**3.1 Classification problem**

We use a data set which classified if a person has heart disease or not. We try to use this data set to create different classification models which can predict if a person has heart disease or not. So, it is obvious that our data set is a binary classification problem.



Fig. Aim of Classification

**3.2 Classifiers for our data set**

3.2.1 Baseline Model

In classification, we use a basic logistic regression model with a bias term and no features as our baseline model, this baseline model can be used to predict whichever label occur more frequently in our data set.

We used our data set to fit a baseline model, and the results as follows:

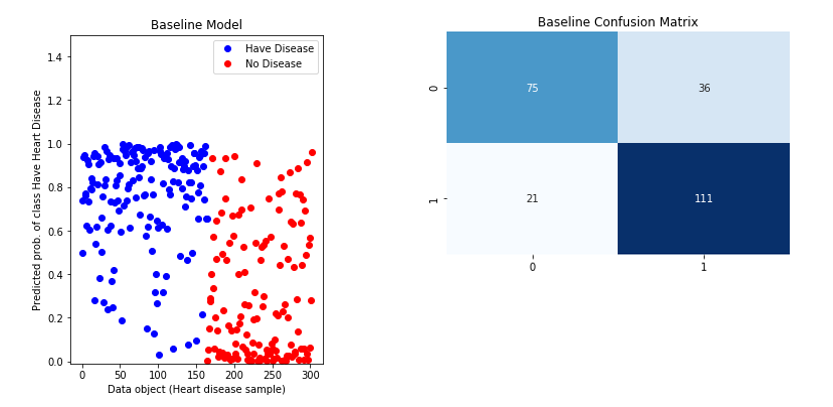


Fig. Performance of baseline model

From the left-hand pane, we get the probabilities of whether a person have heart disease from baseline model, and based on these probabilities, we can calculate the predict value of our test data and determine which class the data belong to. In the right-hand pane, we generate a confusion matrix of test data to show how often the classifier is right. We can also calculate the accuracy and error rate based on following expressions:

So, the results will be:

* Test Error: 23.45679012345679%
* Accuracy: 76.5432098765432%

3.2.2 Logistic Regression with regularization term

In this section, we use regularization term in logistic regression model to manage model complexity. Based on the principle of regularization term in textbook [1], we know that when is large, the models have low variance but high bias; when is large, the models have high variance but low bias. Adding regularization term does not improve the performance on the data set that the algorithm used to learn the model parameters. However, it can improve the performance on new data, which is exactly what we want.

So, in our codes, we designed a for-loop to find the optimal from 10^-5 to 10^3. We use

train\_test\_split(X, y, test\_size=.50, random\_state=0, stratify=y)

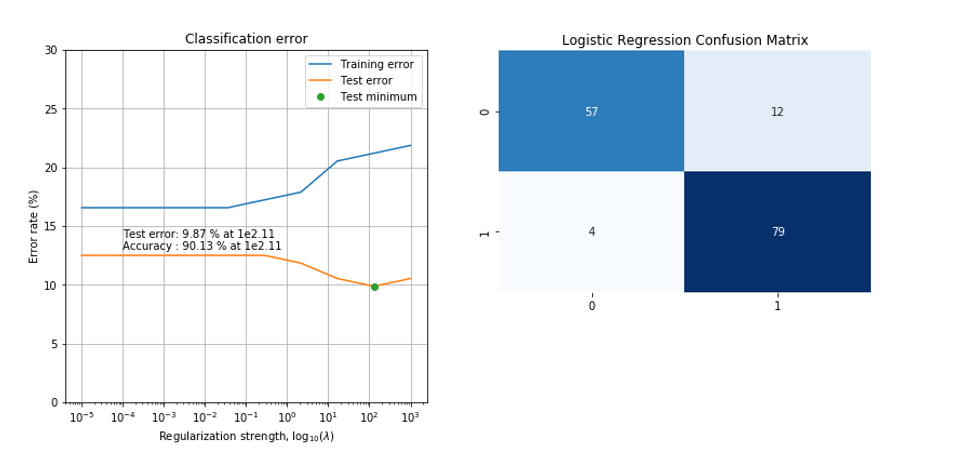
to fairly split the data into two part: train data and test data, and then we selected the optimal by comparing their test error. The optimal and the Confusion matrix of Logistic regression as follows:

Fig. Performance of Logistic Regression

In the left-hand pane, we can find the optimal , we also get the minimum test error when equal to this value. In the right-hand pane, we generate a confusion matrix from test data using logistic regression, the accuracy and error values as follows:

* Test Error: 9.868421052631579%
* Accuracy: 90.13157894736842%

3.2.3 Decision Tree with pruning

Decision tree builds classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. In our codes, we use a technique called *Pruning* in order to control the tree complexity.

We designed a loop with different maximum tree depth and implement this parameter by adding it into function such as

DecisionTreeClassifier(criterion='entropy', max\_depth=t)

Then we can select the best tree depth based on their test errors. We also use ‘Entropy’ as our criterion because it performs better than ‘Gini’. The results of decision tree as follows:

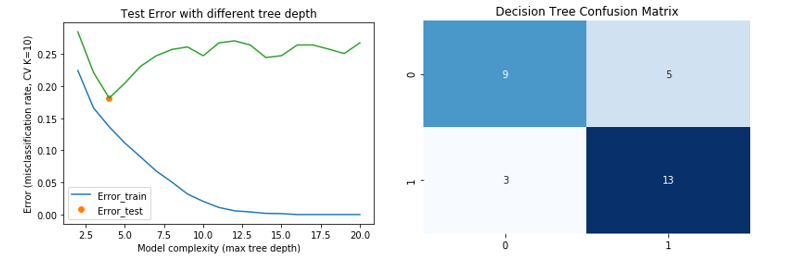


Fig. Performance of Decision tree

In the left-hand pane, we draw a picture to show the test error with different tree depth, it’s obvious that we get the best tree depth which equal to 4, and in the right-hand pane we generate the confusion matrix based on this depth.

We can also get the accuracy and test error value:

* Test Error: 18.150537634408603%
* Accuracy: 81.8494623655914%

**3.3 Model selection by two-level cross-validation**

In this section, we created a table and compared the logistic regression, decision tree, and baseline. We use 2-level cross-validation to calculate select the model and training the model with its optimal parameter. In inner loop, we selected the parameter for Logistic Regression and Decision Tree, and in outer loop, we trained these two models with the optimal complex-controlling parameters, and finally we get 10 optimal parameter sin 10 folds and 10 test errors.

The results of the parameter and test error in each fold as follows:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Outer fold | Logistic Regression | | Decision Tree | | Baseline |
| *i* |  |  | ***t*** |  |  |
| *1* | 1e-8 | 16.12903226 | 2 | 15.72146177 | 15.84147334 |
| *2* | 27.8256 | 12.90322581 | 4 | 12.21001221 | 16.78978429 |
| *3* | 1 | 16.12903226 | 11 | 15.75091575 | 16.43365893 |
| *4* | 1e-8 | 6.4516129 | 2 | 11.46616541 | 17.28327228 |
| *5* | 1e-8 | 12.90322591 | 2 | 13.8053467 | 17.68009768 |
| *6* | 1e-8 | 13.3333333 | 6 | 13.64522417 | 16.54456654 |
| *7* | 774.264 | 13.82478632 | 4 | 13.34318703 | 16.00936101 |
| *8* | 1e-8 | 13.17867318 | 2 | 18.05903648 | 16.51200651 |
| *9* | 59.9484 | 17.24137931 | 2 | 15.74770259 | 16.02055352 |
| *10* | 2.15443 | 10.34482759 | 3 | 10.27568922 | 17.16117216 |

Table 1: Two-level cross-validation table used to compare the three models

From this table, we can say that the test errors of baseline are relatively stable among all folds, we think the reason is that the baseline model doesn’t add any parameter so that it will not be affected too much in different folds. The logistic regression model and decision tree model have their own complex-controlling parameter, and this parameter will be strongly affected in different folds because they have different train and test data in different folds. So, the parameter and will be a little different in each fold.

**3.4 Statistical evaluation**

When we use cross-validation to compare three models, we need to know how to compare them, and is there a true difference in different models. Based on Bayes’ theory, we know there is a symmetric interval which named credibility interval, we can use this interval to determine whether one model is better that the other. We can calculate the difference , and generate a 95% credibility interval based on this difference.

The result of the comparisons of different 3 models as follows:

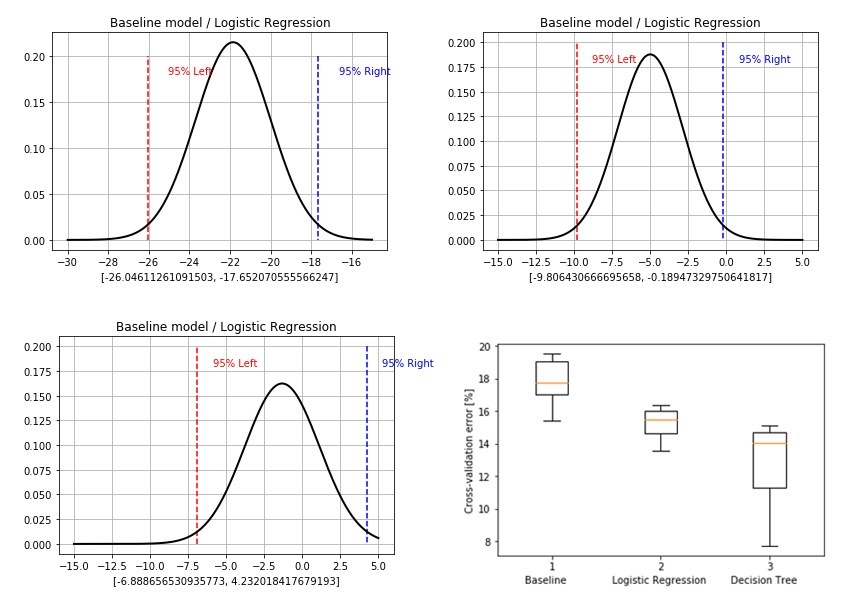


Fig. 95% credibility interval for comparing three models

It’s clear that there are significantly different between baseline and decision tree, and also between logistic regression and decision tree, but there is almost not significantly different between baseline and logistic regression.

Based on the previous discussion, we can say that in our data set, decision tree is better than logistic regression and baseline, and logistic regression is a little better than baseline. And the results of error evaluation for different models can be found in the bottom right-hand pane.

The tables of comparisons as follows:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | *K* | *v* |  |  |  |  |
| *Baseline*  *Decision Tree* | 10 | 9 | -21.84909 | 1.855318 | -26.04611 | -17.65207 |
| *Logistic Regression*  *Decision Tree* | 10 | 9 | -4.997951 | 2.125616 | -9.806430 | -0.189473 |
| *Baseline*  *Logistic Regression* | 10 | 9 | -1.328319 | 2.457980 | -6.888656 | 4.232018 |

Table. 2: Credibility interval for comparing three models

**3.5 Logistic regression using regularization term λ**

In this part, we train a logistic regression model with a suitable regularization term.

We also did a simple feature selection using logistic regression model that we trained before to identify which feature is more important is our data set. We use KFold() and Forward Selection to implement this function.

We generate a picture of the features which were selected in each fold, the results as follows:

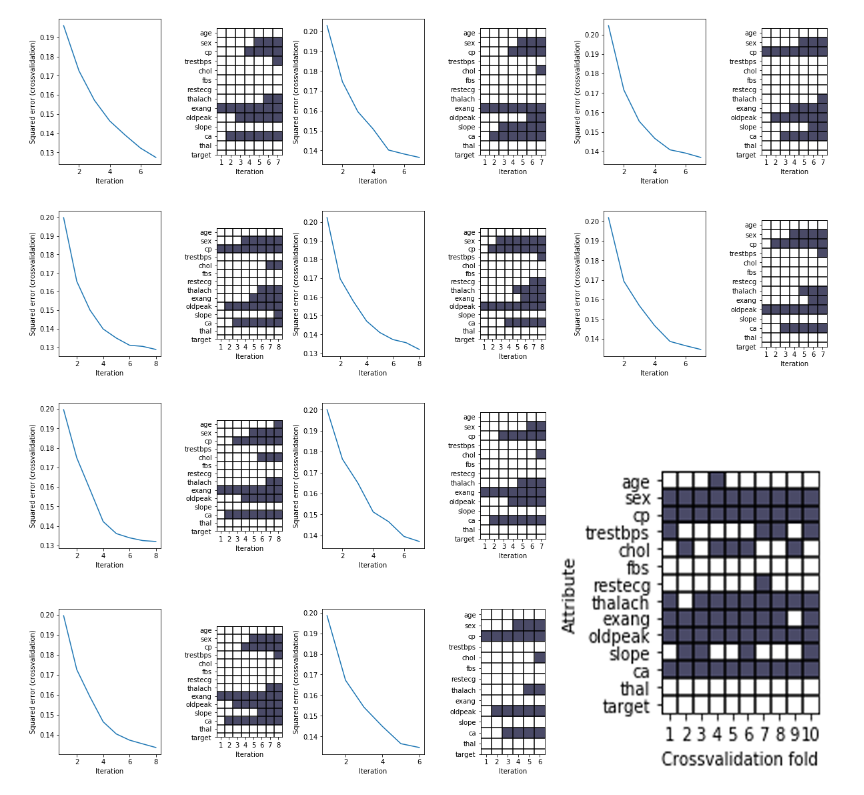


Fig. Feature selection

From this figure, we can say that ‘sex’, ‘cp’, ‘thalach’, ‘exang’, ‘oldpeak’ and ‘ca’ are essential in our data set and we can also get a better performance in training by selecting these features.

Comparing with the features in regression part, these features are not relevant as for the regression part because we use different attribute as our target in classification, and the interdependency between attributes are totally different.