DM1590 Machine Learning for Media Technology
2022-03-18

Predicting Death from Heart Failure

Group 4

Matias Eriksson, Elin Lundqvist, Mille Kåge & William O'Grady

---

Table of contents

Abstract

Using the dataset ‘Heart Failure Clinical Records’, we have applied the Machine Learning process of training and testing classification models to predict the survival of patients who’ve suffered from cardiac arrest. The project uses methods of feature selection, unsupervised and supervised learning to achieve our results. In order to predict the outcome and evaluate the accuracy of our models, we constructed a loop that splits the data into random partitions, and therein creating an average estimation of the best accuracy. We also investigated the dataset’s different features and their necessity in predicting the chance of survival. The result is a model based on decision-trees that with an average accuracy of 83.3% is able to correctly predict a patient’s outcome. The models always favour using few features (in this case 2) instead of all the features and the most chosen features were ‘time’, ‘serum-creatinine’ and ‘ejection-fraction’.

Background and motivation

The goal of this project is to be able to predict if someone will die after they've suffered from heart failure. There are many parameters that in theory could give an estimation of the outcome, whether it's the kind of medication they use, their previous health status or their gender; with all this data you could apply machine-learning to predict patient cases. What is even more interesting would be if only a few features, say two, could perform as well or even better than a model with all of the patient's attributes. We are by no means doctors, so we won't comment on the medical implication of our findings, but hypothetically a model like this could be of real help in a hospital or be used as evidence in medical research.

The main goal, however, was always to experiment and create something of our own with what we've learned about Machine Learning. We took the time to analyse our results, because what's more important than getting good results is defining what 'good' really means and how we got there.

Dataset

Our dataset contains the medical records of 299 patients who suffered from heart failure. It was collected during their follow up period and consists of 13 different features per patient. The features are both binary, like as if they had diabetes, if they smoked or their sex, and continuous like their ejection fraction, level of serum creatinine and platelets.

The data set was collected at the Faisalabad Institute of Cardiology and at the Allied Hospital in Faisalabad (Punjab, Pakistan), during April–December 2015.[1] It was retrieved from the University of California Irvine's website, to which it was donated 2020-02-05.[2]

To perform a binary classification on the data set the feature *Death Event* was used. It states whether the patient died or survived before the end of their follow-up period. The length of the follow-up period varies from patient to patient, represented by the parameter *time*. Also worth noting is that the data set is unbalanced, where 203 patients survived, and 96 died.[3]

The table below shows all features in greater detail.

[1] Chicco, D., Jurman, G. Machine learning can predict survival of patients with heart failure from serum creatinine and ejection fraction alone. (2020)
[2] Anvir Ahmad et.al. Heart failure clinical records data set. UCI Machine learning repository. (2020)
[3] Chicco, D., Jurman, G. Machine learning can predict survival of patients with heart failure from serum creatinine and ejection fraction alone. (2020)

Feature	Explanation	Measurement	Range
Age	Age of the patient	Years	[40,..., 95]
Anaemia	Decrease of red blood cells or haemoglobin	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]
Diabetes	If the patient has diabetes	Boolean	0, 1
Ejection fraction	Percentage of blood leaving the heart at each contraction	Percentage	[14,..., 80]
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

Methods

Visualising the data

The first step for us, to understand the scope of our project and our selected dataset, was to visualise it in different ways. This gave us insight into the thirteen parameters (features) that make the dataset, and what we could read from them.

We used a number of methods to visualise the dataset, the first of which was plotting histograms for each parameter. The histograms differed as some parameters were binary (*high blood pressure* could only be 0 or 1) while others ranged on numerical scale (*age* could be 0-100). The histograms made it plain to see what trends existed in the dataset. (see: appendices)

Another method used was the Pearson Correlation Matrix. It shows the correlation between all features in the dataset. We used it to discover correlation between the features, in order to predict what features were going to play an important role later on (see: figure 1 in Results).

t-SNE is a way of projecting data into a low dimensional space while preserving high dimensional operations such as clustering. More commonly it is used for visualisation of high dimensional data using two-dimensional scatter plots. Using the t-SNE method, our dataset was clustered into groups of zeroes and ones, representing subjects that had survived (zero) or died (one). For example, what we gained from these groupings was that there is a link between the ones and the zeroes; otherwise the clusters would be a random mishmash.

When we saw that the features Time, Serum-creatinine and Ejection-fraction had a high correlation with the death event we decided to plot the patient's outcome with two of those features as the x and y axis. This generated plots that could give us an indication if it was possible for the model to differentiate between the two outcomes. (see: figure 3, 4 in Results)

Feature selection

To find the most important features of our dataset, we chose the SelectKBest method. There are a number of different parameters you could use within this method but the parameter we used was *f_classif* since it reliably gave us good results. We chose SelectKBest because we were familiar with it and it was a quick and easy way to get the results we were looking for. With a relatively limited dataset, we wanted to see how well the model performed with only two features, so *k* was made equal to 2.

Splitting the data

Firstly, using an 80/20 split, the data was partitioned into a train set and a test set. The training set was then split again into a train set and validation set. Resulting in a total of 3 sets with the following portions : Training set: 64%, Validation set: 16%, and Test set: 20%.

The need of a validation set was entirely motivated by the goal to ensure the supervised methods were being performed under the best possible conditions. The validation set makes it possible to tune the parameters, in order to get the best possible results when testing the data.

Supervised learning methods

Decision Trees

Decision trees are used for classification and regression tasks. It works by setting up a binary tree where each node works as a true-or-false question regarding certain values. By following the questions, the tree predicts what class a data point belongs to based on how it stands regarding these questions. Decision trees work well for data that have binary and continuous features. Because the dataset examined had these characteristics the decision was made to utilise this type of supervised learning.

It's easy to overfit this type of model. To mitigate the effects of this there are two methods often used, pre-pruning and post-pruning. Pre-pruning was used in our implementation, this took the form of limiting the depth of the tree. This was done by using the parameter *max_depth*. The choice was made that choosing a smaller *max_depth* for the decision tree stopped the overfitting of the tree and improved the accuracy of the validation set.

K Nearest Neighbour

KNearestNeighbor, or KNN, is a common classification method that uses distance between data points to create a decision boundary between different classes. The K in KNN corresponds to the number of neighbours the model takes into account when creating this decision boundary. If you look at more data points the model tends to underfit the data and vice versa. It is therefore often necessary to fine-tune this parameter which is what we did in this project. Something else to consider when working with KNN is the curse of dimensionality which tells us that the more dimensions that are present, the larger the distance between data points becomes. The implication of this is something that we will discuss further in the discussion part.

Support Vector Machines

SVMs, an abbreviation of (Kernelized) Support Vector Machines, was the final supervised method used in the project. More specifically, we used the *SVC* implementation of the method, which works with classification. Of the three supervised learning methods used in the project, SVC was the one we had the least experience with. This is why we chose it, to

gain insight into methods that we hadn't used before. Support Vector Machines establish a threshold between observations to create classes. When we allow for misclassifications, the distance between the observations and the threshold is called a *soft margin.* When we use the soft margin to determine the location of a threshold, we are using a *Support Vector Classifier*, hence SVC. (The observations on the edge and within the soft margin are called *support vectors).* SVMs allow for complex decision boundaries, which are created by tuning the two parameters *C* and *gamma*.

While SVMs often perform quite well, especially on smaller datasets, they are very sensitive to the parameter settings and to the scaling of the data. In particular, it is vital to have all the features vary on a similar scale (if not, it overfits). It is recommended to manually scale the features by subtracting the minimum values of the individual features and dividing by the range of values, and then fitting the model to this new scaled set. This produces an underfitting regime where you only need to fine tune the parameters for a drastically improved result.

Prediction and evaluation

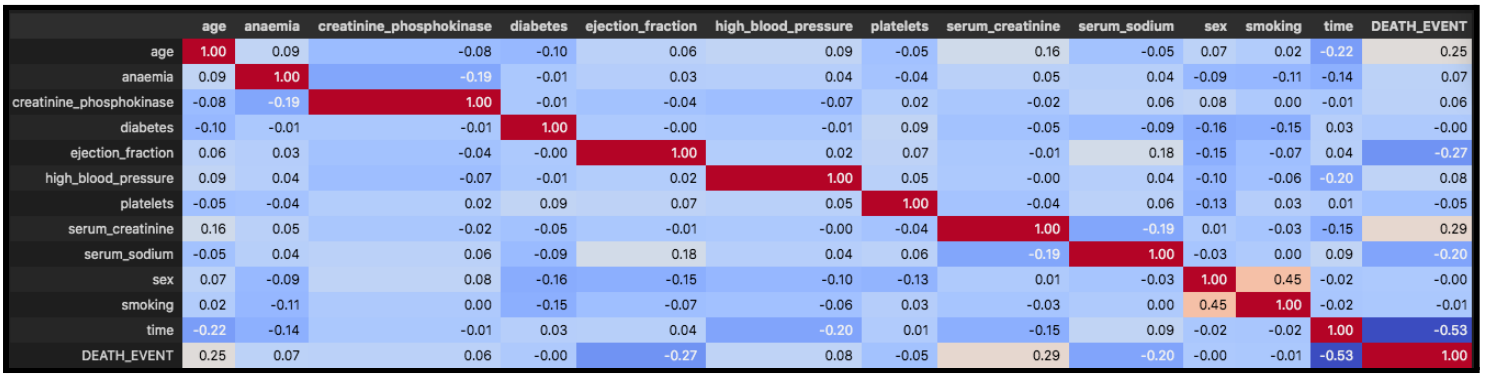
We used a number of methods in getting our results from the models. The accuracy of the models was most important. This is why we ran the supervised models, Decision Tree, KNN and SVC, through a loop which we referred to as '*The Loop*' a hundred times each. Inside *The Loop* there contained the SelectKBest method of feature selection, the tuning of the parameters using the validation sets, and a continuous process of producing a mean accuracy of the model. The accuracy was actually two numbers: one mean accuracy when testing the model with *all* features, and the other only testing on the *selected* features. (See Figure 7). The parameters the model used were stored for later analysis.

To evaluate *The Loop* further confusion matrices were created for each model, for all features and for the selected features respectively. These revealed flaws with early versions of the loop, for instance they first showed that one of the models had assumed that every patient survived. (See figure 9)

Another method of evaluation was to look closer at how the selected features differed when running the loop a hundred times. What feature-pairs were most commonly picked? Since SelectKBest always picked two features, which features seemed to pair well? These questions cropped up when using this evaluation method.

Results

Figure 1:
The Pearson Correlation Matrix



	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatinine	serum_sodium	sex	smoking	time	DEATH_EVENT
age	1.00	0.09	-0.08	-0.10	0.06	0.09	-0.05	0.16	-0.05	0.07	0.02	-0.22	0.25
anaemia	0.09	1.00	-0.19	-0.01	0.03	0.04	-0.04	0.05	0.04	-0.09	-0.11	-0.14	0.07
creatinine_phosphokinase	-0.08	-0.19	1.00	-0.01	-0.04	-0.07	0.02	-0.02	0.06	0.08	0.00	-0.01	0.06
diabetes	-0.10	-0.01	-0.01	1.00	-0.00	-0.01	0.09	-0.05	-0.09	-0.16	-0.15	0.03	-0.00
ejection_fraction	0.06	0.03	-0.04	-0.00	1.00	0.02	0.07	-0.01	0.18	-0.15	-0.07	0.04	-0.27
high_blood_pressure	0.09	0.04	-0.07	-0.01	0.02	1.00	0.05	-0.00	0.04	-0.10	-0.06	-0.20	0.08
platelets	-0.05	-0.04	0.02	0.09	0.07	0.05	1.00	-0.04	0.06	-0.13	0.03	0.01	-0.05
serum_creatinine	0.16	0.05	-0.02	-0.05	-0.01	-0.00	-0.04	1.00	-0.19	0.01	-0.03	-0.15	0.29
serum_sodium	-0.05	0.04	0.06	-0.09	0.18	0.04	0.06	-0.19	1.00	-0.03	0.00	0.09	-0.20
sex	0.07	-0.09	0.08	-0.16	-0.15	-0.10	-0.13	0.01	-0.03	1.00	0.45	-0.02	-0.00
smoking	0.02	-0.11	0.00	-0.15	-0.07	-0.06	0.03	-0.03	0.00	0.45	1.00	-0.02	-0.01
time	-0.22	-0.14	-0.01	0.03	0.04	-0.20	0.01	-0.15	0.09	-0.02	-0.02	1.00	-0.53
DEATH_EVENT	0.25	0.07	0.06	-0.00	-0.27	0.08	-0.05	0.29	-0.20	-0.00	-0.01	-0.53	1.00

Figure 2:
Column 13 of the Pearson Correlation Matrix, where the features correlation to the *Death Event* feature are shown.

Feature	**Death Event**
Age	0.25
Anaemia	0.07
Creatine phosphokinase	0.06
Diabetes	-0.00
Ejection fraction	-0.27
High blood pressure	0.08
Platelets	-0.05
Serum creatinine	0.29
Serum sodium	-0.20
Sex	-0.0
Smoking	-0.01
Time	-0.53
Death Event	1.00

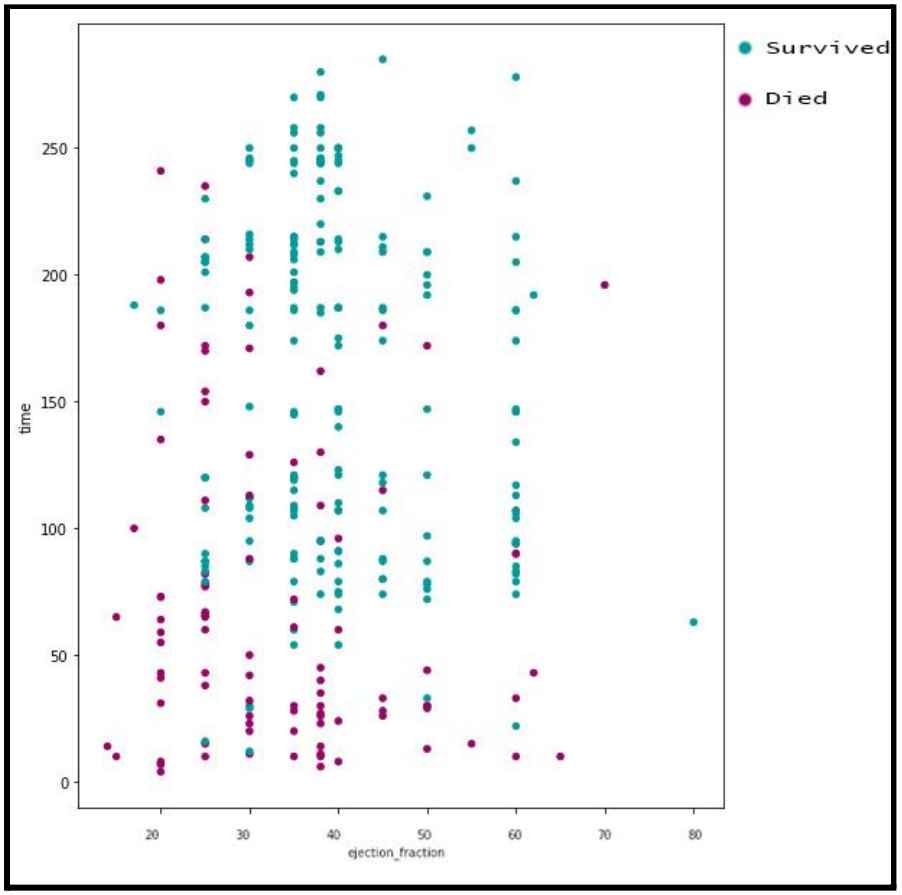


Figure 3:
A plot of features *time* (y-axis) to *ejection fraction* (x-axis). Each data point is colored, blue to indicate the patient surviving, purple to indicate the patient dying.

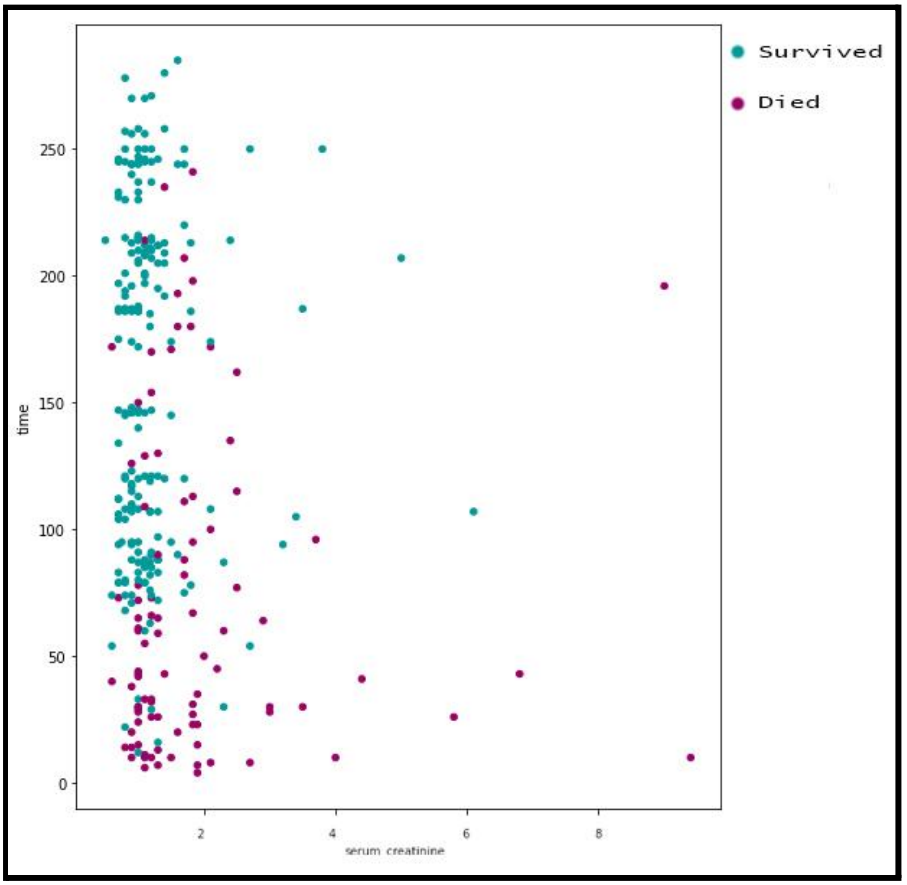


Figure 4:
A plot of features *time* (y-axis) to *serum creatinine* (x-axis). Each data point is colored, blue to indicate the patient surviving, purple to indicate the patient dying.

Figure 5:
t_SNE plot. Each data point is marked with 1 to indicate death, or 0 to indicate survival.

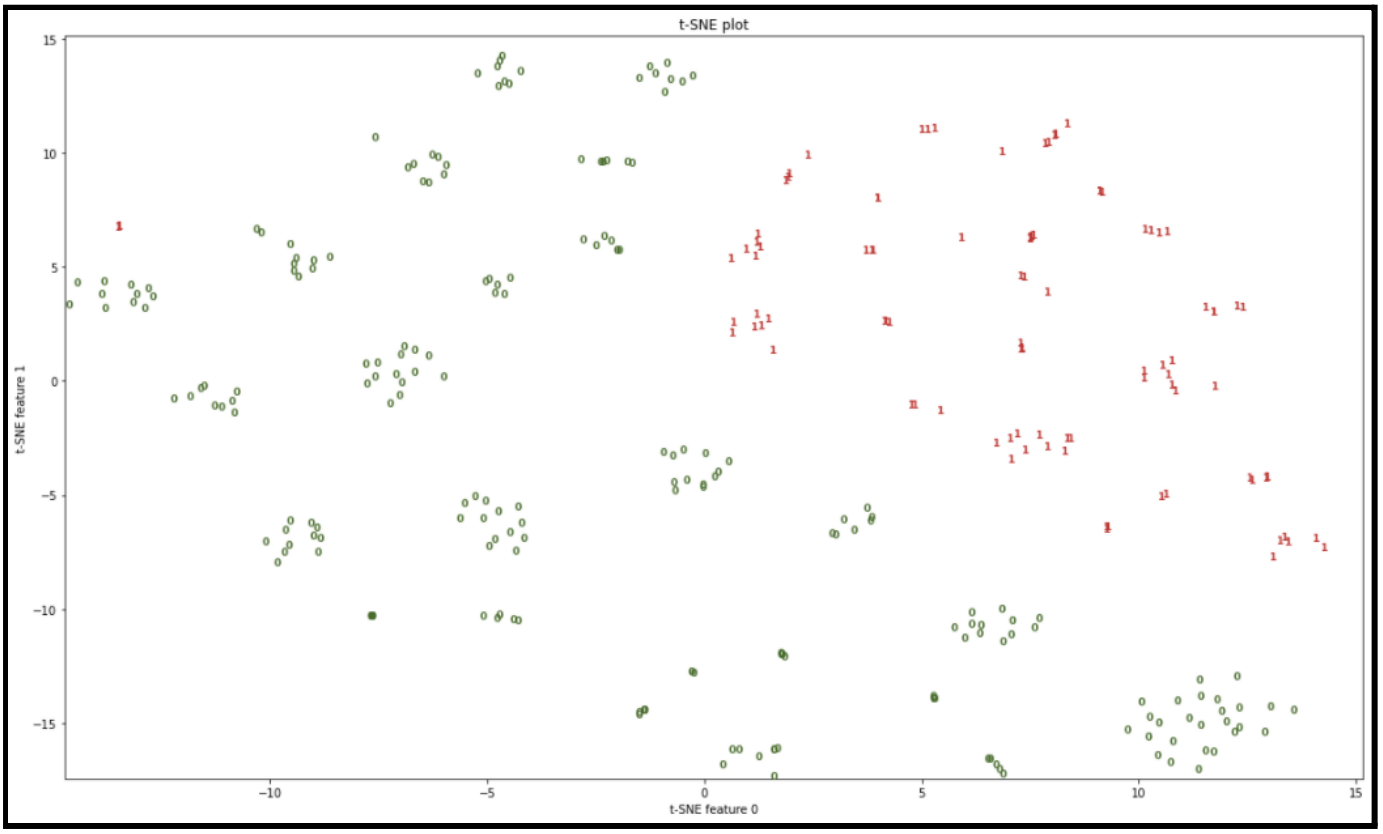


Figure 6:
A table showing what feature pairs SelectKbest chose.

Pair	Number of times tree	Number of times KNN	Number of times SVC	Average
Time & Serum Creatinine	57	46	51	51.3
Time & Ejection Fraction	27	32	28	29
Time & Age	13	17	19	16.3
Time & Serum Sodium	2	3	1	2

Figure 7:
Model accuracies, for all features, best features, and always guessing survival.

	Decision Tree Accuracy	KNN Accuracy	SVC Accuracy
All features	0.829	0.646	0.678
Best features	0.833	0.815	0.716
Guessing survival	0.676	0.678	0.678

Figure 8:
Model accuracies, for all features, best features, and always guessing survival.
With time feature removed

	Decision Tree Accuracy	KNN Accuracy	SVC Accuracy
All features	0.716	0.635	0.672
Best features	0.719	0.702	0.687
Guessing survival	0.672	0.682	0.677

Figure 9:
Binary confusion matrix for the best model [number of patients]

	TP	FP	FN	TN
Decision Tree All features	52	1	4	18
Decision Tree Some features	54	1	4	16
KNN All features	54	3	14	4
KNN Some features	54	1	4	16
SVM All features	60	0	15	0
SVM Some features	53	3	7	12

Hyperparameter tuning

Decision Tree depth

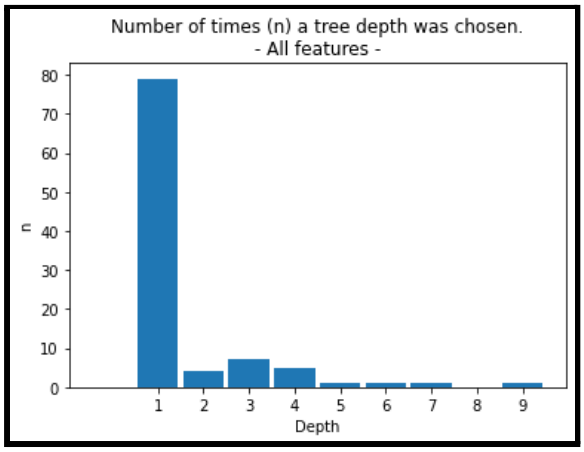


Figure 10.1

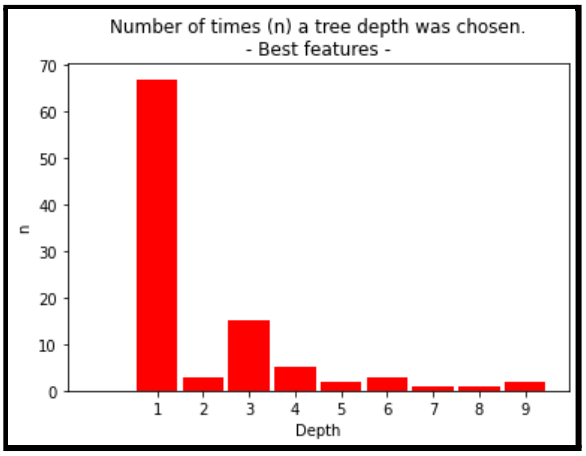


Figure 10.2

K in KNearestNeighbours

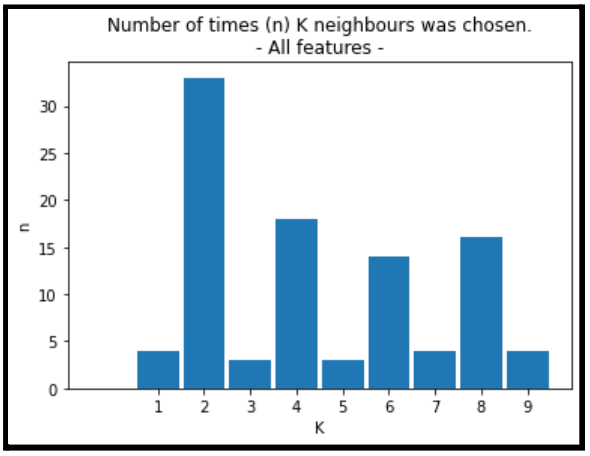


Figure 11.1

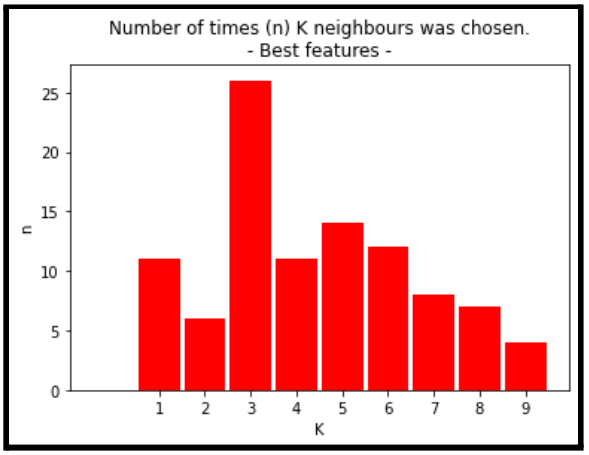


Figure 11.2

C and Gamma in SVC

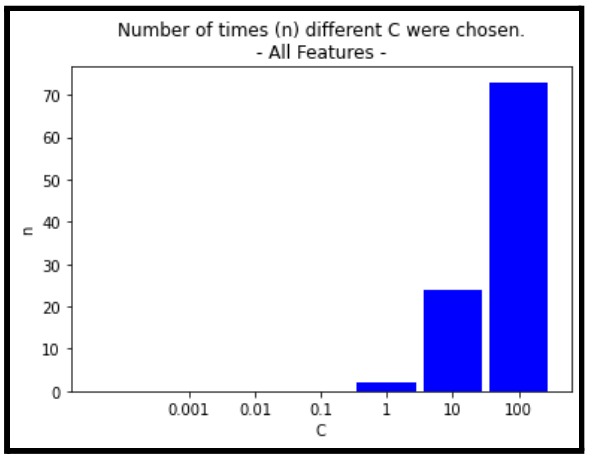


Figure 12.1

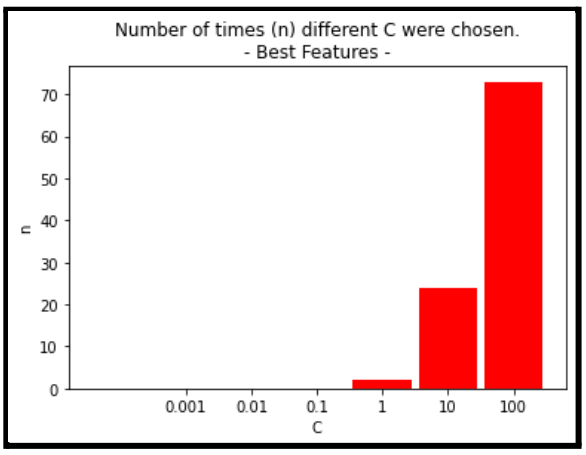


Figure 12.2

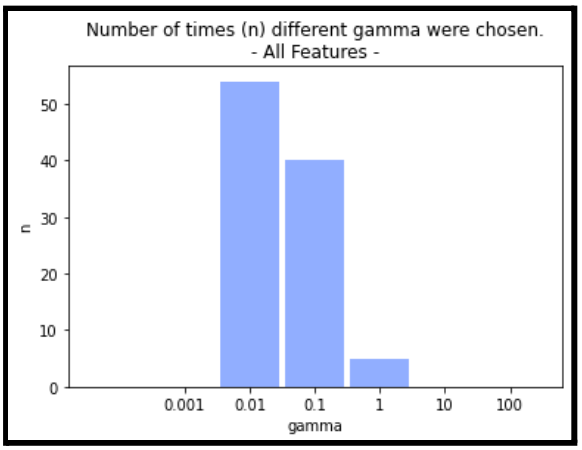


Figure 12.3

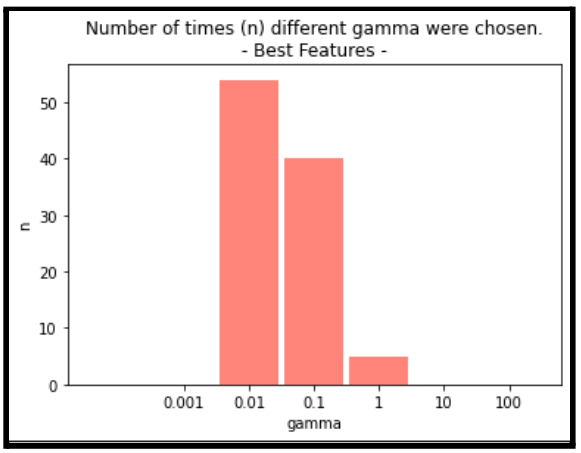


Figure 12.4

Discussion

Analysis of the results

One of the first things created was a correlation matrix (see Figure 1) for all of the features in the dataset. The purpose of this was purely exploratory, we wanted to get a view over what features could be interesting to explore further. The features we found most interesting were ‘Time’, ‘Serum-creatinine’, ‘Ejection-fraction’ since they had a high positive or negative correlation with the death event. In *The loop* however, we chose to use *SelectKBest* to choose our feature pairs.

When viewing the t-sne plot one could observe how there exists some type of connection. One interesting thing shown here is that there are multiple ‘islands’ of data points for both *Death Event = 0* or *1*, suggesting that there are multiple combinations of factors which lead to survival or death. We could also make out clusters in the plots in figure 3 & 4 which indicated that a model could use these few features to differentiate between patient outcomes.

One interesting result when splitting the data and using the SelectKBest method was that we didn't get the same features for every split of the data (See Figure 6). This means that the train/test/validation split does quite heavily affect the model. If we had a larger dataset we would probably get more consistent feature pairs because the big variations for each datapoint would in the grand scheme of things be less relevant for the result. *Time* was almost always chosen as a feature, the natural explanation to this is that *Death Event* is measured within the follow up period, which is the time feature. This is quite problematic because they then get directly linked together without giving any information about any of the other features. In a real-world model the ‘time’ feature would not be included since it isn’t really a medical attribute in the same way the other features are. We tried removing this feature in some runs and as expected we got worse results. (See Figure 8)

The best average accuracy was 0.833 with the decision tree model using only chosen features when predicting. We were quite satisfied with these results since it was clearly better than guessing. This means that we atleast made some sort of improvement using machine-learning which was the goal from the start. In a real world medical setting however this model would probably not be sufficient.

Another interesting result was how fewer features always lead to a better accuracy score. We could see this in the result of the confusion matrices for the different methods used. The best results were acquired from the decision tree model. This is quite interesting because the data we had was both continuous and discrete. This showed us that decision trees truly lend themselves to this type of data.

KNN in particular dropped in accuracy quite drastically when using all features rather than a select few. When only two features were used, the accuracy score came just short of the best

model which was the decision tree classifier. This is probably because of the so-called ‘curse of dimensionality’. When using all 13 features the distance between data points makes the data too sparse for the model to effectively be trained.

The histograms shown in figures 10-12 tell us what fine tuned parameters were most often picked by the model. For decision trees in the majority of cases the best *max_depth* value stayed at 1. In the KNN model however the parameters varied a little between the model with all features and selected features, which probably has something to do with the aforementioned ‘curse of dimensionality’. For SVC, the smallest possible *gamma* was most often used along with the highest possible *C*.

The next step: future work

If we were to continue the work of this project thera are a few things we would be interested in doing. Firstly we would try more models, for the scope of this project we only applied three but there are plenty of others which could be interesting to try out. We are particularly interested in Random Forests. Since the Decision Tree generated the highest accuracy on our data set, it’s not far fetched to guess that a random forest, which uses the same type of structure would perhaps be even better.

Secondly we would like to experiment more thoroughly with the parameter *time*. As discussed earlier it has probably affected our results greatly despite not being the best feature to determine death from heart failure from a medical perspective. We would try to exclude it all together and do more comparisons to the results we got including it.

Next we would also like to evaluate different numbers of selected features, in this project we used 2, but it would be interesting to see how using 1, 3, 4, or more features would affect the result.

Lastly we would like to try our models on larger datasets. 299 data points makes a rather small data set, therefore it would be interesting to see how a bigger one would affect accuracy and what feature pairs are being extracted.

Acknowledgements

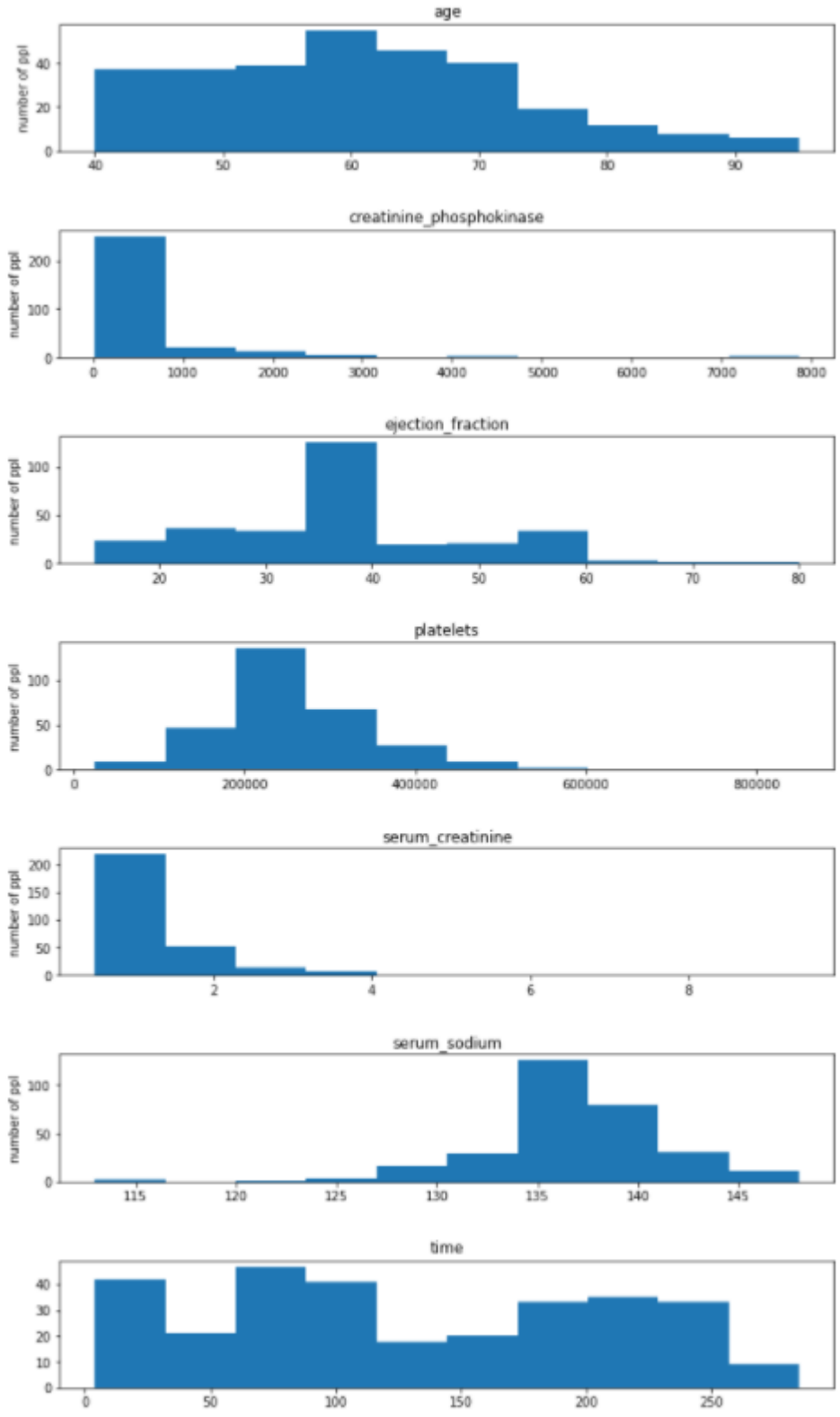
The workload was distributed evenly yet loosely. Any group member was free to work on their own part of the code or edit other members’ parts. The same philosophy went for the report. The only part of the project that was entirely up to the different group members was to write their presentation scripts, yet when rehearsing the presentation we changed things for each script together.

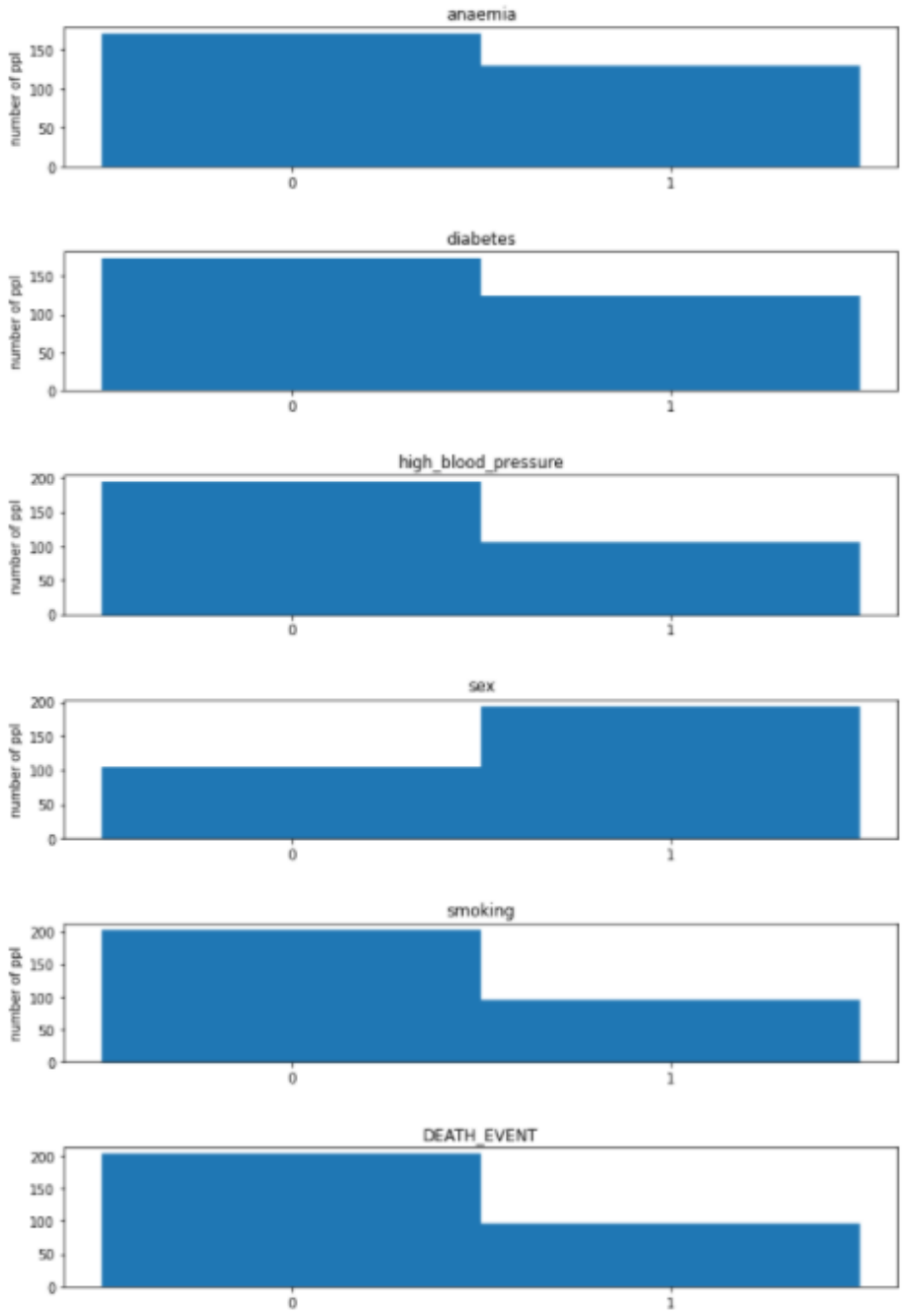
Citation

Anvir Ahmad, Assia Munir, Sajjad Haider Bhatti, Muhammad Aftab, and Muhammad Ali Raza (Government College University, Faisalabad, Pakistan) (2017). Heart failure clinical records data set. UCI Machine Learning Repository Irvine, CA: University of California, School of Information and ComputerScience. https://archive.ics.uci.edu/ml/datasets/Heart+failure+clinical+records (03/03-2022)

Chicco, D., Jurman, G. Machine learning can predict survival of patients with heart failure from serum creatinine and ejection fraction alone. *BMC Med Inform Decis Mak* 20, 16 (2020). https://doi.org/10.1186/s12911-020-1023-5 (03/03-2022)

Appendices


age
number of ppl
0
20
40
40
50
60
70
80
90
creatinine_phosphokinase
number of ppl
0
100
200
0
1000
2000
3000
4000
5000
6000
7000
8000
ejection_fraction
number of ppl
0
50
100
20
30
40
50
60
70
80
platelets
number of ppl
0
50
100
0
200000
400000
600000
800000
serum_creatinine
number of ppl
0
50
100
150
200
2
4
6
8
serum_sodium
number of ppl
0
50
100
115
120
125
130
135
140
145
time
0
10
20
30
40
0
50
100
150
200
250


anaemia
number of ppl
0
50
100
150
0
1
diabetes
number of ppl
0
50
100
150
0
1
high_blood_pressure
number of ppl
0
50
100
150
200
0
1
sex
number of ppl
0
50
100
150
200
0
1
smoking
number of ppl
0
50
100
150
200
0
1
DEATH_EVENT
0
50
100
150
200
0
1