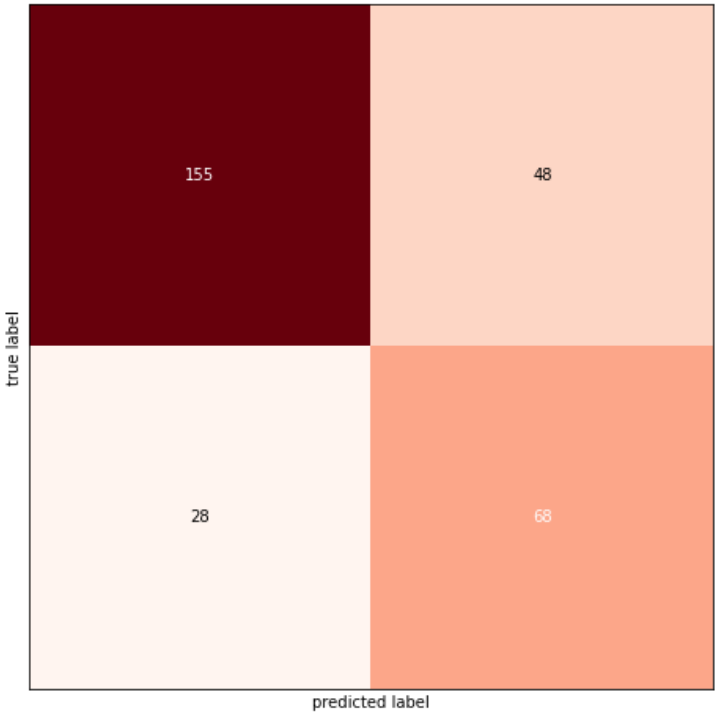
SVM (kernel = “rbf”) Fig 6.5.a



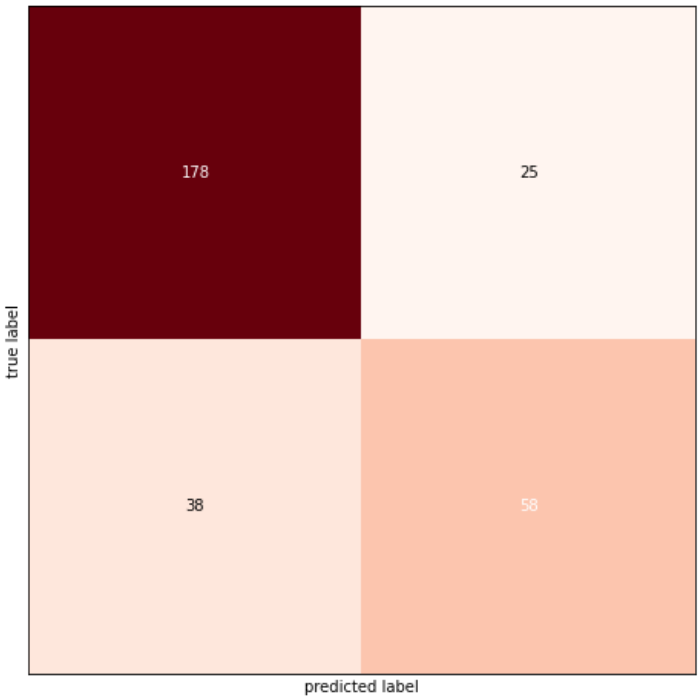
SVM (kernel = “linear”) Fig 6.5.b



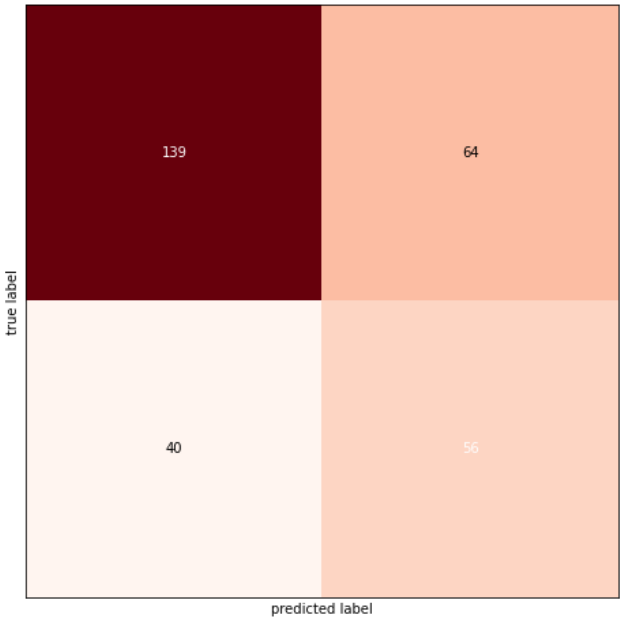
SVM (kernel = “poly”) Fig 6.5.c



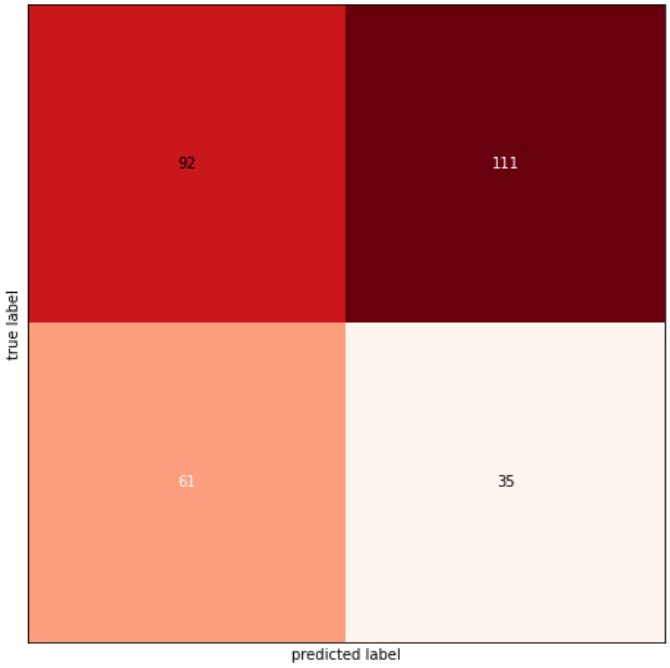
Logistic regression Fig 6.5.d



K-nearest neighbours Fig 6.5.e



AdaBoost Fig 6.5.f



Feature Selection:

When we ran the SelectKBest algorithm on our dataset, we got the following as the 3 features with the highest scores:

a. Ejection Fraction
b. Serum Creatinine
c. Time

So, we ran our six algorithms only on these 3 features and here is a comparison of the accuracy Fig 6.6 in both cases:

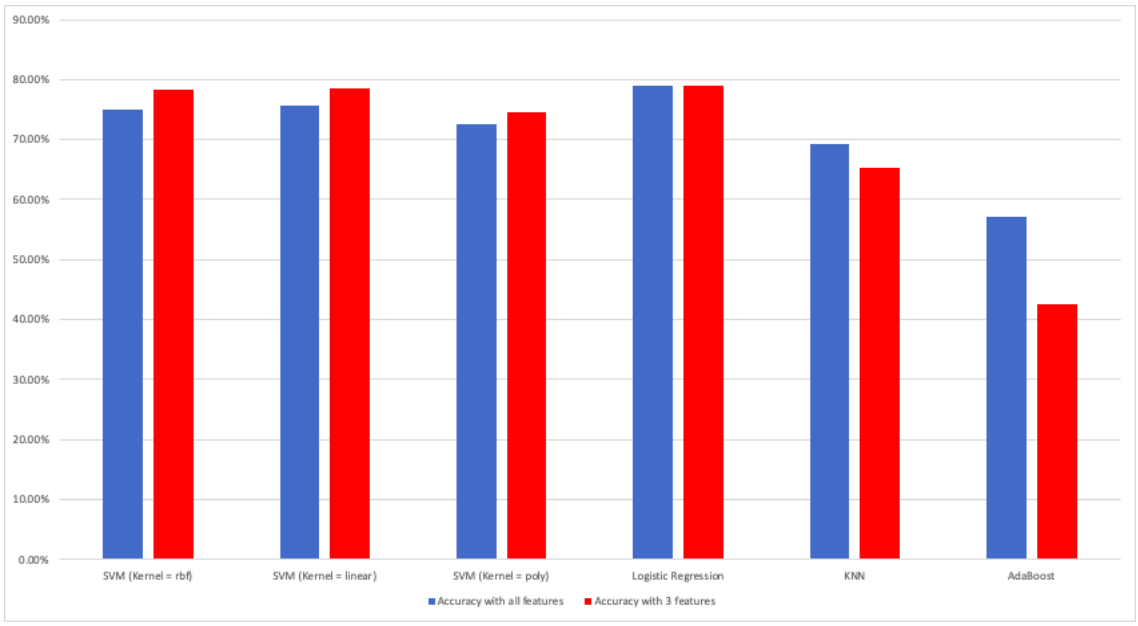


Fig 6.6

We can see that in all cases, the accuracy has remained more or less the same. This implies that the 3 features are indeed the strongest concerning our target variable.

The logistic regression model seems to be doing the best, closely followed by the SVM linear kernel. We believe this is happening due to the small-sized nature of our dataset combined with the fact that the two parameters, Serum Creatinine and Ejection Fraction are linearly separable in nature, leading to the highest accuracy.

Algorithms like ADABoost seem to do worse when the features are reduced as the depth of the tree reduces, impacting the accuracy of the algorithm.

The algorithm accuracy also seems to highly be equal when performed with all features vs just the three top features via the univariate test, which suggests a high relation between Serum Creatinine and Ejection Fraction and heart failure. This revalidates the results produced by both of those in the above research papers.

Limitations:

As mentioned above, the dataset is highly imbalanced in terms of the number of patients who survived. With this being a target feature, this has massive impacts on how learning occurs based on the dataset. Another limitation is the spread of gender and age as mentioned above. The spread of patients with conditions also seemed equally distributed in many cases impacting the data and its learning. The small dataset can also lead to overfitting when these algorithms are tried on real-time data. The limitations of the follow-up period and their time of death if they did, is another misrepresentation of the data.

Methods of improving:

The research needs to be repeated with a more balanced dataset and a larger dataset. The dataset also must be taken from different parts of the world as dietary constraints play a huge role in heart diseases, to reconfirm the results. The dataset not only needs balance in terms of patients who survived and did not but also in terms of gender and age, and take into consideration those that make certain groups more vulnerable when included in the dataset. The follow-up period also needs to be of a larger time scale or rather the data must keep track of what may be considered a significant time post the follow-up period of collection to consider the death not part of the results.

Advantages:

The parameters selected were highly what could be considered to be indicators or highly related to those who may soon have heart failure. Apart from that the data was largely cleaned and did not much vetting from our end.

- Conclusion:

On the whole, we believe that Serum Creatinine and Ejection Fraction are good indicators of those who may soon die of heart failure. Supervised Machine learning methods therefore can be used to predict heart failure with an accuracy of approximately 79%.

We learnt how to handle an imbalanced dataset and how to provide more accurate machine learning models with such datasets. We also had a much better understanding as to where the models may fail and why they would in certain cases such as ADA Boosting.

We were unable to produce a graph on the linear separability of the two features due to the lack of time.

We had multiple bugs and hurdles we have tried to overcome during the project. Starting with trying to understand why dimension reduction would not be suitable in this case to being unable to produce different results for the various models unless we ran them on different code files. This led to a manual collection of accuracy which may have therefore been trained on different training and test sets to show their accuracy. Another bug we encountered was the incorrectly updating of our accuracy produced by our various folds which when edited led to little different results but more accurate results in our opinion.

The results also strongly verified that Serum Creatinine and Ejection Fraction were highly related to heart failure.

With more time not only could we maybe graphically show linear separability than just interpret it from the high success mirrored by both Logistic Regression and the SVM linear kernel. A limitation of our project is that there is no method in verifying our results, if we could collect our try to predict using real-time data, this would help our verification. Getting a doctor's opinion on what other parameters that they look at while trying to care for a patient with heart failure, can help rebuild the new dataset and add more value to the study.