Heart Disease Prediction

Qiyuan Shen
Duke Kunshan University
qs31@duke.edu

Tianlai Li
Duke Kunshan University
tl266@duke.edu

Zedian Shao
Duke Kunshan University
zs100@duke.edu

Abstract

Heart diseases pre-detection, a medical problem, has confused both scientist and doctors for decades. Numerous data scientist once tried to figure out perfect model for this prediction problem. Here, we want to figure out the relations between the human body indexes and the heart diseases detection results. We select both traditional machine learning algorithms like Naive Bayes classifiers and newborn popular method called Catboost as well as widely used comprehensive learning algorithm ensembles including Ada boost, Gradient boost, Random Forest and Extra Tree. By comparing all these six algorithms, we could get a relatively best model for heart diseases pre-detection.

1 Introduction

1.1 Motivation

According to the data collected by center for disease control and prevention, there are 65900 people die from heart disease every year in America which means every four people die, there are one people die due to the heart disease. While the problem of heart disease is severer in China. According to the data collected in 2019, there are 290 million patient who have Angiocardiopathy. Recent researches have showed that the prevention of heart disease will greatly decrease the probability of sudden death caused by heart disease. Moreover, there are several features such as Cholesterol, heart rate, etc. As I mentioned above, how to dragonize the heart disease in advance becomes important. So, what we want to do is to apply the knowledge we learnt about machine learning and make some prediction about heart disease.

1.2 Literature Review

There are various attempts delving into this topic and datasets applying machine learning. Depending on the works we obtained, three of them guide the project most. For the first paper, Page and Ray introduced skewing algorithm to greedy decision tree induction to manage problematic functions with lower computational cost compared with standard approach *Lookhead*. They experiment the algorithm on the same dataset and finally shows impressive performance.

In Zhou and Jiang's work, they integrate decision tree and neural network together and generate NeC4.5 which is a novel decision tree. Nec4.5 takes advantage of neural network ensemble to process the data before training the decision tree and this model is experimented on the heart disease dataset and its performance is better than commonly used ensembles.

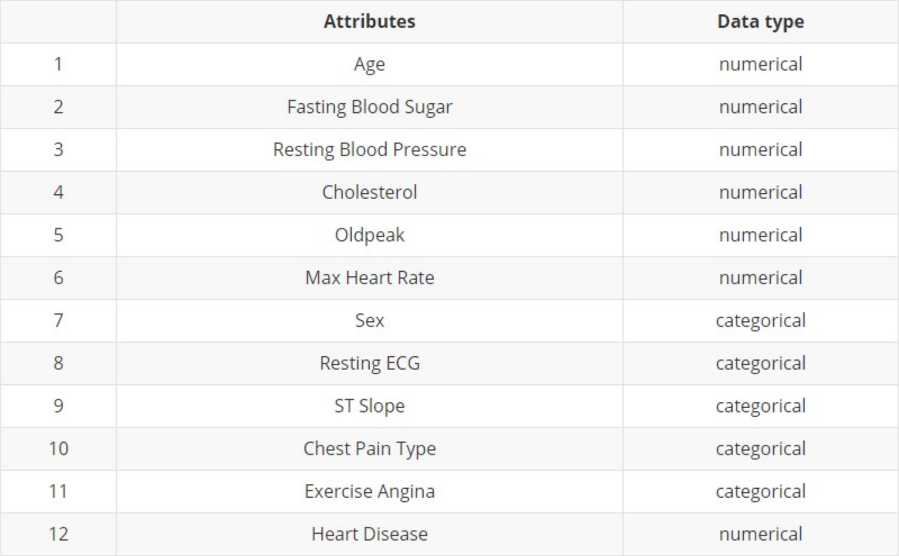
For the third paper, Chai and coauthors proposed a method to get a test-cost sensitive naïve Bayes classifier through a test approach which is able to determine unknown attributes to minimize errors. Their work points out the importance of consideration of test cost associating with searching missing values during model testing.

In this project, we would like to apply both common-used algorithms like Bayesian classifier and ensembles, and also popular and new algorithm like catboost. By comparing the performance of models, we want to understand them better and see the application in real life.

2 Dataset

2.1 Dataset introduction

Our dataset is abstracted from the organization called UCI Machine Learning Repository, which is a quite famous repository gathering numerous data scientists. Furthermore, our dataset has been referred in over 200 articles, which proving the dataset credibility. The dataset has 12 attributes in total, 7 of them are numerical variables and 5 of them are categorical. The attribute named HeartDisease is our prediction object, while the other 11 are features. We made our target as numerical variable for efficient calculation, when HeartDisease equals to 1, the patient does have heart disease. And when it reaches to 0, one does not have that disease. Moreover, we have 900 data examples in datasets for our machine learning project. A large dataset makes our results more confidential.



	Attributes	Data type
1	Age	numerical
2	Fasting Blood Sugar	numerical
3	Resting Blood Pressure	numerical
4	Cholesterol	numerical
5	Oldpeak	numerical
6	Max Heart Rate	numerical
7	Sex	categorical
8	Resting ECG	categorical
9	ST Slope	categorical
10	Chest Pain Type	categorical
11	Exercise Angina	categorical
12	Heart Disease	numerical

Table 1: Attributes in dataset

2.2 Data visualization

In visualization, we mainly did two parts. For categorical variables, we chose to use pie charts, while for numerical ones, we selected the frequency distribution histogram. Pie charts can demonstrate the proportion of each type for corresponding features transparently. On the other hand, frequency distribution histogram could display the trends of distribution of numerical variables. The data for heart diseases patients and non-diseases patients are separated, which is the preparation for further analysis.

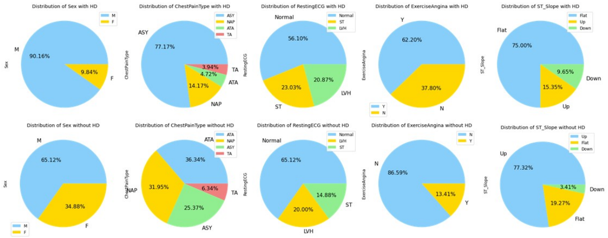


Figure 1: Pie charts for categorical variables

From the visualization, we abstractly comprehend the overall distribution for our data. Apparently, the distribution for categorical variables is reasonable, but we met some troubles in numerical ones.

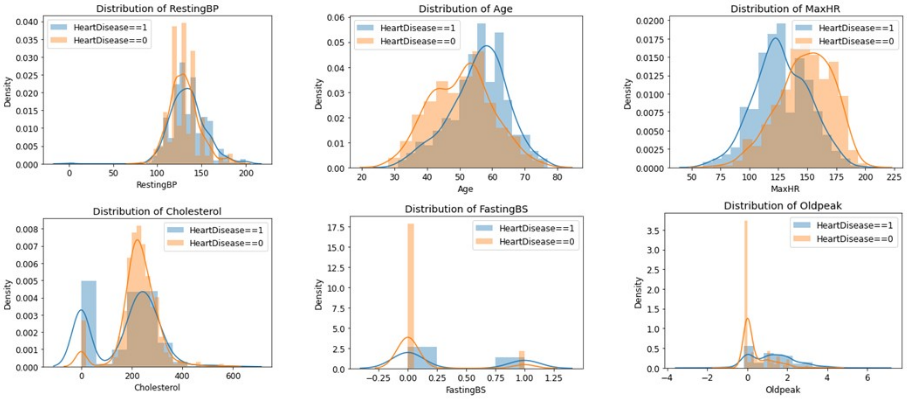
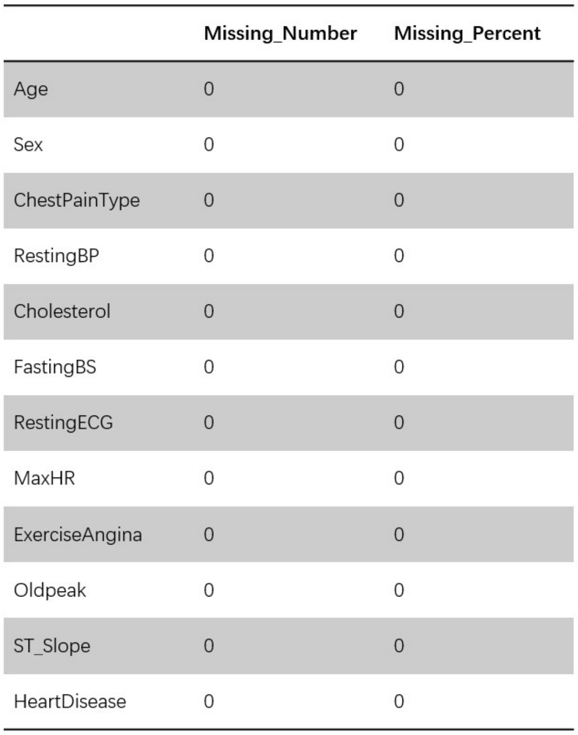


Figure 2: Frequency distribution histogram for numerical variables

As we can see in the figure 2, most of the features are well distributed as gaussian distributions. However, the feature called cholesterol is abnormal. As we all know, cholesterol value can never be zero in human body. The reason for this value reaching to zeros might be the missing ones had been replaced by zeros before, and we need to eliminate the confusion data.

2.3 Data Preprocessing and Cleaning

Generally, the data obtained from the center for machine learning and intelligent system has high quality and utility. No duplicated and null values are observed during the data cleaning process. The result of missing values is shown below. However, multiple zero values are found in the cholesterol feature which is abnormal since cholesterol is a kind of indispensable organic substance in humans' body whose value could not be zero. In this way, it is safe to conclude that the missing values exist in



	Missing_Number	Missing_Percent
Age	0	0
Sex	0	0
ChestPainType	0	0
RestingBP	0	0
Cholesterol	0	0
FastingBS	0	0
RestingECG	0	0
MaxHR	0	0
ExerciseAngina	0	0
Oldpeak	0	0
ST_Slope	0	0
HeartDisease	0	0

Table 2: missing value and missing percent of our dataset

cholesterol feature, and they are replaced by zero before the preprocessing process. The distribution of cholesterol is shown below.

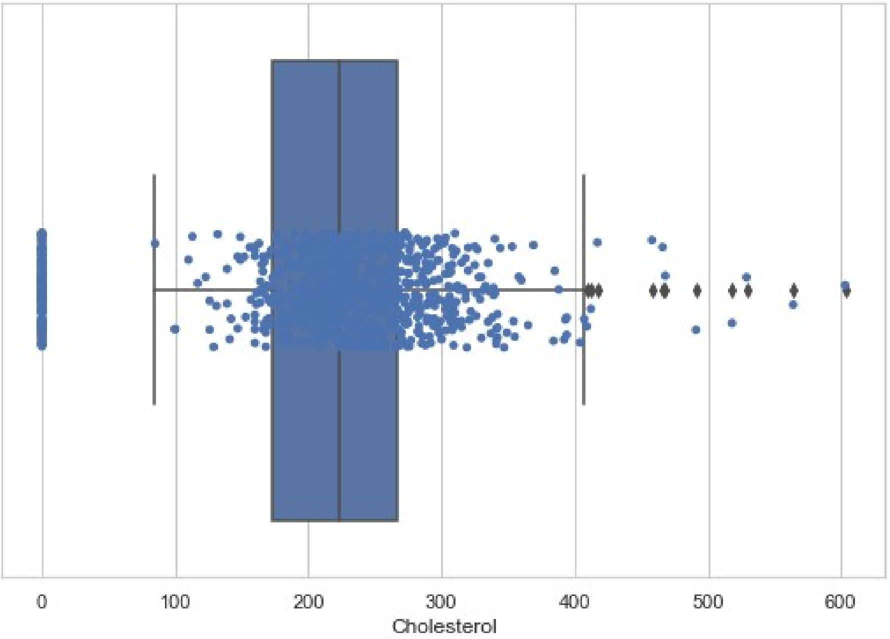


Figure 3: The box figure of cholesterol feature

To avoid the influence of nearly 22 percent of data whose cholesterol feature is zero and other outliers, we decide to clean these outliers in the dataset and keep the entries within the range:

$$[25^{th} - 1.5\delta, 75^{th} + 1.5\delta]$$

where $\delta = 75^{th} - 25^{th}$

3 Method

3.1 Naive Bayes Classifier

We first select to use Naïve Bayes Classifier as our learning model. Naïve Bayes is a simple but powerful model, it selects the largest likelihood as prediction result. In machine learning and data mining, Naïve Bayes could be considered as the basic method, and it could be implemented when features are independent of each other. We chose this model not only for this, but also for the data is high dimensioned. Normally, Naïve Bayes could handle with data efficiently when the dimensionality of inputs is really high. Moreover, as the features are well distributed, the model is predicted to be well performed.

The method called robust scalar is selected as our normalization function. As we mentioned before, the dataset exists outliers. We chose this function to eliminate the influence of extreme values as well as possible. For the calculation function for numerical values, we select Gaussian Naïve Bayes referring to the distribution histogram plotted in visualization part.

$$(X_i|Y) = 1\sqrt{2\pi\sigma_y^2} \exp\left(-(x_i - \mu_y)^2 2\sigma_y^2\right)$$

3.2 Catboost

Catboost is an algorithm for gradient boosting on decision trees. It deals with categorical attributes based on gradient boosting framework through permutation. Catboost has the advantage that it has ordered boosting mode which can overcome overfitting and it can use oblivious trees or symmetric trees for faster execution. When dealing with categorical features, instead of just applying the average of corresponding label as the yardstick of the split of nodes, Catboost takes prior distribution and weights to decrease noise and cut down the influence from categorical values with low accuracy. The transformation could be expressed as

$$\hat{x}_k^i = \frac{\sum_{j=1}^{p-1} [x_{\sigma_{j,k}} = x_{\sigma_{p,k}}] Y_{\sigma_j} + a \cdot p}{\sum_{j=1}^{p-1} [x_{\sigma_{j,k}} = x_{\sigma_{p,k}}] + a}$$

To further reduce the bias, Catboost uses ordered boosting algorithm. In this algorithm, Catboost will train a single model M_i for every sample x_i where M_i is obtained from the training dataset excluding x_i, and then Catboost use this M_i to estimate the gradient of x_i. In this way, ordered boosting will cut down the bias in estimation.

In this project, since nearly half of the dataset are categorical features, Catboost library is applied in this project. It transforms categorical features like `ChestPainType` to numerical features. Considering that the size of dataset we use is not too large to implement ordered boosting, we to reduce bias and relatively optimize the performance.

Moreover, to find suitable hyper-parameters for the Catboost model, we build five models with learning rate changing from 0.01 to 0.05 with step 0.01 and evaluated by their test accuracy in order to find a relatively optimized learning rate for this problem.

3.3 Ensembles

Then we want to test how ensemble will perform on our dataset. We decide to use four popular ensemble algorithms to test the performance. The four algorithms are: adaboost, random forest, gradient boost and extra tree. For each algorithm we mainly use them to classify our dataset into two classes: heart diseases patient and normal.

3.3.1 Adaboost

The first model is adaboost which is also called adaptive boosting. It is a statistical meta-algorithm that trains several different weak learners to form a stronger classifier. The idea of algorithm is shown below:

The importance of a base classifier Ci mainly depends on its error rate

$$\epsilon_i = \frac{1}{N}\left[\sum_{j=1}^{N} w_j I\left(C_i\left(x_i \neq yi\right)\right)\right]$$

Then based on the error rate we can get the importance of a classifier Ci is

$$\alpha_i = \frac{1}{2}\ln\left(\frac{1-\epsilon i}{\epsilon_i}\right)$$

Where the w represent the weight of every function, N is the number of training examples. $I(p) = 1$ if the prediction is true, 0 otherwise. C is the classifier.

The general idea for adaboost is adjusting the weight of every weak learner to get the best training performance and become adaptive. Then for each iteration, we will decrease the weight of incorrectly classified examples and increase the correct ones. After that the model has been built.

3.3.2 Gradient Boost

The gradient boost is a machine learning algorithm that can deal with the regression, classification and other task. Same as adaboost, the gradient boost is also an ensemble of weak decision trees. The main idea of gradient boost method is to narrow the lose function by the way of gradient descending. The main algorithm is shown below.

Suppose we have the θ, and we want to minimize the loss function $L(\theta)$. Then we have the function:

$$\theta = \theta - \alpha\frac{\partial}{\partial\theta}L\left(\theta\right)$$

Then our first step is to initialize the function:

$$f_0\left(x\right) = arg\min\ \sum_{i=1}^{N} L\left(y_i, \right)$$

Then for $m = 1$ to $m = M$ we will operate following step for each iteration.

`Step 1` (calculating the gradient)

$$\tilde{y_i} = -\frac{\partial L\left(y_i, f_{m-1}\left(x_i\right)\right)}{\partial f_{m-1}\left(Xi\right)}$$

`Step 2` (minimize the square error)

$$w_m = arg\min_{w} \sum_{i=1}^{N} \left[\widetilde{yi} - h_m\left(x_i, w\right)\right]^2$$

`Step 3` (use line search to determine the stepsize P_m to minimize L)

$$\rho_m = arg\min_{\rho} \sum_{i=1}^{N} L\left(y_i, f_{m-1}\left(x_i\right) + \rho h_m\left(x_i; w_m\right)\right)$$

`Step 4` (get $f_m(x)$)

$$f_m\left(x\right) = f_{m-1}\left(x\right) + \rho h_m\left(x; w_m\right)$$

After that we will out put $f_m(x)$ as the final model.

3.3.3 Random Forest & Extra Tree

Then I choose the random forest and extra tree as the third model for training and testing. It is a classifier by conducting multiple trees at training time. It is also an ensemble algorithm. And the output is determined by overall result of lots of weak leaners. Here is how the algorithm works:

We have N samples and M feature

`Step 1` We input the feature number m to decide the result of every node on the decision tree, where $m << M$

`Step 2` We use sampling with replacement method to sample for N times to form a training set, use the data out of the sample as the testing set to assess the error.

`Step 3` For every node, we randomly select the features to calculate the best separating method.

`Step 4` Based on the result of different trees we build as above; we can get the whole result for the random forest.

The algorithm for extra tree is also similar to the random forest, but they have two main differences as followed:

1. Random forest uses the bagging model, while the extra tree uses the whole sample, but has randomly selected features. Because it has the process of randomly separation, it will get better result somehow.

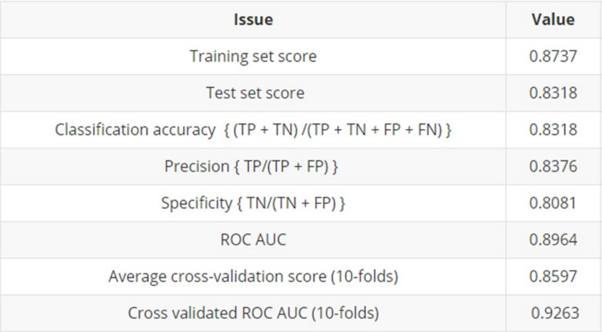
2. Random forest uses the best separation, while the extra tree get the separation just by random selection.

4 Result

4.1 Naive Bayes Classifier

Our results after implementing Naïve Bayes Classifier are demonstrated in Table 1. We can see that the evaluation like test set score, precision and specificity, all these are over 0.8. And we also drawn the roc curve, calculated the ROC AUC which is 0.896. Based on the evaluation table, Naïve Bayes Classifier presents a not bad performance in prediction. Moreover, we even done the cross validation for a little improvement. The confusion matrix and roc curve are also beautiful.

Naïve Bayes performs well in this dataset. The calculation speed is also prepossessing. But for medical issues, accuracy is the first target. We were looking for models with higher accuracy.



Issue	Value
Training set score	0.8737
Test set score	0.8318
Classification accuracy { (TP + TN) /(TP + TN + FP + FN) }	0.8318
Precision { TP/(TP + FP) }	0.8376
Specificity { TN/(TN + FP) }	0.8081
ROC AUC	0.8964
Average cross-validation score (10-folds)	0.8597
Cross validated ROC AUC (10-folds)	0.9263

Table 3: Evaluation table for Gaussian Naïve Bayes Classifier

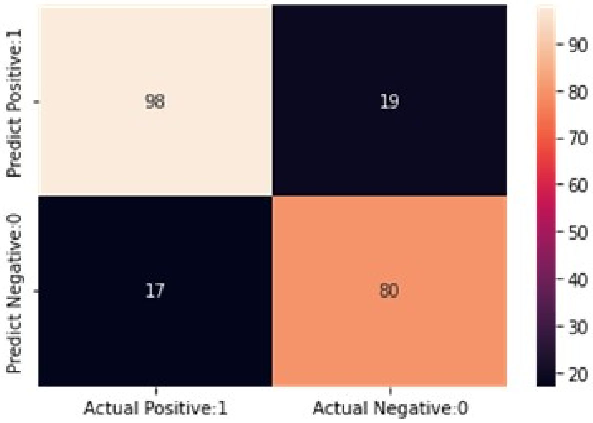


Figure 4: Confusion matrix for Naïve Bayes Classifier

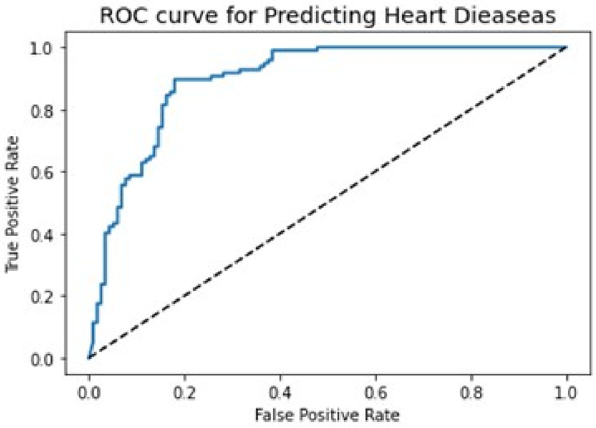


Figure 5: Roc curve for Naïve Bayes Classifier

4.2 Catboost

Table and Figure list the result of the train accuracy and test accuracy with the learning rate from 0.01 to 0.05 with step 0.01. The result is that the model with learning rate equivalent to 0.02 performs best. The poor result of the model with the learning rate equivalent to 0.01 is expected since it is only a third of the default value and it is trained with default numbers of iterations which is 1000. Adding the iteration numbers to 2000 may improve the performance of this specific model.

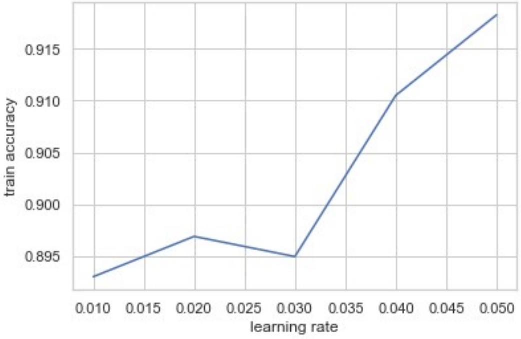


Figure 6: learning rate and train accuracy relation for catboost

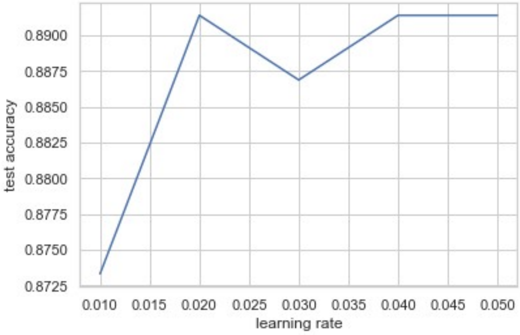


Figure 7: learning rate and test accuracy relation for catboost

For the Catboost model with learning rate equivalent to 0.02, we further evaluate the performance with heat map of confusion matrix and ROC curve.

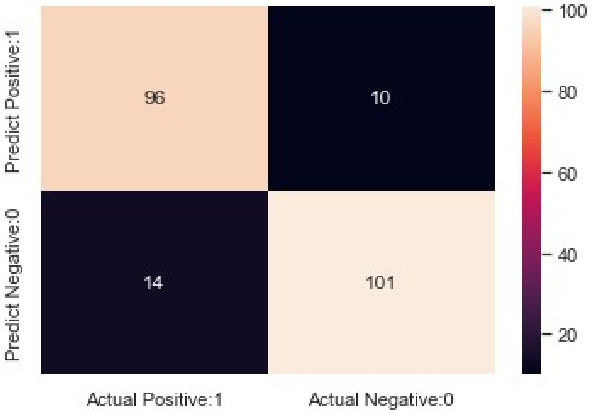


Figure 8: heat map for catboost, learning rate 0.02

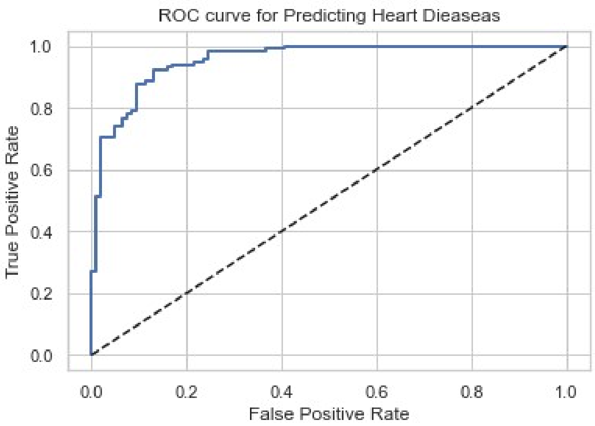
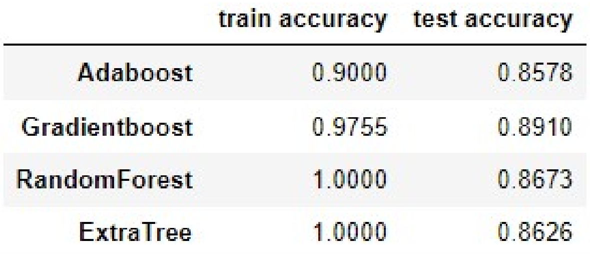


Figure 9: ROC curve for catboost, learning rate 0.02

For this specific model, from figure and figure it has high performance with low overfitting and underfitting, the advantage of Catboost in transforming categorical features to numerical features and application of ordered boosting make Catboost suitable to solve this specific problem.

4.3 Esembles

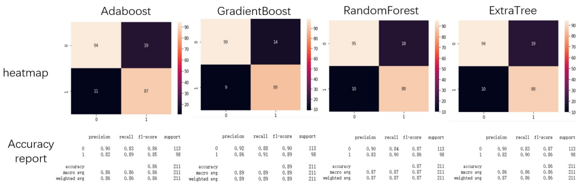
While for the four ensemble methods we have the following results. According to the figure below. The algorithm using decision tree (random forest and extra tree) perform best in the train accuracy which has the 100 percent accuracy. And the adaboost performs worst of both train accuracy and test accuracy. While for the gradient boost, it has the best test accuracy with pretty good train accuracy that the error is acceptable.



	train accuracy	test accuracy
Adaboost	0.9000	0.8578
Gradientboost	0.9755	0.8910
RandomForest	1.0000	0.8673
ExtraTree	1.0000	0.8626

Table 4: training and testing accuracy for 4 ensemble models

Then in order to make our report clearer, we make a heat map and ROC curve to show the training result in a vivid way.



```
              precision    recall  f1-score   support

           0       0.90      0.83      0.86       113
           1       0.82      0.89      0.85        98

    accuracy                           0.86       211
   macro avg       0.86      0.86      0.86       211
weighted avg       0.86      0.86      0.86       211
```

```
              precision    recall  f1-score   support

           0       0.92      0.88      0.90       113
           1       0.86      0.91      0.89        98

    accuracy                           0.89       211
   macro avg       0.89      0.89      0.89       211
weighted avg       0.89      0.89      0.89       211
```

```
              precision    recall  f1-score   support

           0       0.90      0.84      0.87       113
           1       0.83      0.90      0.86        98

    accuracy                           0.87       211
   macro avg       0.87      0.87      0.87       211
weighted avg       0.87      0.87      0.87       211
```

```
              precision    recall  f1-score   support

           0       0.90      0.83      0.87       113
           1       0.82      0.90      0.86        98

    accuracy                           0.86       211
   macro avg       0.86      0.86      0.86       211
weighted avg       0.87      0.86      0.86       211
```

Figure 10: heat map for 4 ensemble models

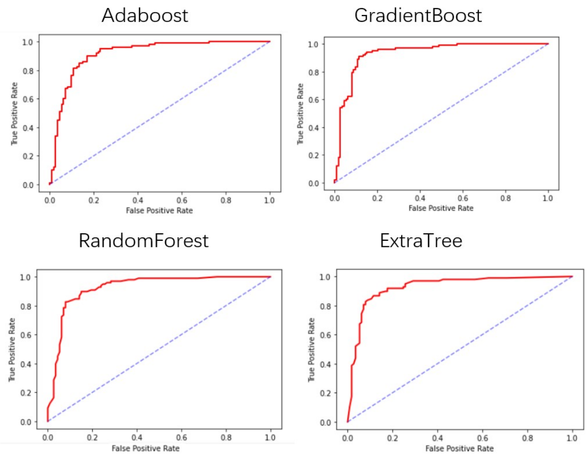
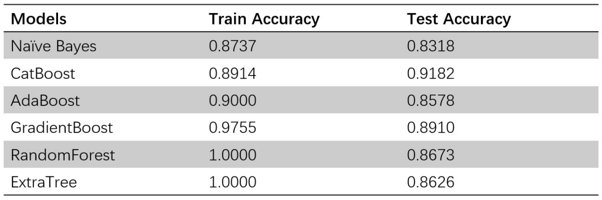


Figure 11: ROC curve for 4 ensemble models

5 Conclusion & Discussion

As the figure shown below this is train accuracy and test accuracy about the 6 models we build. For those algorithms with decision trees (random forest, extra tree), it has the 100 precent correct, but its test accuracy isn't so good. This has caused the overfitting problem that is always seen in the decision tree. So RandomForest and ExtraTree aren't the algorithm we want. While for the Naïve bayes, its performance on train accuracy and test accuracy isn't so good. So, it may because of the underfitting problem. After discussion, we choose catboost as the final model of the heart disease prediction, because it has the best test accuracy, and we can bear the error of the train accuracy.



Models	Train Accuracy	Test Accuracy
Naïve Bayes	0.8737	0.8318
CatBoost	0.8914	0.9182
AdaBoost	0.9000	0.8578
GradientBoost	0.9755	0.8910
RandomForest	1.0000	0.8673
ExtraTree	1.0000	0.8626

Table 5: overall accuracy for this project

For further work, we mainly want to separate it into 3 parts. Since we haven't found any detailed data of heart diseases in China and the database of our project is too old. So, the first step is to collect the data of Chinese patient nowadays since there are many differences between Chinese and American in lifestyle such as diet and exercise habit. Then, the next step is to test more models. Due to the limit time and our knowledge of machine learning, we can only test 6 models, if we have more time, we will test on more models or develop one by ourselves. After that, the third step is to develop an algorithm that can give advice based on the prediction of heart disease to make patients live a better life.

References

[1] Page, D., & Ray, S. (2003, August). Skewing: An efficient alternative to lookahead for decision tree induction. In *IJCAI* (pp. 601-612).

[2] Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2017). *CatBoost: unbiased boosting with categorical features.* arXiv preprint arXiv:1706.09516.

[3] Sanoob, M. U., Madhu, A., Ajesh, K., & Varghese, S. M. (2016). Artificial neural network for diagnosis of pancreatic cancer.*Int. J. Cybernet. Inform,* 5(2), 40.

[4] Zhou, Z. H., & Jiang, Y. (2004). *NeC4. 5: neural ensemble based C4. 5.* IEEE Transactions on Knowledge and Data Engineering, 16(6), 770-773.