Prediction of heart disease using machine learning algorithms

Martin Arsovski (63160359)
Faculty of Computer and Information Science
ma6068@student.uni-lj.si

Abstract—Today we live in a modern time where we can say with certainty that technology has advanced greatly in recent years. One of the positive aspects of technological advancement is that it can help medicine detect various human diseases. Heart disease is one of the most serious diseases that currently exists and can be fatal in a person's life. Although the heart is considered to be one of the most resilient organs, it is certainly exposed to damage. Sometimes a lot of examinations are needed to detect diseases. Our goal in the seminar work was to create different models that would predict whether the patient has heart disease or not, based on previous examinations.

Index Terms—Model, K-fold Cross-Validation, Train dataset, Test dataset

1 Introduction

Heart disease can vary in nature and severity. They can be transient or chronic, gradual or sudden, painful or deadly. Some forms of heart disease, closely related to eating and other habits, can be prevented. Others are the result of genetic inheritance, infections, or other factors beyond our control. On average two out of five Americans will die of heart disease. For these reasons, 2.500 people die every day. According to some statistics, heart disease was the leading cause of death in the United States in 2017[2], which is not so far away. Symptoms can include a feeling of tightness and pain in the chest, rapid heartbeat, shortness of breath, dizziness, etc. A much worse case of this disease is when the person has no symptoms[1]. The number of cases is gradually declining thanks to better medical care and more modern technology.

Can machine learning replace the doctor? That question is often asked and is still somehow open. Most opinions are that doctors know something that machines do not, but could not explain. However, machine learning can certainly help diagnose some diseases. Although the diagnosis is not always 100 percent accurate, it could affect the speed of examinations and which cases have been prioritized in hospitals. This can be very important, because with just a few examinations of the patients, they would predict if he has heart disease and the medics could respond faster.

In our seminar assignment we used the Cleveland heart disease dataset. The dataset consisted of data from various studies performed on patients, for whom it was later known whether they had heart disease or not. We implemented five algorithms (Decision tree, Random Forest, Support Vector Machine, K-nearest neighbors and Neural Networks) and compared our results from different metrics with the results described in the article "A Hybrid Intelligent System Framework for the Prediction of Heart Disease Using Machine Learning Algorithms". Our target variable is the binary classifier (Yes / No) which indicates whether a patient has heart disease or not.

2 Related work

The "A Hybrid Intelligent System Framework for the Prediction of Heart Disease Using Ma-

chine Learning Algorithms"[5] article describes how, using the Cleveland heart disease dataset, several models have been developed to diagnose patients heart disease. The authors selected only six most important attributes using different algorithms (Relief, mRMR and LASSO FS algorithm), which they consider to have the greatest contribution to model learning. They have implemented many algorithms such as Logistic regression, K-nearest neighbor, Artificial Neural Network, SVM, Naive Bayes, Decision tree, Random Forest and tested them using K-path cross-validation and various parameters. They have tested the models on all the attributes and on the six different attributes that are considered the most significant by the algorithms.

In our seminar assignment we choose and implemented some of those algorithms and tested them on all and on the selected (by the authors) attributes. Finally, we compared our accuracy with theirs.

3 Data and data processing

Our model was trained on the Cleveland heart disease dataset[3]. The database consists of 303 patients, 13 attributes and one target attribute. Out of all patients, 297 had value for all attributes, so we removed the other 6 and did not take them into account. The attributes were as follows:

- Age (30 ¡ age ¡ 77)
- Sex (1 = Male, 0 = Female)
- Type of chest pain (1 = atypical angina, 2 = typical angina, 3 = asymptomatic, 4 = nonanginal pain)
- Resting blood pressure (94–200)
- Serum cholesterol (120–564)
- Fasting blood sugar ¿ 120 mg/dl (1 = Yes, 0 = No)
- Resting electrocardiographic results (0 = normal, 1 = having ST-T, 2 = hypertrophy)
- Maximum heart rate achieved (71–202)
- Exercise-induced angina (1 = Yes, 0 = No)
- Old peak = ST depression induced by exercise relative to rest (0–6.2)
- Slope of the peak exercise ST segment (1 = up sloping, 2 = flat, 3 = down sloping)
- Number of major vessels (0–3) colored by fluoroscopy (0, 1, 2, 3)
- Thallium scan (3 = normal, 6 = fixed defect, 7 = reversible defect)
- Target (0 = No, 1,2,3,4 = Yes)

Before we could start learning any of the models, we had to process our data. With a short research we saw that the data could be the easiest to present and use, if we write them in a table with the package "pandas". In order for our model to be learned more effectively, it was necessary to represent all the attributes, which have only a few options as a value, with multiple columns, where we will indicate what value that attribute has for a particular patient. Therefore, with the help of the "dummy" function of all data, we determined a new, numeric value. So we added as many columns as the different possible values the attribute had. Its value was denoted by 1 if it is of this type or by 0 if it is not.

Example: chest pain type may have the following values:

- typical angina
- atypical angina
- non-anginal pain
- asymptomatic

We have four possible options for this attribute, so we add as many columns. If one of the patients has typical angina, for example, that column is marked with 1 and the others with 0. The same goes for the other possible options.

Our target attribute also had multiple values. The article said that if its values are 1,2,3 or 4 then it is a diagnosed disease, and if it is a value of 0 it is not. This is how we adjusted our attribute (value 0 if its value is 0 and value 1 if its initial value is 1/2/3/4).

Finally, we normalized the data with "MinMaxScaler", so that each of the attributes had a value between 0 and 1. With this we have greater success in learning the models (for example, this greatly affects the calculation of the nearest neighbor in the KNN algorithm). That way all of the attributes have equal weight.

4 Train and test data sets

Since we only had 297 examples we decided to use K-fold Cross-Validation[6]. First we randomly shuffled the data and divided it K (in out case K=5) times into a training and test sets. In each of the K examples the training set was composed of 80% of the data, and the test set of the remaining 20%.

5 Machine learning algorithms

In our seminar work we used the following algorithms:

- Decision tree
- Random Forest
- Support Vector Machine (SVM)
- K-nearest neighbors (KNN)
- Neural Networks

5.1 Decision tree

The decision tree[4] is one of the most well-known algorithms used in machine learning. The algorithm belongs to the supervised learning algorithms. It is used in operational research to help identify a strategy that is likely to achieve a goal, but it is also a popular tool in machine learning. The model is in the form of a tree, where each of its nodes is a test, made on one of the attributes by which we learn our model (eg. does the patient have difficulty breathing?). The branches represent the result of the test that was performed (eg. answer Yes / No to the question whether the patient has shortness of breath). The target class is found in the leaves of the tree. In our case we have a binary tree where in the leaves we have a discrete target class (0 if the patient does not have heart disease and 1 if he does). The algorithm is quite simple to understand and gives the same results even if the attribute values are not normalized. It has been said that decision trees are relatively successful with a large amount of data. If a decision tree is made of a large number of attributes, its visualization can be a bit confusing, due to the large number of nodes and branches. Excessive adjustment of the tree with the data can be prevented by cutting certain branches of the tree or determining the maximum number of leaves. In our seminar assignment, in order not to get overfitting we limited the trees to a have a maximum 10 leaves.

One of the biggest weaknesses of decision trees is that small changes in the data greatly affect the model (big change in the appearance of the tree, its nodes and branches). The decision tree for our data (with all attributes and normalized data) is presented in the image below (Fig 1).

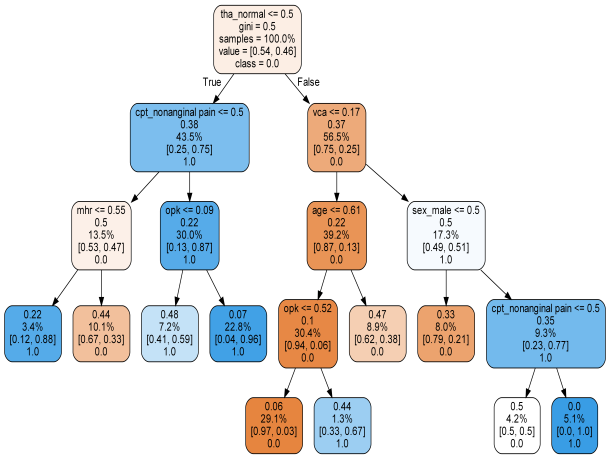

Fig. 1: Decision tree (all attributes, normalized data)

5.2 Random Forest

Random forest[9] is an ensemble learning method for classification, regression, and other tasks that work by constructing multiple decision trees during training. This algorithm is used to improve decision trees. For classification tasks, the random forest output is the class selected by most trees. This is a kind of voting, where each tree determines in which class a certain test sample belongs. In the end, the test sample gets the class that has the most votes. In regression, the final value is equal to the average value of the values of all the trees individually. Because this algorithm is also based on decision trees, its results do not depend on whether the data is normalized or not. Usually this algorithm gives better results than decision trees. In order not to have too much overfitting in our seminar assignment, we set a maximum depth of 4 levels for the trees. One of the

decision trees in the random forest is presented in the image down below (Fig. 2).

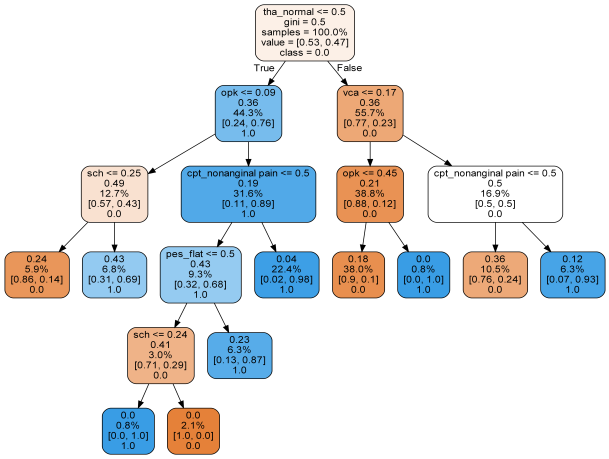

Fig. 2: Decision tree in random forest (all attributes, normalized data)

5.3 Support Vector Machine (SVM)

Support Vector Machine (SVM)[10] is a supervised machine learning algorithm that can be used for both classification or regression challenges (more commonly used in classification problems). In the SVM algorithm, we plot each data item as a point in n-dimensional space, where n is the number of features we have. The value of each attribute is the value of a specific coordinate. Finally, we classify by finding the hyper-plane that distinguishes the two classes very well. The best separation is achieved by the hyperplane that has the largest distance to the nearest training-data point of any class (that is called functional margin). That is because, in general, the larger the margin is, the lower generalization error of the classifier we have. The advantage of the SVM algorithm is that it works well with unstructured and semi-structured data such as text, images and trees. It scales relatively well to high dimensional data. Of course this algorithm has its weaknesses. Sometimes training in large data sets can take a long time and it is also difficult to understand and interpret the final model, variable weights, and individual impact.

5.4 K-nearest neighbors (KNN)

The KNN algorithm[7] is also one of the most famous in the world of machine learning. It can be used for classification and regression. In both cases, the new building is dependent on its closest K neighbors. The distance between neighbors depends on the differences in values between the attributes. The algorithm can also be adjusted by assigning weight to neighbors, so that closer neighbors will have more influence in determining class (in classification) or value (in regression). Because attributes can have quite different numeric values, normalization significantly improves the success of this algorithm and we can say that it is necessary. In the classification, the object is classified by a plurality of votes of its neighbors (the new object has the same class as most of its K neighbors). In regression, the final value is equal to the average of the values of its K neighbors. Because the calculation of distances between a new object and its neighbors can take quite a long time (if we have a lot of data), different mechanisms are used to accelerate and more easily represent the points in space (eg. ctree). The algorithm is quite sensitive to data in which some of the values of the attributes are not specified. It is also almost always necessary to test it on different values of the parameter K, in order to increase the accuracy of the algorithm.

In our seminar assignment for parameter K we set a value of 20.

5.5 Neural Networks

Neural Networks[8] are based on a collection of connected units or nodes called artificial neurons. Each neuron receives a signal, which then processes it. The ”signal” of the connection is a real number. The output of each neuron is calculated by some nonlinear function of the sum of its inputs. Ties are called edges. They, along with neurons, can have a weight that adjusts to learning. The weight increases or decreases the signal strength when connected. Neurons can have a threshold such that a signal is sent only if the aggregate signal crosses that threshold. Most neurons cluster in layers. Different layers can perform different transformations on their

inputs. The signals travel from the first layer (input layer) to the last layer (output layer), where we get our result.

6 METRICS

After training our models for each of the algorithms and getting our predicted target variables for each of the patients, we calculated different metrics for the data obtained:

- accuracy
- standard division
- brier measure
- sensitivity
- specificity
- auc score

We also plotted the ROC curve for each of the algorithms.

6.1 Accuracy

Accuracy shows us how many of our examples we have correctly classified in relation to all examples (expressed in percentages).

6.2 Standard division

A standard deviation is a simple measure of variability or dispersion in a data set. A small standard deviation indicates that all elements of the data set are very close to the mean. A large standard deviation indicates that the group is wide, with a wide range of values.

6.3 Brier score

In our seminar assignment with the help of the function "$predict\_proba$" we obtained percentage values for the probability that the patient has heart disease. With the help of these values we calculated a Brier score.

Brier Score is a strictly appropriate scoring function that measures the accuracy of probabilistic predictions. By this measure we somehow know with what certainty our model has classified the data.

6.4 Sensitivity and specificity

Sensitivity and specificity are calculated according to the following equations:

$$Sensitivity = TP/(TP + FN) \quad (1)$$

$$Specificity = TN/(TN + FP) \quad (2)$$

Sensitivity - the ability of a test to correctly identify patients with a heart disease.

Specificity - the ability of a test to correctly identify people without the heart disease.

True positive (TP) - the person has the disease and the test is positive.

True negative (TN) - the person does not have the disease and the test is negative.

False positive (FP) - the person does not have the disease and the test is positive.

False negative (FN) - the person has the disease and the test is negative.

6.5 ROC curve and AUC score

ROC curve is a performance measurement for the classification problems at some threshold settings. ROC is a probability curve. AUC represents the degree or measure of separability. Higher the AUC, the better the model is at predicting "0" classes as "0" and "1" classes as "1" (the model is better at distinguishing between patients with and without heart disease).

7 RESULTS

7.1 All attributes

Because we used K-fold Cross-Validation, we calculated the average value for each of the metrics.

The table below (TABLE 1) shows the results of the calculated metrics for each of the machine learning algorithms (Decision tree = DT, Random Forest = RF, Support vector machines = SVM, K-nearest neighbors = KNN, Neural Networks = NN).

In our case of all metrics the most important are sensitivity and specificity, because they show us how many patients we have correctly detected that they have / do not have heart disease. We see that we have obtained the greatest sensitivity by using Neural Networks

/	DT	RF	SVM	KNN	NN
accuracy	0.75	0.835	0.818	0.822	0.842
standard division	0.056	0.048	0.05	0.05	0.047
brier measure	0.205	0.128	0.125	0.129	0.125
sensitivity	0.723	0.79	0.773	0.787	0.811
specificity	0.784	0.883	0.865	0.859	0.878
auc score	0.794	0.912	0.911	0.902	0.907

TABLE 1: Results obtained on all attributes.

(81.1%), while we have the greatest specificity with Random Forest (88.3%).Here we can give a slight advantage to the Neural Networks because the difference in specificity is not very large (about 0.5%), while the sensitivity differs by about 2%. If we always have predicted that our example belongs to the majority class (patients in whom no heart disease was found) we would have 53.9% accuracy (out of 297 patients, 160 did not have heart disease). This means that all our models have learned quite well (the lowest sensitivity is in the decision tree and is 72.3%). However, because the highest sensitivity is 81.1% (using Neural Networks), we can not completely say that these models would be a quality replacement for doctors. The authors of the article have obtained the best sensitivity with the Random Forest (94%) but the specificity there is only 70%.

The picture Fig.3 shows the ROC curves for each of the algorithms in one of the K iterations with split data.

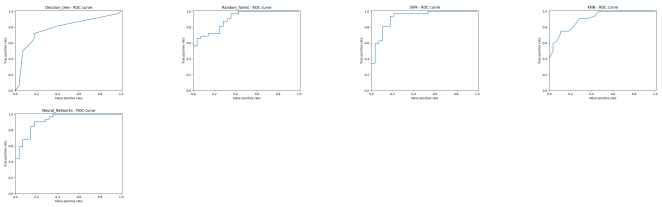

Fig. 3: Roc curve (all attributes)

7.2 Selected attributes

Just like when calculating metrics for all attributes, here we used an average value for all values obtained during K-fold Cross-Validation. We used the same attributes that were selected with:

- Relief algorithm
- mRMR algorithm
- LASSO FS algorithm

in the article.

7.2.1 Selected attributes with Relief algorithm

With the Relief algorithm the following attributes were selected as the most important:

- Thallium scan
- Exercise-induced angina
- Type of chest pain
- Slope of the peak exercise ST segment
- Number of major vessels (0–3) colored by fluoroscopy
- Maximum heart rate

The table below (TABLE 2) shows the results of the calculated metrics for each of the machine learning algorithms (Decision tree = DT, Random Forest = RF, Support vector machines = SVM, K-nearest neighbors = KNN, Neural Networks = NN).

/	DT	RF	SVM	KNN	NN
accuracy	0.798	0.818	0.828	0.838	0.811
standard division	0.052	0.05	0.048	0.047	0.051
brier measure	0.183	0.129	0.125	0.13	0.126
sensitivity	0.704	0.789	0.788	0.771	0.795
specificity	0.882	0.852	0.87	0.9	0.834
auc score	0.829	0.9	0.907	0.888	0.903

TABLE 2: Results obtained on attributes selected by Relief algorithm.

Here we can see that we have a slightly lower sensitivity than when using all the attributes. Again the best result we have obtained is when using Neural Networks (79.5%). The specificity is better than before. Using the KNN algorithm the specificity is 90%. The authors of the article have received 100% sensitivity with the Artificial Neural Network, but in that example the specificity is only 2%, which means that for many examples they have been diagnosed with heart disease, but in fact they have not. This can not be said to be a good result. They have obtained one of the better results with SVM (81% sensitivity and 82% specificity).

The picture Fig.4 shows the ROC curves for each of the algorithms in one of the K iterations with split data.

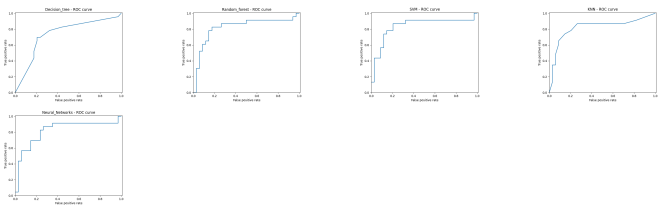

Fig. 4: Roc curve (attributes selected by Relief algorithm)

7.2.2 Selected attributes with mRMR algorithm

With the mRMR algorithm the following attributes were selected as the most important:

- Type chest pain
- Serum cholesterol
- Slope of the peak exercise ST segment
- Number of major vessels (0–3) colored by fluoroscopy
- Sex
- Thallium scan

The table below (TABLE 3) shows the results of the calculated metrics for each of the machine learning algorithms (Decision tree = DT, Random Forest = RF, Support vector machines = SVM, K-nearest neighbors = KNN, Neural Networks = NN).

/	DT	RF	SVM	KNN	NN
accuracy	0.815	0.831	0.824	0.838	0.804
standard division	0.05	0.048	0.049	0.048	0.051
brier measure	0.184	0.128	0.124	0.129	0.127
sensitivity	0.763	0.806	0.813	0.828	0.791
specificity	0.864	0.857	0.84	0.851	0.821
auc score	0.813	0.908	0.904	0.89	0.903

TABLE 3: Results obtained on attributes selected by mRMR algorithm

By selecting only these 6 attributes we obtained better sensitivity with the KNN algorithm than by using all 13. This is also the best sensitivity we had received in the seminar assignment. With this algorithm we have a sensitivity of 82.8% and a very good specificity of 85.1%. We have obtained slightly weaker results with SVM and Random Forest. The authors of the article with Naive Bayes have received 77% sensitivity and 90% specificity.

The picture Fig.5 shows the ROC curves for each of the algorithms in one of the K iterations with split data.

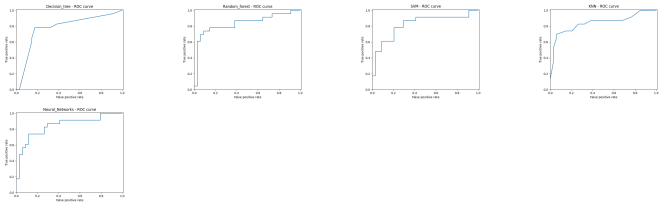

Fig. 5: Roc curve (attributes selected by mRMR algorithm)

7.2.3 Selected attributes with LASSO FS algorithm

With the LASSO FS algorithm the following attributes were selected as the most important:

- Sex
- Number of major vessels (0–3) colored by fluoroscopy
- Exercise-induced angina
- Type chest pain
- Slope of the peak exercise ST segment
- Thallium scan

The table below (TABLE 4) shows the results of the calculated metrics for each of the machine learning algorithms (Decision tree = DT, Random Forest = RF, Support vector machines = SVM, K-nearest neighbors = KNN, Neural Networks = NN).

/	DT	RF	SVM	KNN	NN
accuracy	0.815	0.832	0.818	0.835	0.821
standard division	0.05	0.048	0.05	0.048	0.05
brier measure	0.179	0.127	0.125	0.129	0.123
sensitivity	0.737	0.795	0.781	0.793	0.809
specificity	0.887	0.868	0.859	0.877	0.841
auc score	0.839	0.911	0.909	0.897	0.91

TABLE 4: Results obtained on attributes selected by LASSO FS algorithm.

With this selection of attributes we got the best sensitivity with the Neural Networks(80.9%). The specificity here is 84.1%. The Random Forest algorithm with 79.5% sensitivity and 86.8% specificity and SVM with 78.1% sensitivity and 85.9% specificity are also quite good. The authors of the article have obtained

the best results in Logistic regression with 76% sensitivity and 97% specificity.

The picture Fig.6 shows the ROC curves for each of the algorithms in one of the K iterations with split data.

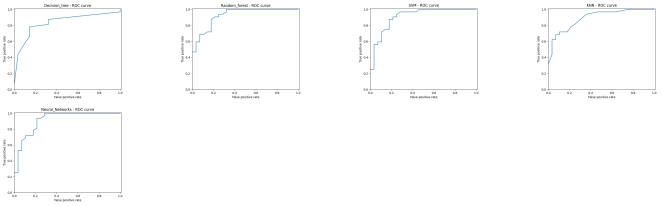

Fig. 6: Roc curve (attributes selected by LASSO FS algorithm)

8 Conclusion

In general we can say that in each selection of attributes we got results different from those of the authors (we have slightly better specificity and lower sensitivity). However, all our models are well trained because they have much better results than those if we always classify the example in the majority class. Nevertheless, it is difficult to say that any of the models would be used in reality because the best sensitivity we got was 82.8%. It is, however, perhaps too weak because more than 17% of patients would get results that they do not have the disease, but in fact have.

Although the Neural Networks had the best sensitivity in three of the four cases, we can say that Random Forest, SVM, and even KNN did not express themselves as bad and had slightly poorer results.

To improve these results, models with multiple attributes and a larger database can be tested. This of course requires a lot of examinations and patients who would agree to use their data.

References

[1] Cardiovascular disease. https://en.wikipedia.org/wiki/Cardiovascular$_{disease}$.
[2] Cdc: Heart disease, cancer leading causes of death in 2017. https://www.healio.com/news/cardiology/20181129/cdc-heart-disease-cancer-leading-causes-of-death-in-2017.
[3] Cleveland dataset. https://archive.ics.uci.edu/ml/datasets/heart+disease.
[4] Decision tree. https://en.wikipedia.org/wiki/Decision$_{tree}$.
[5] A hybrid intelligent system framework for the prediction of heart disease using machine learning algorithms. https://www.hindawi.com/journals/misy/2018/3860146/.
[6] K fold coross validation. https://en.wikipedia.org/wiki/Cross-validation$_{(statistics)}$.
[7] Knn. https://en.wikipedia.org/wiki/K-nearest$_{neighbors}$$_{algorithm}$.
[8] Neural networks. https://en.wikipedia.org/wiki/Neural$_{network}$.
[9] Random forest. https://en.wikipedia.org/wiki/Random$_{forest}$.
[10] Svm. https://en.wikipedia.org/wiki/Support-vector$_{machine}$.