Heart Failure Classification

Shaima Abdulmajeed Alharbi

Table of Contents

Summary or Executive Summary

In this project, I compared the performance of several algorithms in classifying people if they had heart failure or not. The accuracy of the algorithms (Perceptron, decision tree, Random Forest, logistic regression, K nearest neighbours, or Support vector classifier) was compared to solve this problem. LR gave better results than the other classifiers. I can later improve the performance of the two algorithms by adding Layers on its structure, help in obtaining better performance of algorithms.

Introduction

Today, the intervention of technology in various areas of life has become necessary because of the facilities and assistance it provides and the reduction of the human and administrative burden, in addition to the modern advantages that this technology provides in various fields.
Among these fields i find the medical field, where technology has contributed to many different medical and health issues, and has helped provide better treatments and reduce suffering at the same time. Not only that, but technology has been able to provide new machines, medicines, and treatments that have helped protect many people and improve their health. Treatment opportunities.
Given the statistics and increasing numbers of heart patients, in this project, I built a system that detects the presence of heart failure in a patient according to several factors, based on data found on the Kaggle website [1].

Report Body

In this section, I will give an overview of the techniques i used in implementing the project, and i will also explain the complete steps along with the results i reached.

Methods/AI techniques used

In this project, i used six classifiers to classify a person whether he has heart failure or not. The classifiers that ire used are:

1- Decision tree (DT)

   Decision Tree is a supervised machine learning algorithm used for both classification and regression tasks. It creates a tree-like model of decisions and their possible consequences. The algorithm builds the tree by recursively splitting the data based on different features, aiming to maximize the information gain or minimize the impurity at each split. The final tree structure can be used to make predictions on new input data by traversing the tree from the root node to a leaf node, where a decision or prediction is made. Decision Trees are easy to interpret and can handle both numerical and categorical data [2].

2- Random forest (RF)

   Random Forest is an ensemble learning method that combines multiple decision trees to make predictions. It is a combination of bagging and random feature selection. The algorithm creates an ensemble of decision trees, where each tree is trained on a random subset of the original data with replacement (bootstrap sampling), and at each split, a random subset of features is considered. The final prediction is made by aggregating the predictions of individual trees, either through voting (classification) or averaging (regression). Random Forests are known for their ability to handle large datasets, high-dimensional data, and reduce overfitting [3].

3- logistic regression (LR):

Logistic Regression is a statistical algorithm used for binary classification problems. Despite its name, it is a linear model that predicts the probability of an instance belonging to a particular class. The algorithm calculates the log odds of the probability using a linear combination of the input features and applies the logistic function (sigmoid) to map the result between 0 and 1. A threshold is then applied to determine the class label. Logistic Regression is widely used due to its simplicity, interpretability, and efficiency in handling large datasets [4].

4- K nearest neighbours (KNN):

The K-Nearest Neighbour classification algorithm is one of the data mining algorithms that falls under the category of classification and prediction algorithms, and it is one of the most common and widely used algorithms. This algorithm is suitable to be used when there is not enough prior knowledge about the data and its distribution. [4]

This algorithm is based on the idea that adjacent samples are similar (e.g. people who live in the same geographic area usually have a similar average income). The idea that neighbours are similar is dropped on data samples since usually adjacent data samples are similar. [6]

One of the most important features of the KNN algorithm is that it achieves high classification efficiency and accuracy in addition to the lack of need for much prior knowledge about the data and the insensitivity of the algorithm to outliers in addition to its compatibility with noise reduction techniques. [5]

5- Support vector classifier (SVM):

SVM classifier is a set of supervised learning methods used for classification, regression, and outlier detection. SVMs can solve linear and nonlinear problems and they work ill for many practical problems. [4]
The idea of SVM is simple: the algorithm creates a line or hyperplane to separate the data into classes. According to the SVM algorithm, the points closest to the line from each of the categories will be found and these points are called support vectors. Next, the distance between the line and the support vectors must be calculated and this distance is called the margin.
The goal is to maximize margin. The ultra-level for which the margin is the maximum is the optimal ultra-level. Thus, SVM tries to define decision boundaries in such a way that the separation between the two classes is as wide as possible, although SVM is a linear model, it is also used to solve nonlinear classification problems by increasing the number of dimensions. [6]

6- Perceptron:

The machine-based algorithm used for supervised learning of various binary sorting tasks is called Perceptron. Furthermore, Perceptron also plays an essential role as an artificial neuron or neural connector in figuring out some input data computations in business intelligence. The perceptual model is also ranked as one of the best and most specific types of artificial neural networks. Being a supervised learning algorithm for binary classifiers, i can also consider it as a single-layer neural network that contains four main parameters: input values, weights and bias, net sum, and activation function [7].

Implementation of AI techniques

First, I chose the dataset from the Kaggle website [1], as the dataset contains 13 features, as these features help detect the presence of heart failure in the patient. Using this data, i compared the performance of several algorithms in detecting the presence of heart failure according to factors specific to the patient or whether he is a healthy person.
Relying on the Python language, which is the leading language in the field of artificial intelligence and deep learning, i wrote the code for the project on the Anaconda working environment, and used auxiliary libraries and packages (Sklearn, Pandas, matplotlb) to build the Classifier, to display the data and represent it graphically.
i searched for missing elements, and duplicate lines, and pre-processed the data before entering it into the classifiers

```
In [9]: df.isnull().sum()
Out[9]: age                       0
In [5]: df.shape
Out[5]: (299, 13)
In [6]: df = df.drop_duplicates()# remove the doplicate rows
In [7]: df.shape
Out[7]: (299, 13)
        smoking                   0
        time                      0
        DEATH_EVENT               0
        dtype: int64
```

Below is a report of the results for each classifier used in the code:

1) Decision Tree:

```
# Present the overal report of the model's evaluation results using the method classification_report() in metrics
report = metrics.classification_report(y_test, y_pred)
print(report)
```

```
              precision    recall  f1-score   support

           0       0.89      0.77      0.83        44
           1       0.50      0.71      0.59        14

    accuracy                           0.76        58
   macro avg       0.70      0.74      0.71        58
weighted avg       0.80      0.76      0.77        58
```

2) Random Forest:

```
In [18]: # Present the overal report of the model's evaluation results using the method classification_report() in metrics
         report = metrics.classification_report(y_test, y_pred)
         print(report)
```

```
              precision    recall  f1-score   support

           0       0.81      0.87      0.84        39
           1       0.69      0.58      0.63        19

    accuracy                           0.78        58
   macro avg       0.75      0.73      0.73        58
weighted avg       0.77      0.78      0.77        58
```

3) Logistic Regression:

```
In [28]: # Present the overal report of the model's evaluation results using the method classification_rep
         report = metrics.classification_report(y_test, y_pred)
         print(report)
```

```
              precision    recall  f1-score   support

           0       0.83      0.92      0.88        38
           1       0.81      0.65      0.72        20

    accuracy                           0.83        58
   macro avg       0.82      0.79      0.80        58
weighted avg       0.83      0.83      0.82        58
```

4) KNN:

```
In [39]: # Present the overal report of the model's evaluation results using the method classification_
         report = metrics.classification_report(y_test, y_pred)
         print(report)
```

	precision	recall	f1-score	support
0	0.65	0.87	0.74	38
1	0.29	0.10	0.15	20
accuracy			0.60	58
macro avg	0.47	0.48	0.44	58
weighted avg	0.52	0.60	0.54	58

5) Support Vector Machine:

```
In [50]: # Present the overal report of the model's evaluation results using the method cl
         report = metrics.classification_report(y_test, y_pred)
         print(report)
```

	precision	recall	f1-score	support
0	0.66	1.00	0.79	38
1	0.00	0.00	0.00	20
accuracy			0.66	58
macro avg	0.33	0.50	0.40	58
weighted avg	0.43	0.66	0.52	58

6) Perceptron

```
n [61]: # Present the overal report of the model's evaluation results using the method classification_
         report = metrics.classification_report(y_test, y_pred)
         print(report)
```

	precision	recall	f1-score	support
0	0.00	0.00	0.00	38
1	0.34	1.00	0.51	20
accuracy			0.34	58
macro avg	0.17	0.50	0.26	58
weighted avg	0.12	0.34	0.18	58

Comparing the different classifiers, Logistic Regression seems to perform better compared to the other classifiers, as the highest accuracy, Precision, and F1-score ire achieved. While decision tree and random forest show similar performance, the latter achieves higher value for classification accuracy and F1-score.
KNN and SVM suffer from poor performance, while Perceptron shows completely unacceptable performance.

Comparisons of different AI techniques

Our results are shown in the following table:

Classifier	Accuracy	Precision	F1-score	Recall	Score Validation
DT	75.86%	79.94%	77.10%	75.86%	0.5
RF	77.58%	76.95%	77.04%	77.58%	1
LR	82.75%	82.61%	82.23%	82.75%	0.5
KNN	60.34%	52.24%	53.69%	60.34%	0
SVM	65.51%	42.92%	51.68%	65.51%	0
Perceptron	34.48%	11.89%	17.68%	34.43%	1

I will provide an analysis of each classifier separately and then compare them.

1) Decision tree:

Accuracy was achieved at 75.86%, classification accuracy at 79.94%, F1 measurement at 77.10%, and recall at 75.86%. These results show that the decision tree has moderate performance in classification.

Performance improvement techniques such as modifying model parameters or applying feature extraction techniques can be attempted to improve classification accuracy and F1 value.

2) Random forest:

Accuracy was achieved at 77.58%, classification accuracy at 76.95%, F1 measurement at 77.04%, and recall at 77.58%. These results indicate that random forest performs better than decision tree, as higher accuracy and higher values of classification accuracy and F1 measure are achieved.

The number of trees in the random forest can be increased to improve overall performance.

3) Logistic Regression:

Accuracy was 82.75%, classification accuracy was 82.61%, F1 measurement was 82.23%, and recall was 82.75%. These results indicate that Logistic Regression achieves better performance than decision tree and random forest, as higher accuracy and higher values of classification accuracy and F1 measure ire achieved.

The use of a variety of features or optimization of feature extraction techniques can be explored to improve model performance.

4) KNN:
Accuracy was achieved at 60.34%, classification accuracy at 52.24%, F1 measurement at 53.69%, and recall at 60.34%. These results show that KNN suffers from poor performance, as low accuracy and low values of classification accuracy and F1 measure are achieved.
It may be useful to increase the number of neighbors in KNN as the value of the parameter.

5) SVM:
Accuracy was achieved at 65.51%, classification accuracy at 42.92%, F1 measurement at 51.68%, and recall at 65.51%. These results show that SVM also suffers from poor performance, as low accuracy and low values of classification accuracy and F1 measure are achieved.
Using different kernels and modifying parameters in SVM can be explored to improve model performance.

6) Perceptron:
Accuracy was achieved at 34.48%, classification accuracy at 11.89%, F1 measurement at 17.68%, and recall at 34.43%. These results indicate that Perceptron suffers from very poor performance, with very low accuracy and low values for classification accuracy and F1 measurement achieved.
Adding layers, increasing the number of training times, or changing its parameters.

Conclusions and Recommendations

In our results, logistic regression had the best performance among the remaining five classifiers, as it gave an accuracy of 82.27%. In future work, i can develop the work by improving the performance of the remaining algorithms.
I can make recommendations for adding layers to Perceptron to improve its performance or changing factors that affect the accuracy of the classifier

List of References

[1] https://www.kaggle.com/datasets/andrewmvd/heart- failure-clinical-data

[2] What is a Decision Tree | IBM. (n.d.). https://www.ibm.com/topics/decision-trees

[3] What is Random Forest? | IBM. (n.d.). https://www.ibm.com/topics/random-forest

[4] Kwakye, K., & Dadzie, E. (2021). Machine learning-based classification algorithms for the prediction of coronary heart diseases. arXiv preprint arXiv:2112.01503.

[5] Srivastava, K., & Choubey, D. K. (2020). Heart disease prediction using machine learning and data mining. International Journal of Recent Technology and Engineering, 9(1), 212-219.

[6] Meshref, H. (2019). Cardiovascular disease diagnosis: A machine learning interpretation approach. International Journal of Advanced Computer Science and Applications, 10(12).

[7] Banoula, M. (2023) 'What is Perceptron: A Beginners Guide for Perceptron,' Simplilearn.com

[Preprint]. https://www.simplilearn.com/tutorials/deep-learning-tutorial/perceptron.

Individual reflection

In this research, i benefited from studying the performance of several algorithms, as i decided to use them to study their performance in detecting heart failure based on several factors so that the project could be developed, by linking the Model Learning Machine with the best performance with a user interface, so that it would be easy to deal with our system.