Machine Learning 2016

HW-5：Random Forest

Deadlines: **2016.06.07**-23:59:59

In this homework, you are asked to use the **Random Forest** models dealing with the classification problem of the **hand-written numbers** $\boldsymbol{x}$. The whole mission can be accomplished easily with the Matlab function **TreeBagger()** which is highly recommended to adopt. The explanation website of the function is shown bellowed:

- http://www.mathworks.com/help/stats/treebagger.html

◆ Data

Input data are the extracted information from the picture of the hand-written numbers (**MNIST** Database). There are two data files offered: **Training _data_hw5.mat** file and one **Test_data4_hw5.mat** file.

Training _data_hw5.mat:

- **X_train** is a 20000x784 matrix. Every row corresponds to a 28x28 gray-scale image.
- **T_train** is a 20000x1 matrix, which records the class of the training samples.

Test_data4_hw5.mat:

- **X_test** is a 5000x784 matrix. Every row corresponds to a 28x28 gray-scale image.
- **T_test** is a 5000x1 matrix, which records the class of the test samples.

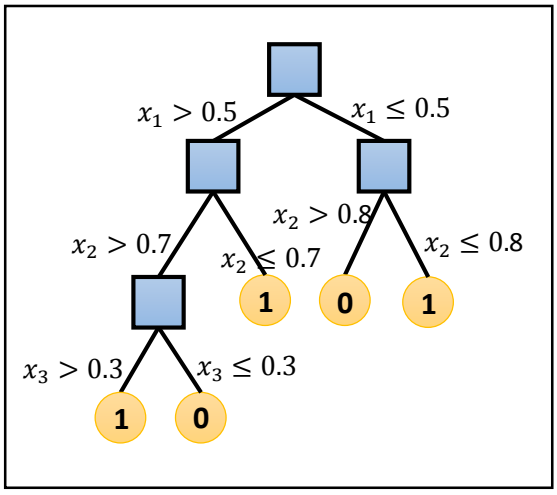
(Please do remember that the test data is not used for optimizing your model!!)

◆ Models

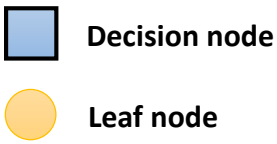
A random forest model is constructed from lots of "**decision trees**," and that's why we call it "**forest**." Before trying to use the random forest models to solve the problem, you must know what a decision tree is first.

● Decision Tree

A decision tree is a flowchart-like structure in which each **decision node** represents a "test" on an attribute (e.g. whether a coin flip comes up heads or tails), each **branch** represents the outcome of the test and each **leaf node** represents a class label (decision taken after computing all attributes). The paths from root to leaf represents classification rules.

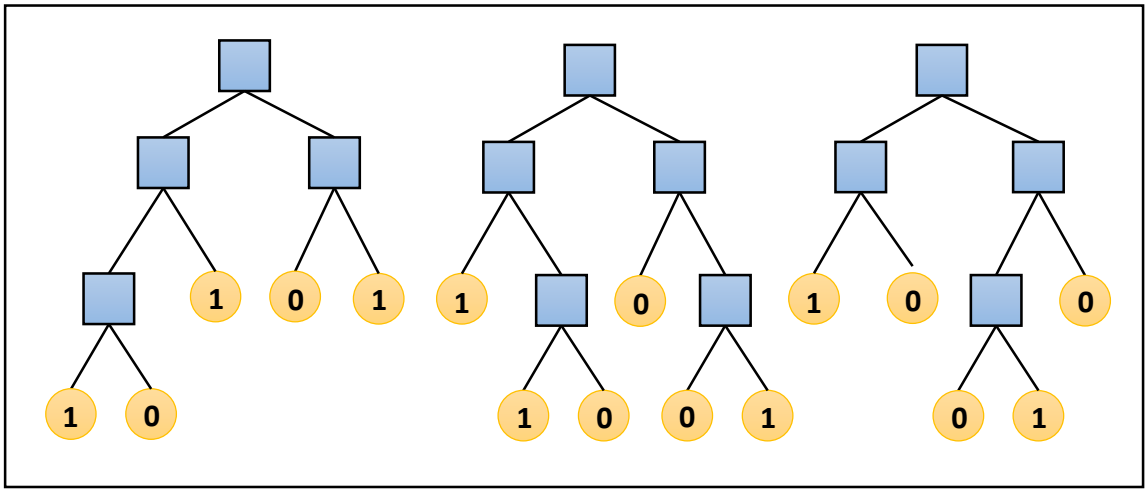


$$X = \begin{bmatrix} 0.1 & 0.7 & 0.6 \\ 0.6 & 0.2 & 0.6 \\ 0.7 & 0.8 & 0.2 \\ 0.6 & 0.9 & 0.4 \\ 0.3 & 0.9 & 0.1 \end{bmatrix} \qquad T = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$



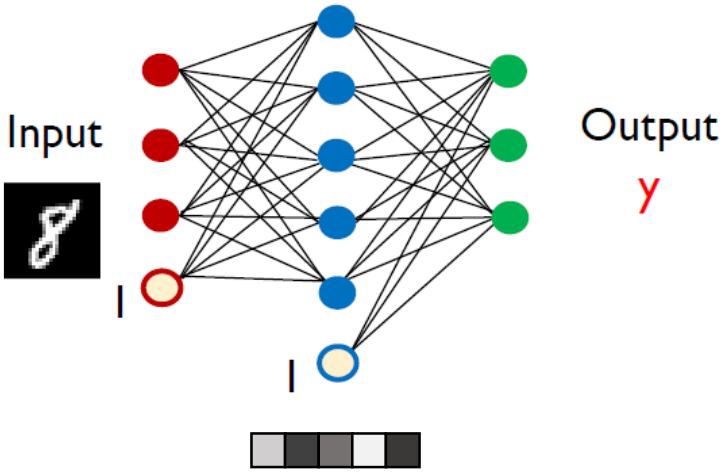
● Random Forest

Combining lots of decision trees with respect to the same training data, a random forest is built for averaging the predictions from theses decision trees to achieve a more stable and robust outcome.



- **Feature Extractor**

  Since the random forest models have trouble with the high-dimensional curse, it is recommended to do some pre-processing. Adding a feature extractor is a quiet fascinating measure to enhance the performance. For example, a one-hidden-layer **neural network** with the size of the hidden layer equals to five help to compress the origin information into a five-dimensional vector (representation).



  Besides, the **Principal Component Analysis** is a popular dimensionality reduction algorithm which can also be used as a feature extractor. Help yourself to find the appropriate feature extractors for intensifying the random forest models.

◆ Tasks

1. Random Forest

Please implement a random forest which contains 100 decision tress. There are some settings should be followed:

- The minimum number of samples per leaf node $= 1000$
- The fraction of samples to be randomly selected with replacement for constructing each decision tree $= 50\%$

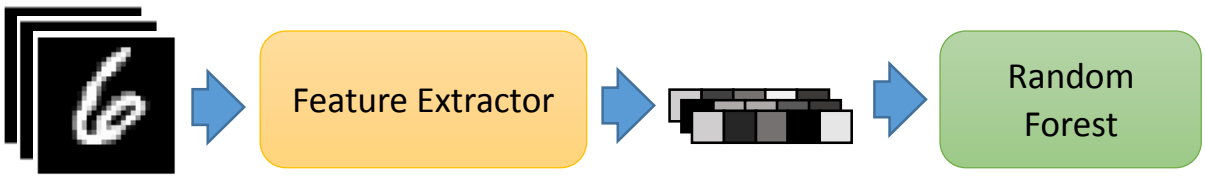
Draw any three decision trees and compute the accuracy of each of them. Show the final accuracy of the random forest model you've trained and explain why the performance doesn't look well?

If you're using the Matlab toolbox, here is the example code:

```
% How many trees do you want in the forest?
nTrees = 100;

% Train the TreeBagger (Decision Forest).
Random_Forest = TreeBagger(nTrees,X_train,T_train,'FBoot',0.5,...
    'MinLeaf',1000, 'Method', 'classification')

% Draw the first decision tree's structure
view(Random_Forest.Trees{1},'Mode','graph')

% Test the trained model
Y_test = str2double(Random_Forest.predict(X_test));
```

2. Feature Extractor

Please use a feature extractor to pre-process the input data and send it into the random forest model which is identical to the task 1. A good choice will make your model much more powerful!



3. Parameters Adjustment

Try to find the relationship between the accuracy and the following parameters:

- The **number of decision trees**
- The **minimum number** of samples **per leaf node**
- The **fraction of samples** to be randomly selected with replacement for constructing each decision tree

Plot the accuracy (or error) curve with respect to the parameters.

What should be uploaded?

- ☐ Your source code with comments.
- ☐ The **ReadMe.txt** file which describes how to run your program.
- ☐ Your **report** in the format of .pdf or .doc.

Reminders:

- ☐ There won't be a need for demonstration.
- ☐ Please make sure your source code can be compiled by **Matlab, Python or C++**.
- ☐ **DO NOT COPY!!!（懶人包、考古題亦同）**