Machine Learning

Heart Disease Prediction

Nurbek Alibekov, Nurzhan Bekzhanov, and Nursaule Erkinkyzy

School of Engineering and Digital Sciences

Nazarbayev University

December 4, 2021

Abstract

Cardiovascular disease is one of the main causes of people's death all over the world. Several factors can be risky for a heart attack such as age over 65, high blood pressure, etc. Machine Learning can help to predict a heart attack according to the data of patients' history and risk factors. There are different researches provided on this topic. In this project we observed heart disease data and implemented different models to find out the most accurate one to make the future prediction. During the pre-processing we divided features into categorical and continuous. As continuous functions needed scaling we used MinMaxScale and StandardScaler. Then applied different models of machine learning as K-nearest neighbors (KNN), Support Vector Machines (SVM), Logistic regression and Neural Networks (NN) to compare. We identified that SVM and Logistic regression with 10-fold cross validation show the highest accuracy with equal value of 99%. We conclude that Neural Networks work better for bigger dataset and that SVM, Logistic regression for our data result in high accuracy with CV.

Keywords
Heart disease prediction, deep learning, neural network, hyperparameter

1 Introduction

Cardiovascular disease (CVD) is the prime cause of death in the world. By the estimation of the World Health Organisation (WHO) each year 17.9 million people die because of this [1]. CVDs include heart and vascular diseases such as coronary heart disease, cerebrovascular disease, rheumatic heart disease and others. Heart attack and strokes account for more than four-fifth of deaths from CVD and one-third of these deaths occur in people who are at age about 70. According to the report in 2018 Kazakhstan took the highest place in the rate of CVD mortality among the EU, Central Asia and Europe. In order to prevent heart diseases there are some ways such as keeping fit, eating healthy food, etc. Consequently, predicting and preventing heart disease is important and can be done using machine learning algorithms.
In this paper we implement 4 types of machine learning model. Their parameters were changed many times in order to find an optimal and to avoid overfitting. Our aim is to choose the best model that precisely predicts whether a person has heart disease or not. In other words, to obtain very high accuracy of the models. For this we compared our results with one of the previously done research. In their paper named "Using machine learning for heart disease prediction." machine learning professors Salhi and Tari found that Neural Networks achieve the highest accuracy of 93% [2]. In this paper we will discuss implementation of our methods, obtained results and conclude which model's accuracy exceeds given in the literature.

2 Methods

Heart Disease Dataset
In real life data is not as accurate as given here. So our data is almost pre-processed and without any missing data [3]. We can see this from Figure 1 below. The data is divided into 13 different variables such as age, sex, cp (chest pain) and other factors with 303 lines of observations

which contribute to heart disease.

```
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       303 non-null    int64
 1   sex       303 non-null    int64
 2   cp        303 non-null    int64
 3   trestbps  303 non-null    int64
 4   chol      303 non-null    int64
 5   fbs       303 non-null    int64
 6   restecg   303 non-null    int64
 7   thalach   303 non-null    int64
 8   exang     303 non-null    int64
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    int64
 11  ca        303 non-null    int64
 12  thal      303 non-null    int64
 13  target    303 non-null    int64
```

Figure 1: Data without missing value

Now what is left for us is to normalise using scaling and dummies. For the features 'sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'ca', 'thal' information is given in categorical way, meaning that they can be divided into columns consisting binary numbers. We converted them from one column to two. If for male it is 1 we add 0 for the second column - female. Thus to convert categorical data into dummies get_dummies() function is used. There are other features that are continuous 'age', 'trestbps', 'chol', 'thalach', 'oldpeak'. Some of them are not normally distributed as you can see in Figure 2, that is why we have to scale them using diffrent methods. For all models we used MinMaxScale except simple SVM without CV which works better with Standard Scaler. These methods were best to have a precise prediction for them. MinMaxScale scales and transforms all data that are not normalized to the range that is given. Our range is from -1 to 1. StandardScaler standardizes data by changing the mean of values and scales to unit variance.

At each algorithm after downloading, 80% of data is taken to train, another 20% to test. And the train data is later divided to validation data while using CV. Next we turned to choose the best learning machine among KNN, SVM, Logistic regression and Neural Networks. Each of them has their own hyperparameters whose variety affects the accuracy. For the KNN we change the K-values which are the number of neighbors varying from 1 to 10 in our case. For others we implemented a cross validation (CV), because simple KNN shows better results without CV. There is a standard of 10-fold CV, so we kept it

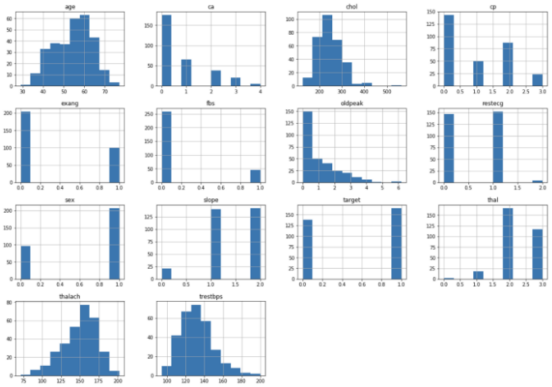

Figure 2: Data set description

in this way for all methods. Also, for the SVM algorithm in both cases the values taken for constraints and degree were the same (1st degree, C=0.2). In Neural Networks we set the number of nodes in the input layer as 30, because after implementation of dummies the number of columns increases from 13 to 30. In general, there are 1 input, 1 output and 3 hidden layers. Output layer consists of 2 nodes indicating 0 or 1 as in target. Hidden layers consist of 800, 500 and 200 nodes consistently. And the batch size is 1. It is notable to mention about the application of sigmoid function as an optimal version rather than ReLU, hyperbolic and others. Furthermore, there we use Adam optimization with learning rate equal to 0.0005 which can regulate scattered gradients on noise issues.

3 Results

After comparing different models we identified that Logistic Regression and SVM with their CV has higher accuracy compared to KNN and Neural Networks. Now, we can explain deeper each model.

3.1 *K-nearest Neighbors*

Firstly, KNN's accuracy is 90%. The important step here is the number of neighbors. We presented in a Figure 3 accuracy for k-values in range 1-10 and it reaches the peak at 90% when k=8. Overall, the trend increases as the number of neighbors are increasing. The range of scores are between 75- 90%. However taking so many neighbors can lead to overfitting. Consequently, it can be seen that as k-value gets higher than

8 then the accuracy for the KNN classifier declines.

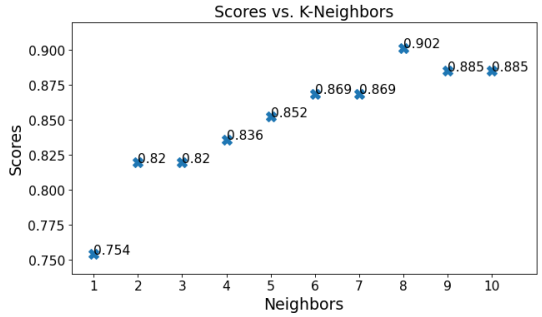

Figure 3: Scores of k-values in range 1-10.

3.2 *Support Vector Machine*

Secondly, the SVM algorithm has been tested. Here we can compare simple SVM and SVM with CV. The main goal of SVM is forming a hyperplane which divides the classes as much as possible. This is done by handling the distance between the hyperplane and the points of the data. There are different kind of Kernels that influence in the forming of the hyperplane. We choose 4 types among them: linear, poly, radial basis function (rbf) and sigmoid. For simple SVM there is a bar graph with accuracies for 4 kernel types. You can see it in Figure 4 which is created by the 'rainbow' method to make it colorful. Initial 2 types were precise for 93%, sigmoid is approximately 92% and rbf is 90%.

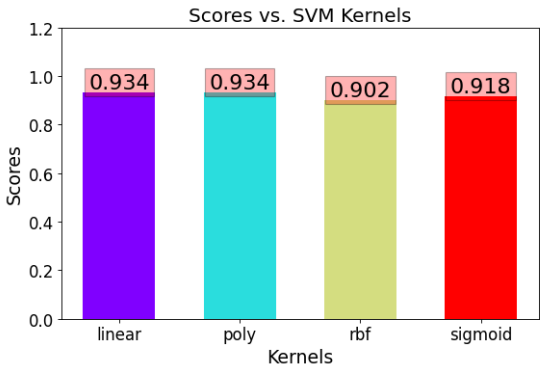

Figure 4: Scores of different SVM Kernel types

3.3 *Support Vector Machine with CV*

For the SVM with CV precision values are higher compared to the simple SVM. Linear Kernel still remains the best with 99% average accuracy, whereas others indicate 93%. Here as CV is included the data are divided into 10 fold and we have calculated an accuracy score for each of them and the 1st fold in all kernel types were lower than others. Additionally, in both cases the rbf kernel types shows the lowest accuracy.

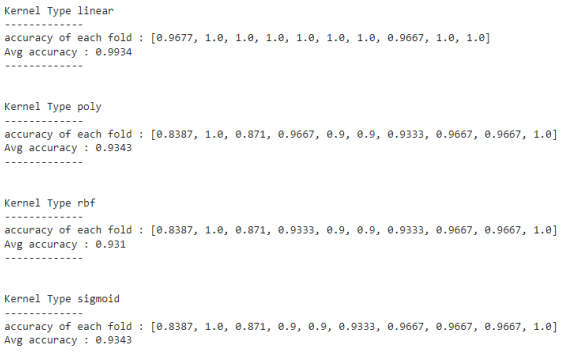

Figure 5: Scores of different SVM Kernel type with CV

3.4 *Logistic Regression*

Logistic Regression's value was the same as for Linear Kernel SVM resulting in high accuracy of 99%. In general, most of the folds show 100% and the exceptions are 2nd and the 10th folds. This is can be because the Logistic Regression is a classification that find a probability of success and failure events.

3.5 *Neural Networks*

The last and essential is implementation of deep neural networks. For our data it was the least optimal algorithm with 83% correctness. In Figure 6 you can see average accuracies for each fold. Here we used torch which provides us Tensor same as 'numpy', but the difference is that it can be run on GPUs. Moreover, it gives automatic differentiation for building and training neural networks.

```
Fold 0: 74.19354838709677 %
Fold 1: 74.19354838709677 %
Fold 2: 83.87096774193549 %
Fold 3: 100.0 %
Fold 4: 70.0 %
Fold 5: 86.66666666666667 %
Fold 6: 93.33333333333333 %
Fold 7: 80.0 %
Fold 8: 86.66666666666667 %
Fold 9: 83.33333333333334 %
Average: 83.2258064516129 %
```

Figure 6: Scores for each fold in Neural Networks

In fact, this value varies each time we run the code because of randomness.

Conclusions

To sum up, we could reach the highest accuracy of 99% in two methods using Logistic Regression and SVM with CV. So, our assumption is justified. Our results are presented in the table below.

KNN	SVM	SVM + CV	Logistic Regression + CV	Neural Networks
90.2%	93.44%	99.34%	99.34%	83.23%

Figure 7: Accuracy results for different models

However, in the researchers' work the data weren't clear and pre-processed. In addition they had 20 features and data set size is 1200, while we had 13 features and data set size is just above 300. When they represented the accuracy of the different algorithms as can be seen in Figure 8, the data is divided by the size of the data set starting from 600 lines to 1200 lines. In all

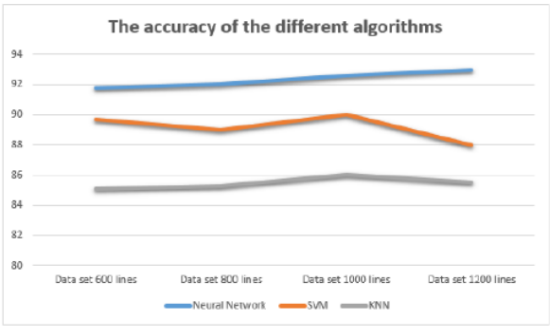

Figure 8: Accuracy results for models with different size of data

of them Neural Networks show the highest accuracy. The difference between the scores of Neural Networks and the SVM decreases as the size of data becomes smaller. And for our data with 300 lines it gives 99% precision. A big size of data set is one of the reasons why the neural networks work better than other methods in their paper. Also the data had to be normalized to avoid overfitting. It was troublesome to use PyTorch in deep learning and we used outer sources to perform CV in neural networks [4]. Another difficulty was to get higher accuracy by changing the hyperparameters. Running the cross validation is very time-consuming, so it took a lot of time to find the best model with their correctly picked hyperparameters. For future improvement we can increase the size of data and deep learning will show better results. In addition, the optimization methods can be changed and then compared with current results.

Contribution of members

Nurbek: selecting best pre-processing, running deep learning and logistic regression models
Nurzhan: running SVM learning model and implementing cross validation
Nursaule: running KNN learning model and writing this report
All other work as searching for paper, information and explanation of code in the video is done together.

References

[1] World Health Organization. Cardiovascular diseases (cvd).

[2] Dhai Eddine Salhi, Abdelkamel Tari, and M-Tahar Kechadi. Using machine learning for heart disease prediction. *Advances in Computing Systems and Applications*, pages 70–81.

[3] Karan Bhanot. Predicting presence of heart diseases using machine learning.

[4] Chris Versloot. How to use k-fold cross validation with pytorch?