Discussions and Figures

1. Read the article and reproduce the results presented in Table-4 using Python modules and packages (including your own script or customised codes). Write a report summarising the dataset, used ML methods, experiment protocol and results including variations, if any. During reproducing the results:
i) you should use the same set of features used by the authors.
ii) you should use the same classifier with exact parameter values.
iii) you should use the same training/test splitting approach as used by the authors.
iv) you should use the same pre/post processing, if any, used by the authors.
v) you should report the same performance metrics as shown in Table-4.

A:

Dataset Description:

The dataset contains 299 (203 Non-Death Event, 96 Death Event) and samples and 13 length feature-vector for each sample. The data provided denotes the factors which might be responsible in occurrence of a death event due to heart failure. This has 11 clinical features, 1 column for follow-up time and one binary target variable denoting DEATH_EVENT.

The **pre-processing** done after loading the data includes using only the clinical columns for the result described in Table 4 of the article (**Time column is not included** in this analysis and hence dropped from the data frame). Further pre-processing required for training and prediction is described in experimental protocol section.

The data type and name of each column is:

```
age                         float64
anaemia                       int64
creatinine_phosphokinase      int64
diabetes                      int64
ejection_fraction             int64
high_blood_pressure           int64
platelets                   float64
serum_creatinine            float64
serum_sodium                  int64
sex                           int64
smoking                       int64
time                          int64
DEATH_EVENT                   int64
dtype: object
```

Used ML Methods:

The authors have used 10 ML methods

1) Random Forest
2) Decision Trees
3) Gradient Boosting
4) Linear Regression
5) One Rule Classification
6) ANN
7) Naïve Bayes Classifier
8) SVM-Radial
9) SVM-Linear
10) KNN

Experiment Protocol:

The experiment has two different protocols for these set of models:

- Models which required hyper parameter tuning
    - KNN, ANN, SVM (Both): For these set of models, the dataset was divided into three parts, train, validation and test (in ratio 0.6, 0.2, 0.2 respectively)
    - For these models grid search was done for best Matthew's correlation coefficient for every iteration. [As denoted in the git repository too]
        - K fr KNN
        - C for SVM
        - Hidden Layer and Units for ANN
- ***Results section highlight the values obtained and differences from reported.***
- Models with default params:
    - Remaining models formed this set, here the dataset was divided into 2 parts, train and test (0.8 and 0.2).

For both these sets the models were executed 100 times with random data, best hyper param, model training and prediction being updated in each iteration. Finally for the **post-processing** step, the mean scores over 100 iterations across all metrics was reported as consolidated result.

Result Comparison:

	Method	MCC	F1Score	Accuracy	TPR	TNR	PR_AUC	ROC_AUC
0	RF	0.372141	0.534462	0.741833	0.470366	0.872317	0.620215	0.781443
1	DT	0.266613	0.499256	0.677167	0.510155	0.758025	0.585205	0.634090
2	GradBoost	0.356656	0.536310	0.725667	0.495953	0.839351	0.599557	0.750469
3	LinearReg	0.349602	0.496964	0.739167	0.417154	0.894582	0.616947	0.763360
4	OneR	-0.008590	0.308625	0.560333	0.314168	0.677230	0.415025	0.499809
5	ANN	0.000000	0.058665	0.640833	0.120000	0.880000	0.658583	0.500000
6	NB	0.197654	0.319070	0.705333	0.222607	0.935383	0.559977	0.732475
7	SVM_R	0.000000	0.000000	0.674667	0.000000	0.999251	0.350351	0.502056
8	SVM_L	0.369248	0.511947	0.739833	0.435530	0.888236	0.625601	0.775458
9	KNN	-0.000134	0.159860	0.624167	0.123778	0.865860	0.346353	0.487741

(a) *Results from recreating experiment mentioned in article [Mean over 100 runs per model]. Top result across every metric highlighted in blue.*

Table 4 Survival prediction results on all clinical features – mean of 100 executions

Method	MCC	F_1 score	Accuracy	TP rate	TN rate	PR AUC	ROC AUC
Random forests	**+0.384***	0.547	0.740*	0.491	0.864	0.657	0.800*
Decision tree	**+0.376**	0.554*	0.737	0.532*	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	0.750*	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	0.981*	0.594	0.754
k-nearest neighbors	**-0.025**	0.148	0.624	0.121	0.866	0.323	0.493

(b) *Results mentioned in article [Mean over 100 runs per model]. Top result across every metric highlighted in blue*

**** Please note that the results will change for multiple runs as there is no Random seed used. This is done because the authors metrics rely heavily on randomizing the data for every run, so using seed is counter intuitive, Although, the approximate results should be the same across all runs.***

Here, we can see that the overall result reflects the findings of the authors with variation in SVM-radial, where MCC and F1 score are 0, this is because SVM – radial achieves very high (near perfect) TNR and has 0 TPR. The other difference is in MCC for ANN, this can be attributed to different best hyper param in experimental vs reported setup. Other models have their figures very close to the reported figures, with minor differences because of randomization.

This reinforces the observation from the article that Random Forest outperformed every model and SVM had highest TPN. This also reinforces that every model performed poorly on TPR criteria.

Hyper Param comparison:

From the article [Table 4, caption]:

Support Vector Machine with linear kernel. Our hyper-parameter grid search optimization for *k*-Nearest Neighbors selected $k = 3$ on most of the times (10 runs out of 100). Our hyper-parameter grid search optimization for the Support Vector Machine with radial Gaussian kernel selected $C = 10$ on most of the times (56 runs out of 100). Our hyper-parameter grid search optimization for the Support Vector Machine with linear kernel selected $C = 0.1$ on most of the times (50 runs out of 100). Our hyper-parameter grid search optimization for the Artificial Neural Network selected 1 hidden layer and 100 hidden units on most of the times (74 runs out of 100). We report in blue and with * the top performer results for each score.

KNN (K =3), SVM-Radial (C = 10), SVM-Linear (C=0.1), ANN (Hidden Layer = 1, Hidden Units = 100)

When I counted these numbers in my setup I found:

```
KNN K = 3:  9 runs out of 100
SVM C-Radial = 10:  0 runs out of 100
SVM C-Linear = 0.1:  25 runs out of 100
ANN Hidden Layer = 1:  99 runs out of 100
ANN Hidden Units = 100:  0 runs out of 100
```

Whereas the best features based on maximum frequency for best Matthews corrcoef were:

```
Best C for SVM RBF:  0.001 Occurs  98  times out of  100
Best C for SVM linear:  0.001 Occurs  47  times out of  100
Best K for KNN:  5 Occurs  15  times out of  100
Best Hidden units for ANN:  5 Occurs  100  times out of  100
Best Hidden layers for ANN:  1 Occurs  100  times out of  100
```

The differences in hyper parameters can be attributed to the difference due to randomization and difference in implentaion between R and Python algorithms. Moreover, the lack of default parameters result in difference between One Rule Classifer as the basis of classification was not mentioned in the article.

2. Design and develop your own ML solution for this problem. The proposed solution should be different from all approaches mentioned in the provided article. This does not mean that you must have to choose a new ML algorithm. You can develop a novel solution by changing the feature selection approach or parameter optimisations process of used ML methods or using different ML methods or different combinations of them. This means, the proposed system should be substantially different from the methods presented in the article but not limited to only change of ML methods. Compare the result with reported methods in the article. Write a technical report summarising your solution design and outcomes. The report should include:

i) Motivation behind the proposed solution.
ii) How the proposed solution is different from existing ones.
iii) Detail description of the model including all parameters so that any reader can implement your model.
iv) Description of experimental protocol.
v) Evaluation metrics.
vi) Present results using tables and graphs.
vii) Compare and discuss results with respect to existing literatures.
viii) Appropriate references (IEEE numbered).

A:

Motivation and Abstract

This approach proposes creating an ensemble of Random Forest and SVM-Radial classifier to create an Ensemble Voting Classifier. This is coupled with pre-processing techniques to achieve best result on given dataset. The pre-processing can be classified into 3 parts:

1) Performing exploratory data analysis and using full set of features (including follow-up time) instead of only clinical features.
2) Using standard scaler to normalize the data. This helps in reducing the effect of data imbalance as well as the effect of dominating samples causing noise in model training.
3) Using grid search to use best hyper parameters for the base classifiers.

The motivation behind this approach was to maximize the TPR (Recall) as all the models in the reported article performed poorly on this metric. Moreover, the proposed approach utilizes the strength of ensembles by coupling two different family of classifiers.

Difference from existing approach

There are three major differences from existing approach:

1) The existing approach uses naïve approach of selecting best hyper parameter for every run. In the proposed approach the best parameter is chosen once using Grid Search as there is very minimum deviation across the best parameter for different randomization. This results in an efficient solution in terms of execution time.
2) The existing approach does very little in handling the non-convergence of classification techniques due to class imbalance. The proposed approach normalizes the data which yields in higher values across all metrics [Details in result section]
3) Although the existing approach has used ensemble methods (Gradboost and Random Forest), they both come from similar family of ensembles. The proposed approach

couples SVM with another ensemble Random Forest thus enforcing the existing ensemble in area where it was lacking.

Model description and Experiment Protocol

As mentioned earlier there are 3 parts to this approach:

1) Exploratory data analysis:

Two methods were used to achieve this, the data was divided into two parts (Death vs Non-Death Events) and distribution of all the variables were observed. This shows that time feature had good basis of being a able separator of two classes. After this, the correlation between variables was calculated and plotted on the heatmap, this shows that age is an important factor with a high correlation (0.25) with Death Events. The highest was serum creatinine, which is also reported in the article.

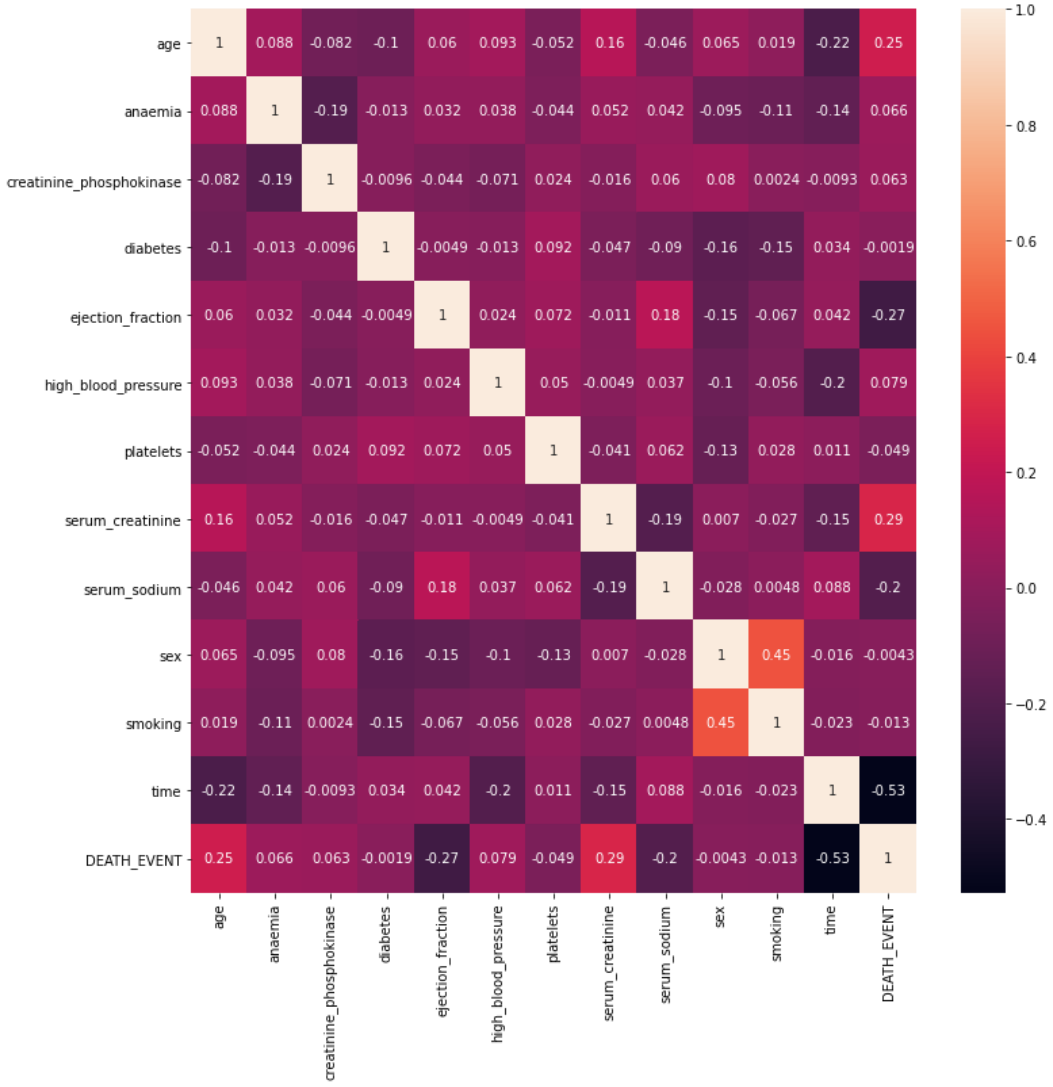

Figure: Correlation between variables

2) **Pre-processing and Grid Search:**
   a) The dataset was randomly divided into two parts: train (0.8) and test (0.2). This was done while preserving class ratio across splits.
   b) For finding best hyper parameters, grid search with 5-fold stratified cross validation was done on the training part. The scoring criteria was recall score as the reported methods performed poorly in this metric.

3) **Model training**
   The best hyper parameters obtained from the grid search step were used to train a Voting ensemble of SVM-radial and Random Forest to boost TPN rate of Random Forest and TPR rate of SVM. The model training was done on fresh splits for every run to mimic method used in the article.

Evaluation Metrics

The evaluation metric for this approach is kept like the one reported in article, i.e., performance metrics across 100 runs were averaged to present a consolidated result. This was done to have a fair comparison ground with the existing approach.

The metrics used are:

1) MCC
2) F1Score
3) Accuracy
4) TPR
5) TNR
6) PR_AUC
7) ROC_AUC

Results:

The results show that the proposed approach outperform every model in all the categories for the given dataset, except for SVM-Radial in TNR which achieves a perfect score for some runs (possible case of skewed model). On the other hand, the Voting Ensemble method coupled with normalizing the data and grid search for optimal hyper parameters boasts of nearly double gain in MCC as compared to the top performing reported model (Random Forest). The gain across other metrics is also supportive of the robustness of the proposed approach.

	Method	MCC	F1Score	Accuracy	TPR	TNR	PR_AUC	ROC_AUC
0	RF	0.362097	0.528201	0.734000	0.478605	0.857024	0.620158	0.780090
1	DT	0.271654	0.494836	0.682000	0.507180	0.764472	0.582695	0.635826
2	GradBoost	0.332048	0.518327	0.720333	0.477428	0.841797	0.581999	0.745975
3	LinearReg	0.338030	0.482978	0.735000	0.399733	0.893393	0.598540	0.761533
4	OneR	0.017236	0.325226	0.570667	0.332281	0.685181	0.439700	0.509025
5	ANN	0.000000	0.123955	0.591000	0.250000	0.750000	0.662333	0.500000
6	NB	0.203230	0.327917	0.700167	0.234979	0.923467	0.539799	0.721906
7	SVM_R	0.000000	0.000000	0.673000	0.000000	1.000000	0.351091	0.502557
8	SVM_L	0.328643	0.484942	0.730500	0.425394	0.868686	0.594871	0.753973
9	KNN	0.011467	0.164399	0.631833	0.126725	0.867891	0.344566	0.497099
10	VotingEnsemble	0.595960	0.722804	0.827667	0.712632	0.880976	0.799319	0.893543

Figure: Highest values in 7 out of 8 metrics

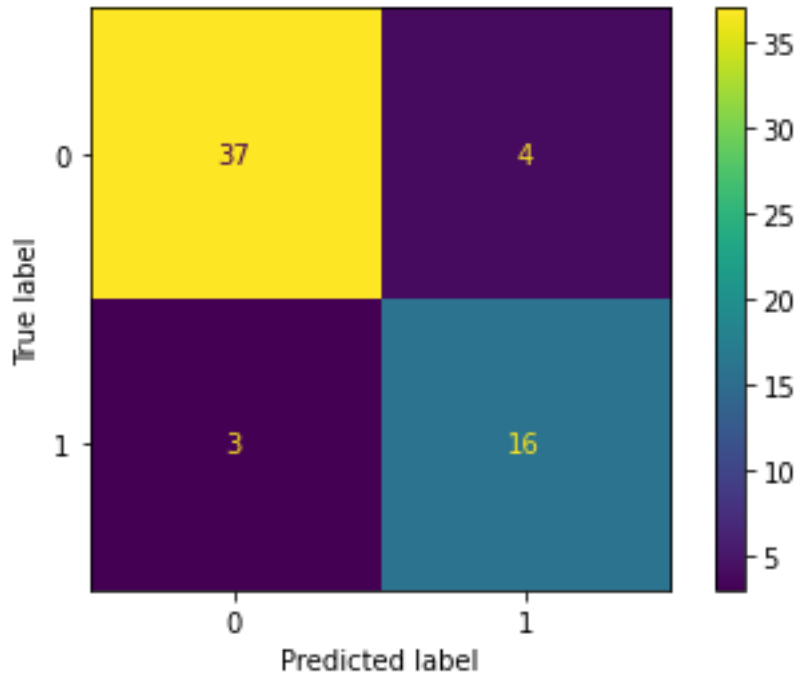

Figure: Confusion Matrix

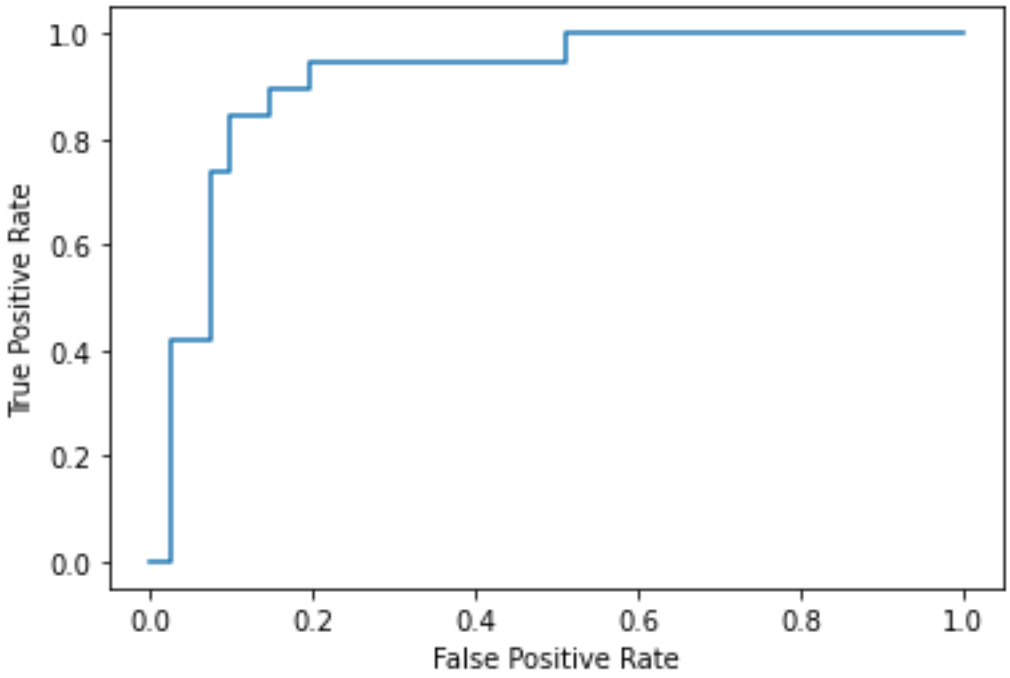

Fig: ROC curve

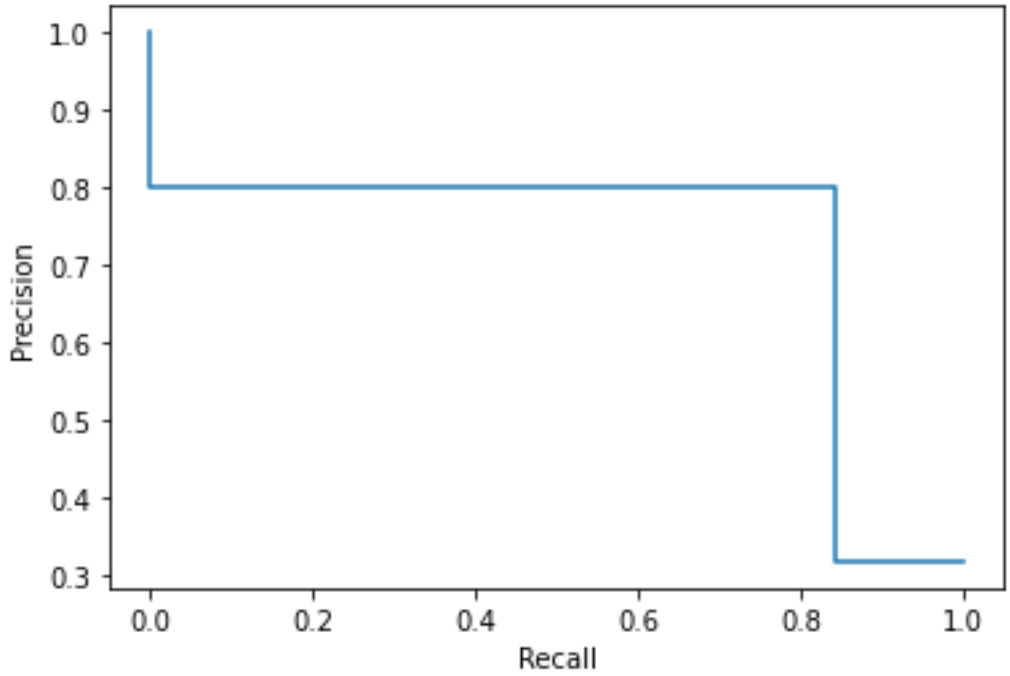

Fig: PR Curve

Other existing works and comparison with proposed method

Ishaq et al. [1] use SMOTE to boost accuracy across all classification techniques used:

Models	Accuracy	Precision	Recall	F-Score
DT	0.8278	0.83	0.83	0.83
AdaBoost	0.8852	0.89	0.89	0.89
LR	0.8360	0.84	0.84	0.85
SGD	0.5409	0.54	0.54	0.53
RF	0.9180	0.92	0.92	0.92
GBM	0.8442	0.84	0.84	0.84
ETC	0.9262	0.93	0.93	0.93
GNB	0.7459	0.75	0.75	0.75
SVM	0.7622	0.76	0.76	0.76

[Performance as reported by [1]

While this approach [1] has comparable performance than the proposed approach across common metrics it is similar in Recall score which is highly indicative of good performance on this data due to the imbalance. Moreover, the approach by Ishaq et al. [1] suffers from another drawback, the dataset is exposed to SMOTE prior to splitting into train and test set. This exposes it to the probability that test set might contain leaked samples from the artificial data, this weakens the high metric values. Whereas in the proposed approach of Voting Ensemble, the pre-processing step (Normalization) model is trained on the training set to avoid the similar situation from happening.

References:

[1] A. Ishaq *et al*., "Improving the Prediction of Heart Failure Patients' Survival Using SMOTE and Effective Data Mining Techniques," in *IEEE Access*, vol. 9, pp. 39707-39716, 2021, doi: 10.1109/ACCESS.2021.3064084