Project 4 Decision Tree

Parser

As requirement, the project should have ability to parser special dataset in txt format and extract data from them. So the parser python file is used to complete the task.

Parser function

It read the data from the file then remove the useless space and ‘,’ from data.

Test

Main function

In this function it is the main entrance for the program.

Validation function

In this part, I choose k cross-validation as test method to check learning performance. It can change k value to set the size of test set and training set. Finally take the average of all correct rates in different situations.

Randomorder function

Sometimes the dataset from UCI database are not as same as other dataset. For example the classier attribute are not the last one in line. So in this function I will adjust the index of the main attributes. And I distribute all the data in dataset in random order which will improve the correctness and validness of final result.

Tree

Node class

It defines the structure of the node in decision tree. Every node will have the several attributes including attribute – the chosen attribute, parent – the chosen attribute of the node's parent, value – the value of the chosen attribute in the node, next – the branches of the node, output – the classier attribute's value. If the node is not terminal node, the output will be None.

Dt class

It defines the decision tree database. It contains examples - the dataset, attribute - all the attributes, and importance -the Entropy set, values – possible values of every attributes.

CreateDatebase function

It will save all the possible value of every attributes from dataset.

Rank function

It adapts information entropy as entropy function and as definition of the function to rank all the attribute importance.

calH function

It just uses the mathematic function to calculate the entropy.

Remain function

it will extract all the date from dataset based on the chosen attribute. And then the new dataset will be used for next cursively processing.

Classification function

When all the remaining example have the same value of classier attribute, it can be regarded as terminal node. And the function will return the output value of classier function.

Plurality_value function

When the node is not terminal node, but there is no other attribute remained it will choose value for classier attribute based on the proportion in the remaining dataset.

SelectAttr function

When creating every node it will choose the best attribute based on their entropy.

CreateTree function

It is like the algorithm in AIAM 18.3. and used to create the whole decision tree. When there is no attribute left or the remaining dataset can be classified into one label. It just create the terminal node. Or it recursively creates the subtree.

Test function

It is for testing and input test date it will search the whole tree and find the label as tree. Then compare with test data label.

CheckTree function

It will recursively search the whole tree and find the label.

Result and Analysis

Loss function: for my dataset, the different class has the same utility. So for absolute loss function - $L_1(y, \hat{y}) = |y - \hat{y}|$, it can be expressed in correctness rate form.

Entropy function:

Empirical entropy H(D) based on dataset D :

$$H(D) = -\sum_{k=1}^{K} \frac{|C_k|}{|D|} \log_2 \frac{|C_k|}{|D|}$$

Empirical and conditional entropy with A attribute H(D|A) based on dataset D :

$$H(D|A) = \sum_{i=1}^{n} \frac{|D_i|}{|D|} H(D_i) = -\sum_{i=1}^{n} \frac{|D_i|}{|D|} \sum_{k=1}^{K} \frac{|D_{ik}|}{|D_i|} \log_2 \frac{|D_{ik}|}{|D_i|}$$

Gain g:

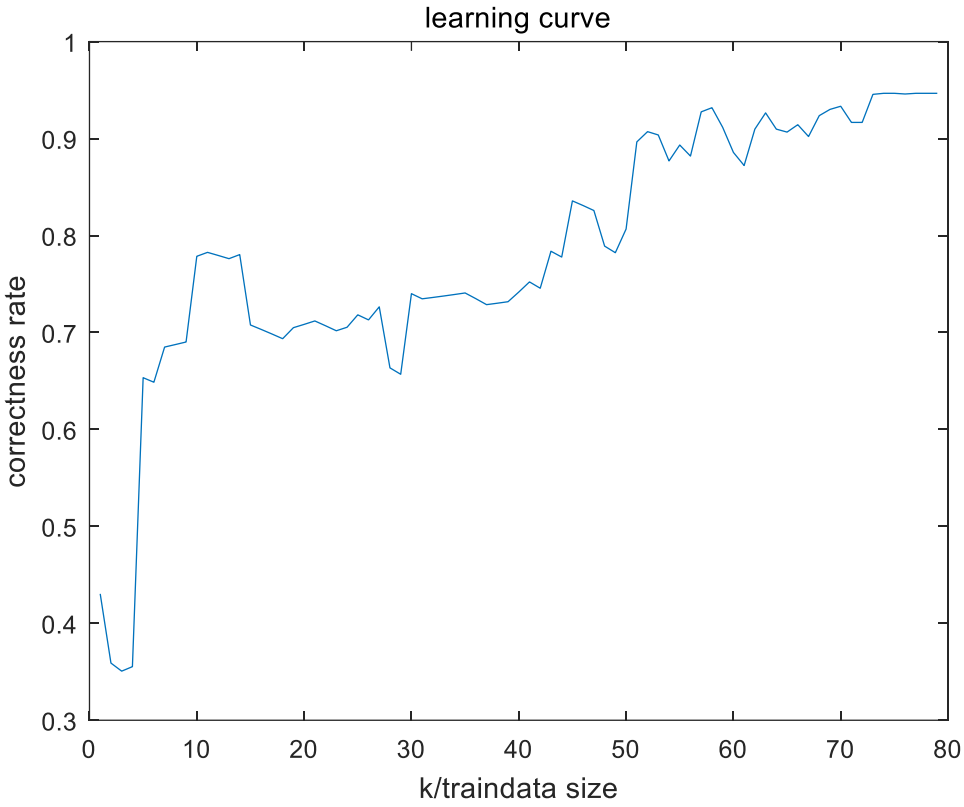
$$g(D, A) = H(D) - H(D|A)$$

Note

The **Loss function** and **Entropy function** are the same when processing different dataset.

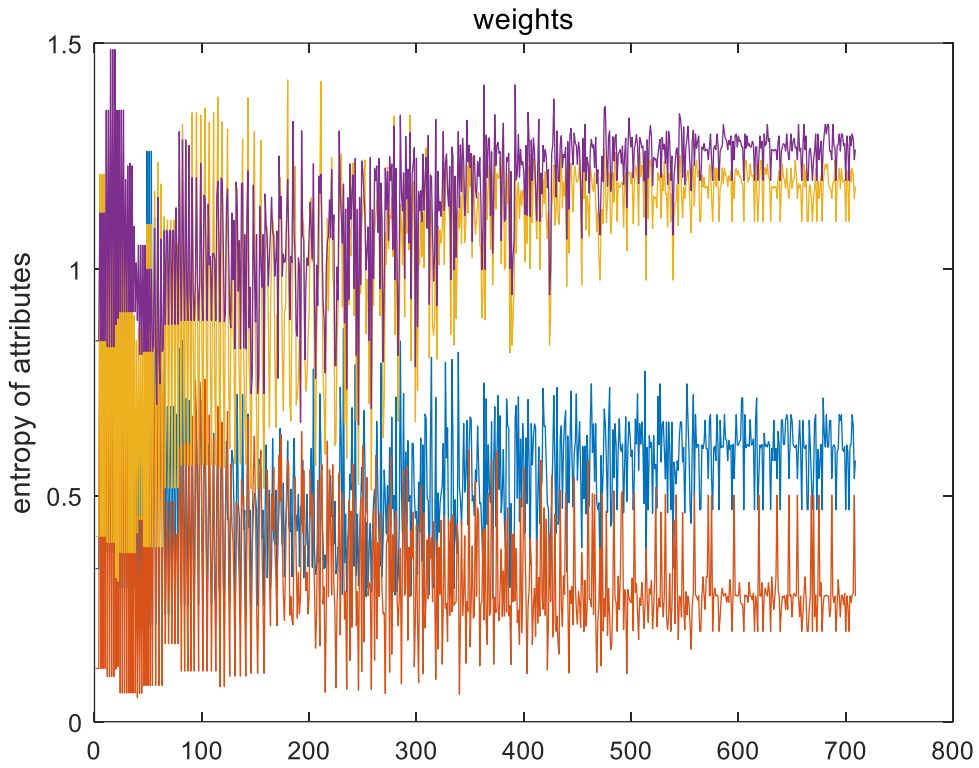
For Iris-discrete dataset

the picture blow is learning curve about the correctness change as traindata size changing:



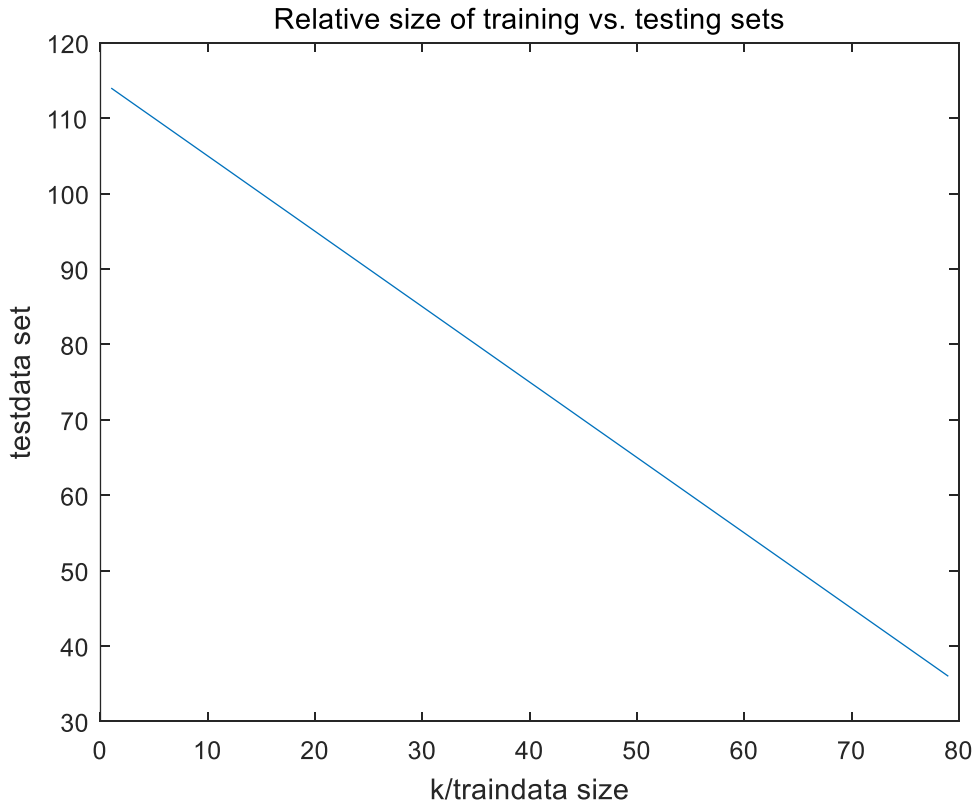
K is traindata set and it is also the number of partitions for cross-validation (k)

As we can see, the correctness rate is increasing as the training size raises. Then it will keep relatively consistent at 95%. The size of whole dataset is more than 100 which basically meet the numerical requirement of learning.



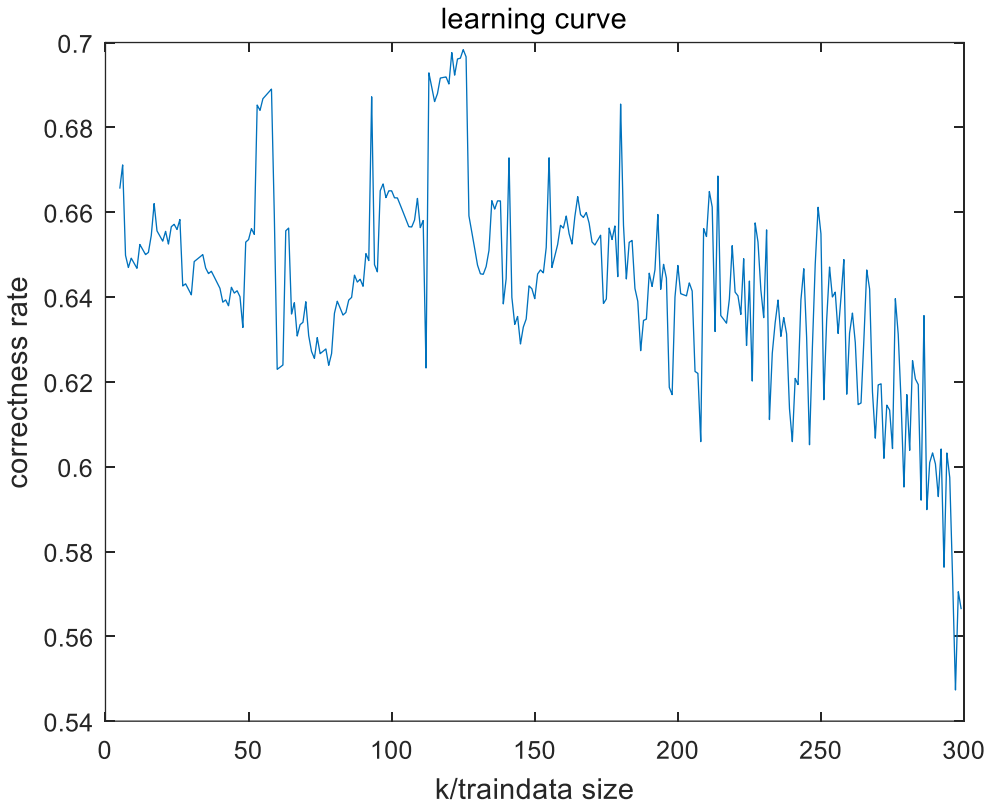
This is the entropy of different attributes. As the training size changes, the entropy becomes more and more consistent.

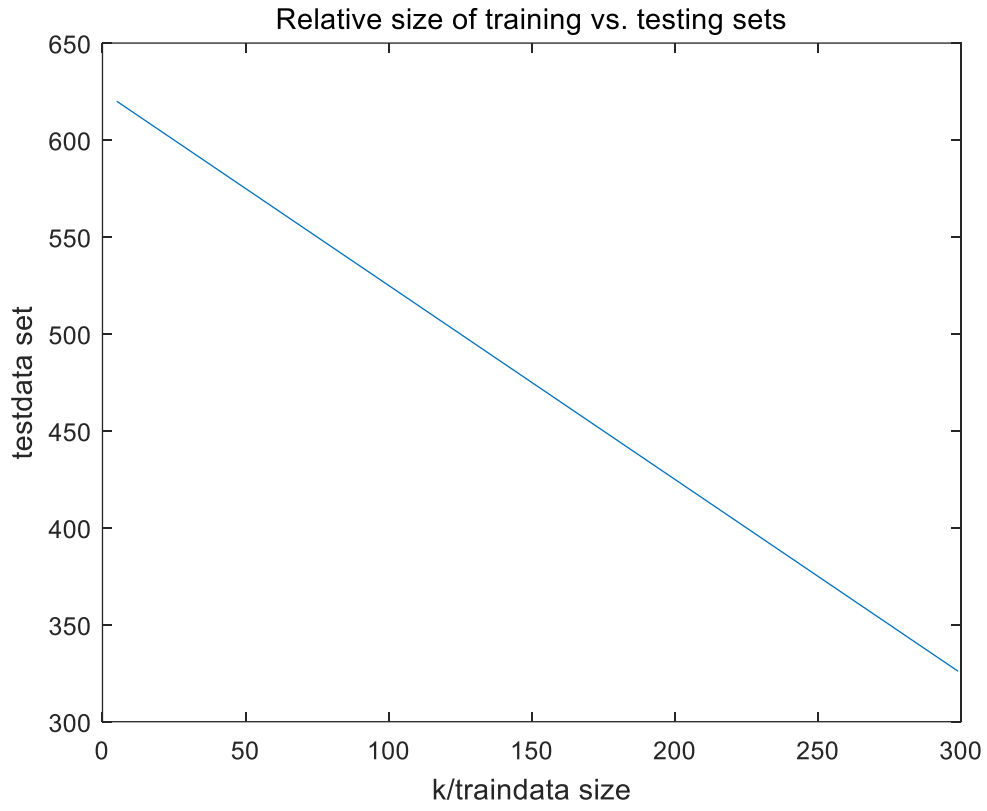
This is comparison between training size and testing size.

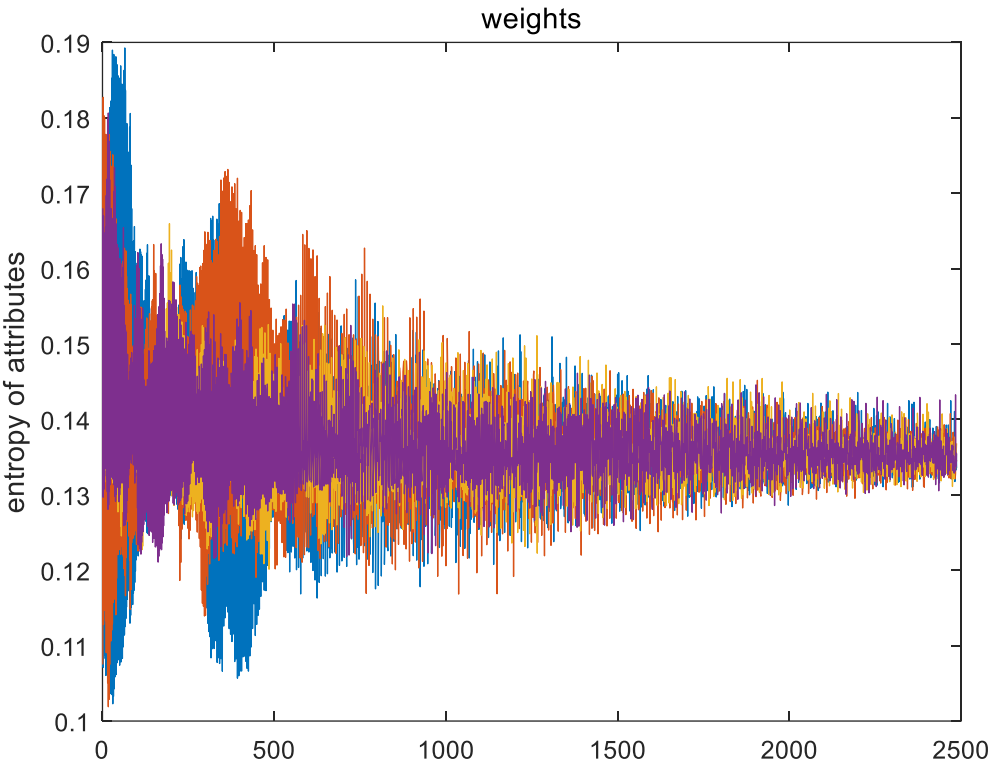


For iris data

This dataset is from UCI repository and I transform the data into txt file. And it contains 625 samples. the picture blow is learning curve about the correctness change as traindata size changing





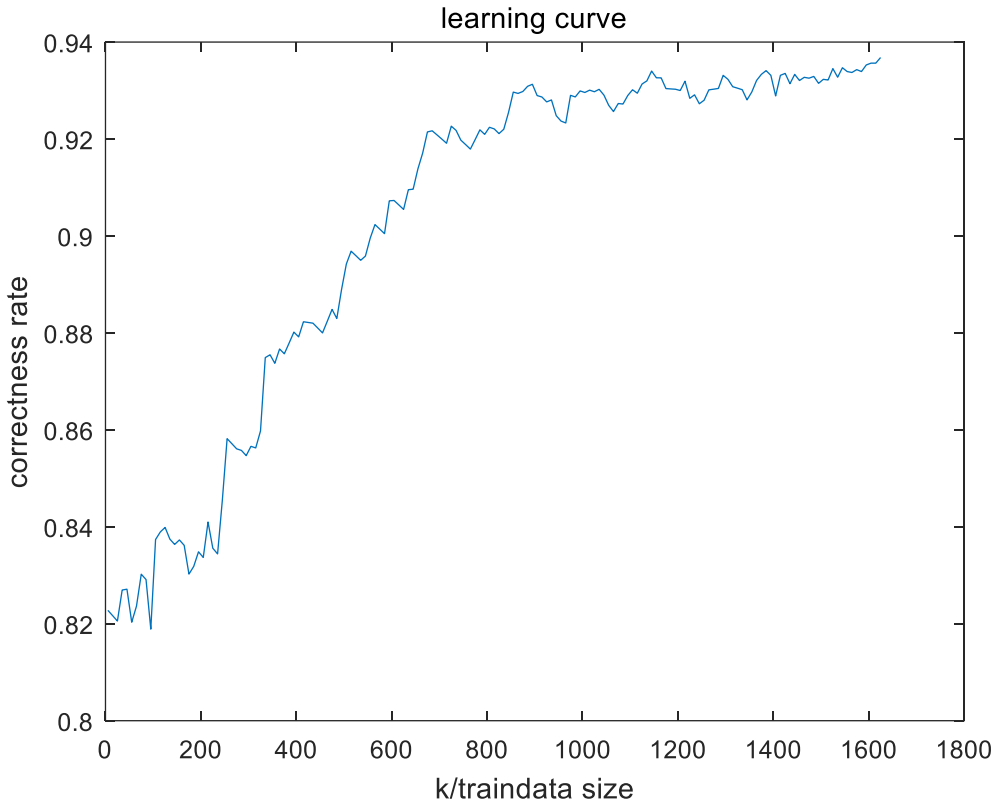


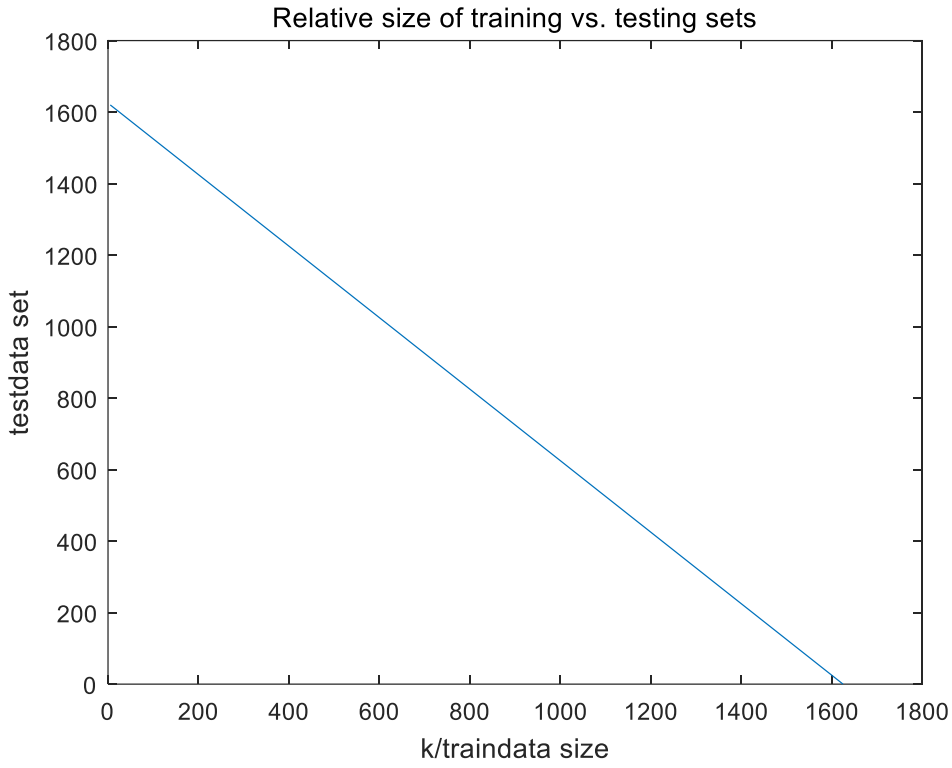
As the picture all the attribute provide the same entropy in the end. So the learning process cannot be reliable. Besides, as we can see even though the traindata size is raising but the correctness rate doesn't follow the trend. It is because the testdata size is too large and there are many noises in the tree or there happens overfitting phenomenon. So the correctness rate keep at 0.66 and gets its peak.

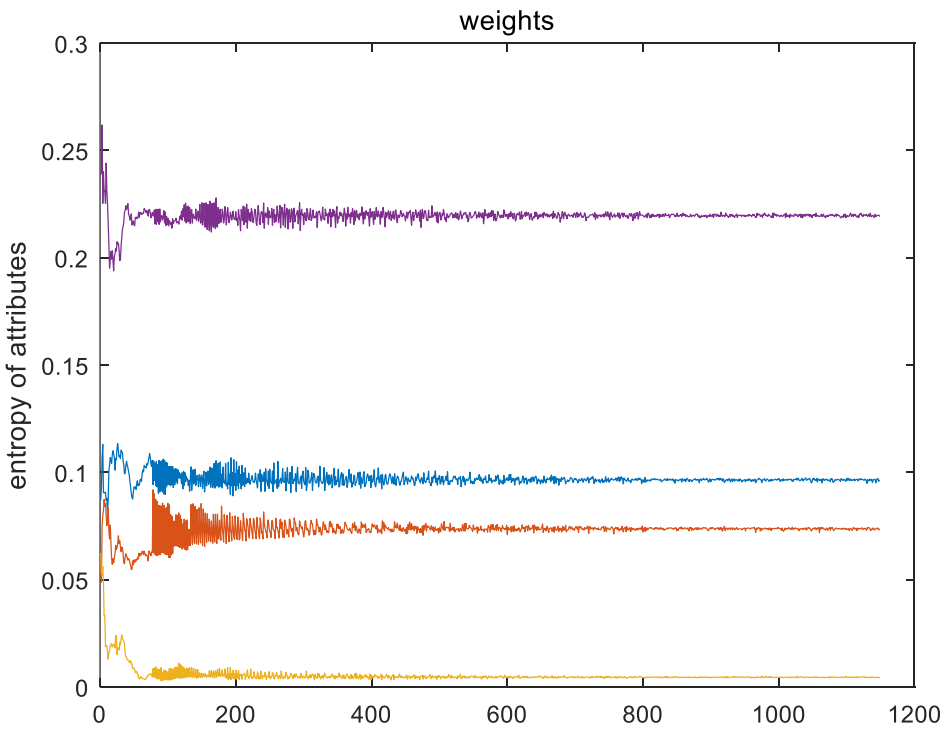
The decision tree just like randomly judge but as traindata increases the learning effect make the rate decreases at 0.62. It is because when traindata become larger the overfitting phenomenon become more serious. So the learning become more unreliable. In this case the rate get decreased.

Moreover, the value of attributes is number. In fact it implies that the different value will get different entropy, but according to the design the loss function is the same. So in this situation the decision tree are not suitable for the dataset. And if neural network algorithm can be applied in the dataset, or change the entropy function as dataset implies, the correctness rate will much increase.

For car dataset







For this dataset, the size of dataset has been more than 1700, so it is really suitable for learning process. Even though two of all attributes still divided into integer, but others are totally discrete string which means their utility are the same. In this case decision tree can make effect. As the picture, the correctness rate continues raise until it gets 93% which satisfied the learning process. As the trainset is increasing the tree become more suitable. Besides, all the attributes' entropy get consistent when training dataset's size become larger.