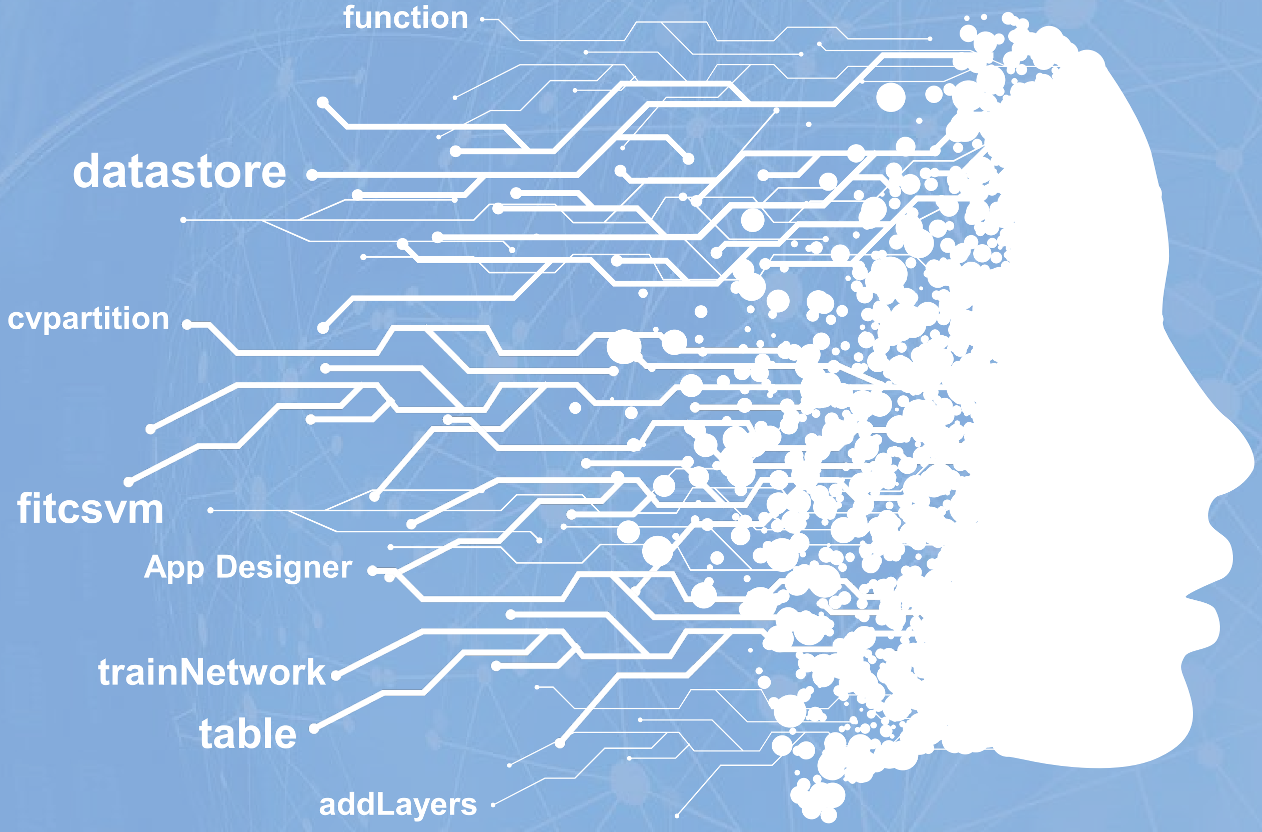


Classification: Decision Tree

MATLAB進階程式語言與實作

盧家鋒 Chia-Feng Lu, Ph.D.
Department of Biomedical Imaging and Radiological Sciences, NYCU
alvin4016@nycu.edu.tw

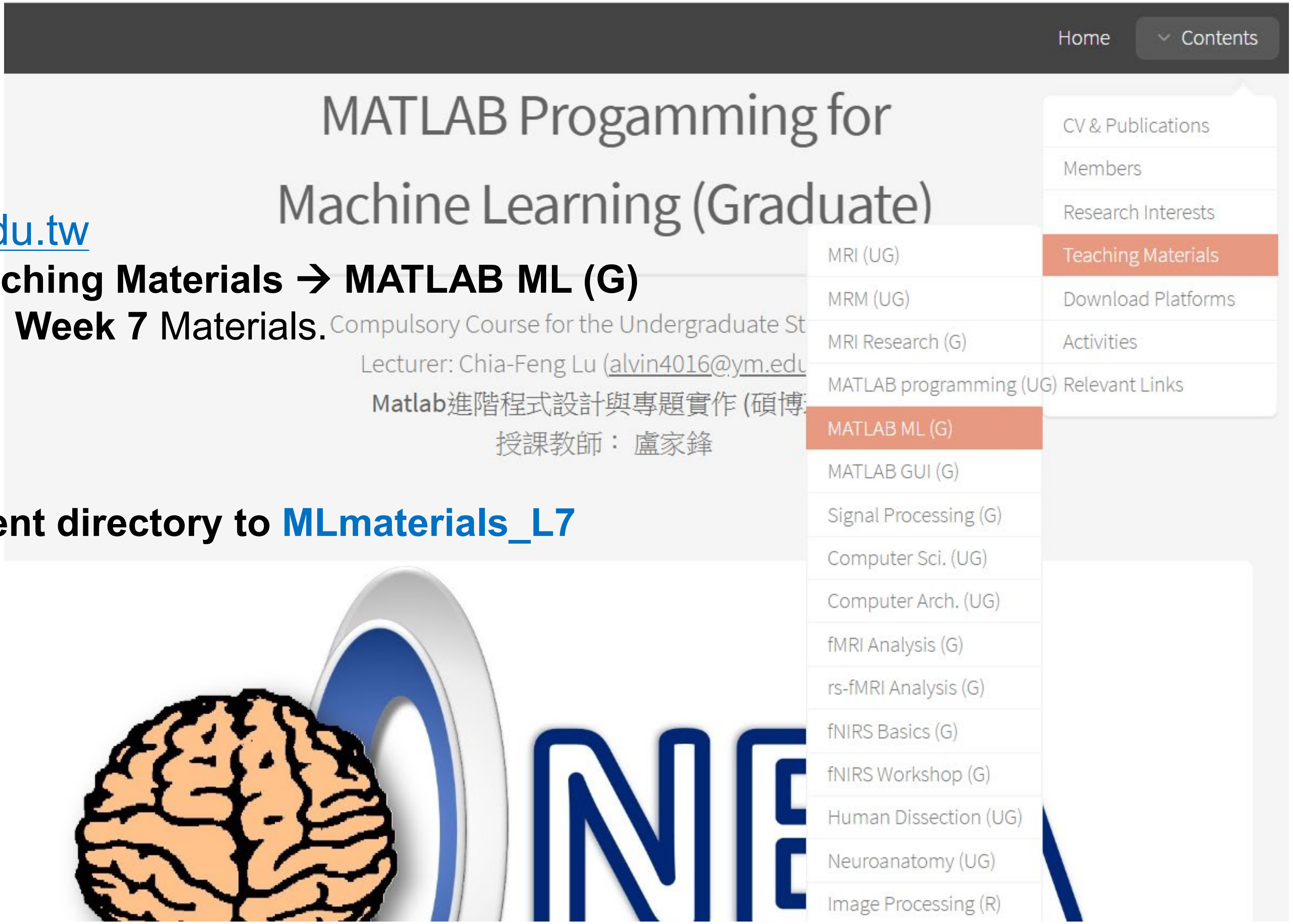
Teaching Materials

cflu.lab.nycu.edu.tw

Contents → Teaching Materials → MATLAB ML (G)

Please download **Week 7** Materials.

Please set current directory to MLmaterials_L7



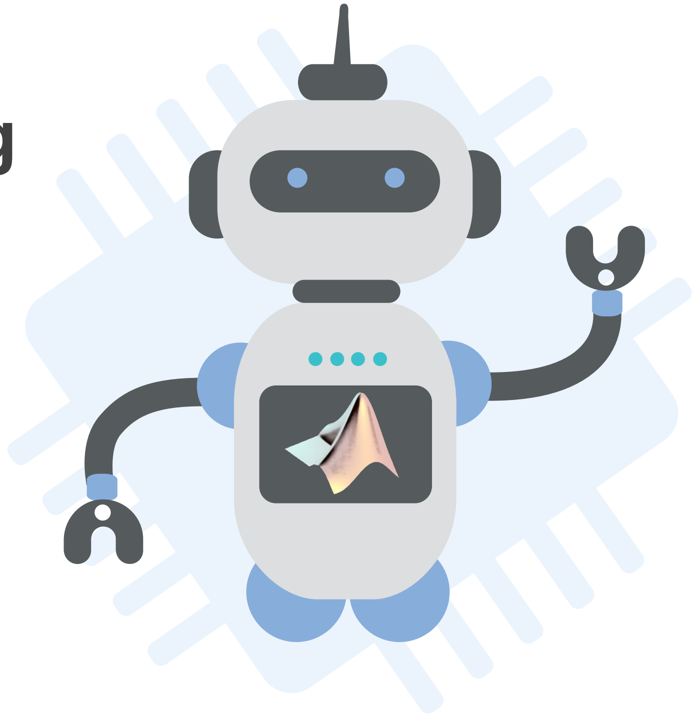
Contents in this Week

01 Classification Trees

Concepts, Pros & Cons

02 Bagging, Random Forests, Boosting

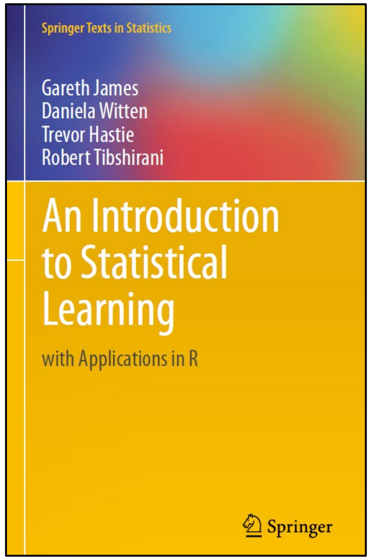
Tree voting, Sequential Learning



References

[Textbook 3]

- **An Introduction to Statistical Learning, 2nd edition, 2013**
  Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani
- **Online resources:** https://github.com/rghan/ISLR
- **Online resources:** https://github.com/JWarmenhoven/ISLR-python
- **Tree-based Methods (Ch.8)**



Classification Trees

Concepts, Pros & Cons

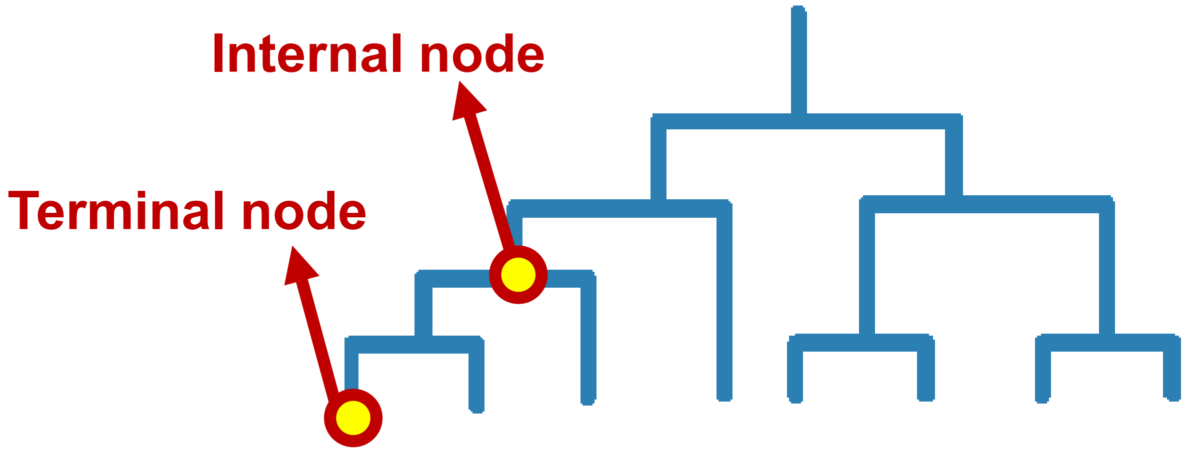
Decision Trees

Classification

- Prediction of a **categorical response**.
- we predict that each observation belongs to **the most commonly occurring class** of training observations in the region to which it belongs.
- The **class proportions** among the training observations that fall into to the same terminal node is of interest.

Regression

- Prediction of a **quantitative value.**
- The prediction is based on **the mean response** of the training observations that belong to the same terminal node.



Algorithm of Decision Tree

1. Use **recursive binary splitting** to grow a large tree on the training data, stopping only when each terminal node has fewer than some minimum number of observations (**'MinLeafSize'**).

Tree growing

2. Apply **cost complexity pruning** to the large tree in order to obtain a sequence of best subtrees, as a function of α (**'PruneAlpha'**).

3. Return the subtree from Step 2 that corresponds to the chosen value of α. (**prune**)

Tree pruning

Decision Trees

Classification

- Prediction of a **categorical response**.

Classification error rate

$$E = 1 - \max_{k}(\hat{p}_{mk})$$

The proportion (probability) of training observations in the *m*th node/subgroup that are from the *k*th class.

Regression

- Prediction of a **quantitative value.**

Residual sum of squares (RSS):

To minimize:

$$\sum_{x_i \in R_1(j,s)} \left(\underset{\text{true}}{y_i} - \underset{\text{estimated}}{\hat{y}_{R_1}}\right)^2 + \sum_{x_i \in R_2(j,s)} \left(y_i - \hat{y}_{R_2}\right)^2$$

Classification error is not sufficiently sensitive for tree-growing.

Decision Trees

Classification

- Prediction of a **categorical response**.

Classification error rate

$$E = 1 - \max_{k}(\hat{p}_{mk})$$

Node purity is typically used to evaluate the quality of splitting.

Node purity (certainty)

(1) Gini index:

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$

(2) Cross entropy:

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$$

Tree Growing

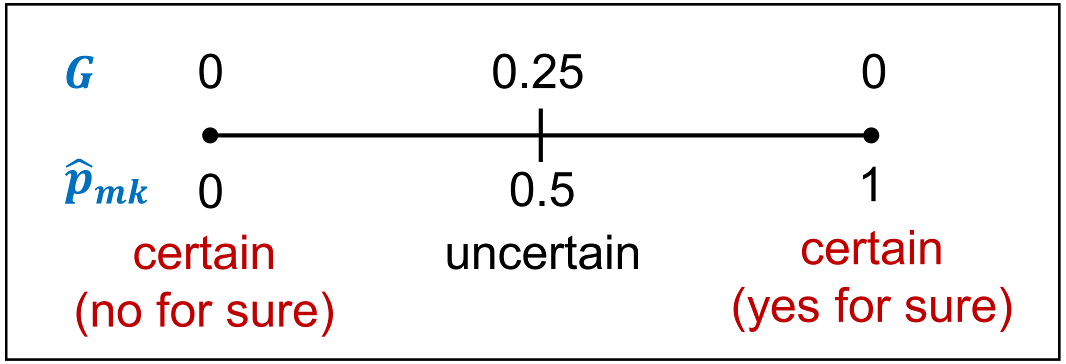
Classification

- Prediction of a **categorical response**.

(1) Gini index:

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$

Gini index is referred to as a measure of node ***purity***—a small Gini value indicates that a node contains predominantly observations from a single class.



Tree Growing

Classification

- Prediction of a **categorical response**.

(2) Cross entropy:

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$$

Like the Gini index, the cross-entropy will take on a small value if the *m*th node is **pure**. In fact, it turns out that the Gini index and the cross-entropy are quite similar numerically.

D	~0	0.35	0
$\hat{p}_{mk}$	0	0.5	1
	certain (no for sure)	uncertain	certain (yes for sure)

Tree Pruning

Classification error rate

$$E = 1 - \max_k(\hat{p}_{mk})$$

Node purity (certainty)

(1) Gini index:

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$

(2) Cross entropy:

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$$

- Any of these three approaches might be used when *pruning* the tree, but the classification error rate is preferable if prediction accuracy of the final pruned tree is the goal.

Heart Dataset Including data from 303 patients.

- **Age**
- **Sex**
- **Chest Pain:** typical angina, atypical angina, non-anginal pain, asymptomatic
- **Rest BP:** resting blood pressure
- **Chol:** serum cholestoral in mg/dl
- **FBS:** fasting blood sugar > 120 mg/dl (1: true; 0: false)
- **Rest ECG:** 0: normal;1: having ST-T wave abnormality; 2: showing probable or definite left ventricular hypertrophy by Estes' criteria
- **Max HR:** maximum heart rate achieved
- **ExAng:** exercise induced angina (1: yes; 0: no)
- **Oldpeak:** ST depression induced by exercise relative to rest
- **Slope:** the slope of the peak exercise ST segment
  1: upsloping; 2: flat; 3: downsloping
- **Ca:** number of major vessels (0-3) colored by flourosopy
- **Thal:** Thalium stress test **(**normal, fixed defect, reversable defect)

AHD (the predicted attribute, diagnosis based on angiography)
(Yes: ≥ 50% diameter narrowing; No: < 50% diameter narrowing;)

MLmaterials_L7\Heart.csv

Exercise – Classification Tree

- Should remove patients with missing data.
- Perform cvpartition to hold out 30% data.

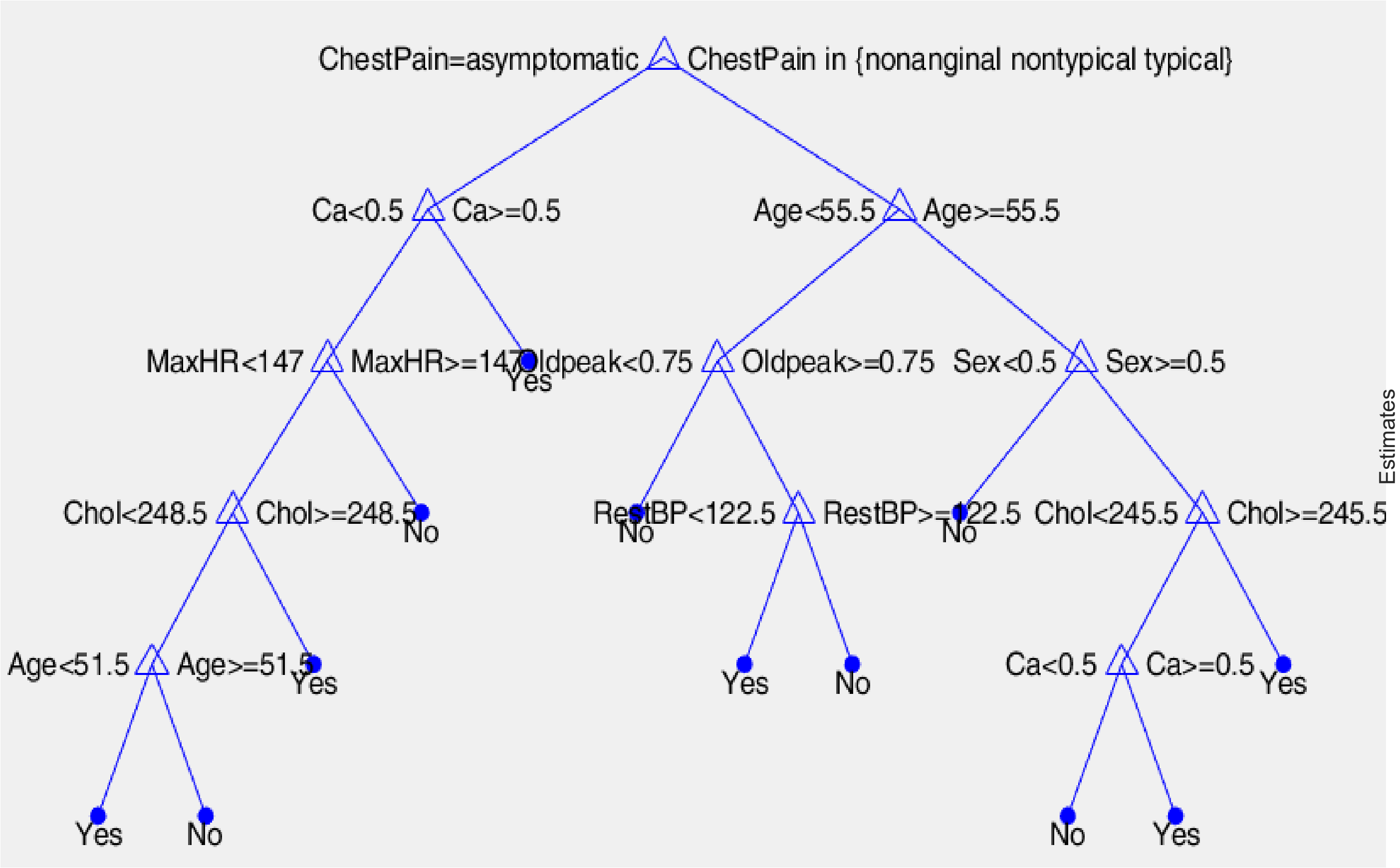
- Predict “AHD" (Yes or No)

```
predictors={'Age', 'Sex', 'ChestPain', 'RestBP', 'Chol', 'Fbs', 'RestECG', 'MaxHR', 'ExAng', 'Oldpeak', 'Slope', 'Ca', 'Thal'};
tree_allv = fitctree(dataTrain,'AHD','PredictorNames',predictors,...
    'SplitCriterion', 'deviance', ...  % 'deviance': cross entropy, 'gdi': Gini index
    'OptimizeHyperparameters','all',...
    'HyperparameterOptimizationOptions',...
    struct('UseParallel',true,'AcquisitionFunctionName','expected-improvement-plus','kfold',5));
```

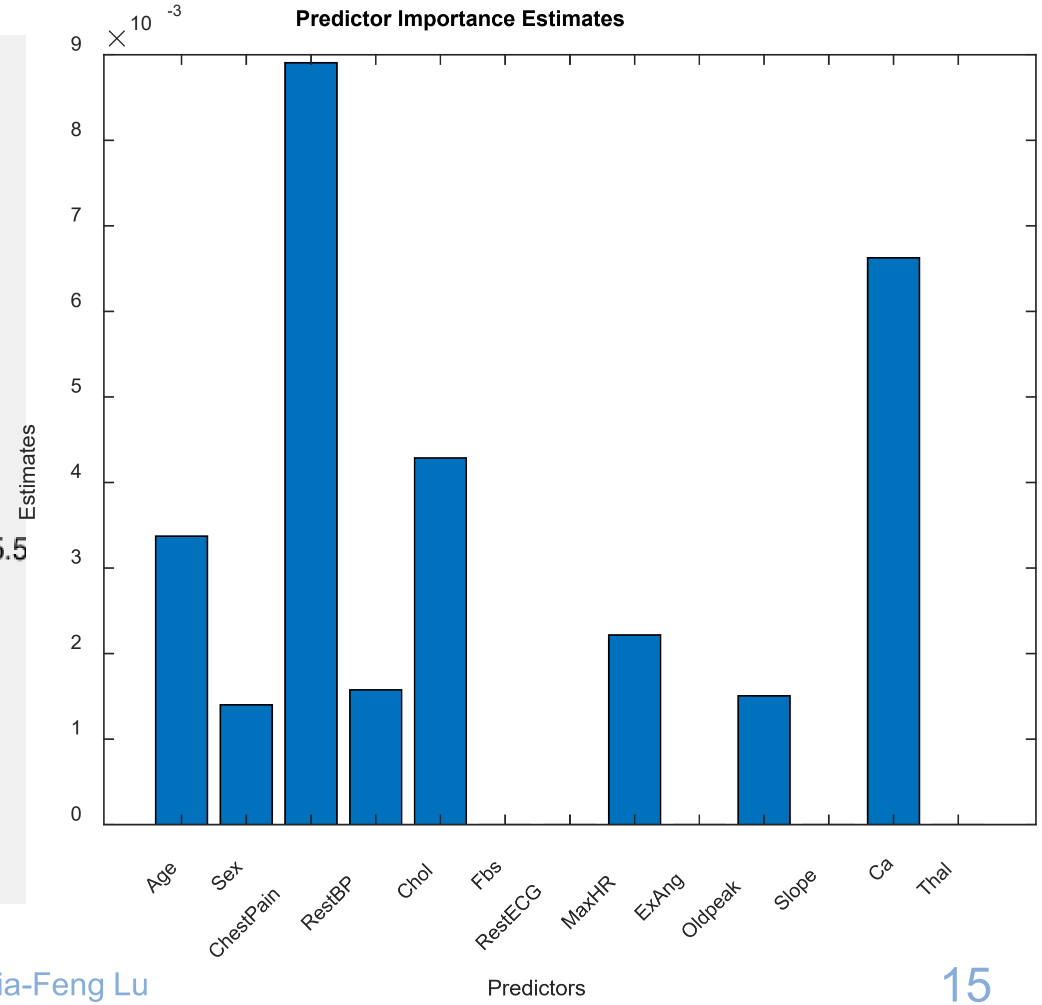
Lines 33 to 38 in MLmaterials_L7\Ex_ClassificationTree.m

Exercise – Classification Tree

view(tree_allv,'Mode','graph')



imp = predictorImportance(tree_allv);



Exercise – Classification Tree

```
AHD_predict=predict(tree_allv,dataTest);
[cm,order] = confusionmat(dataTest.AHD,AHD_predict)
accurarcy = trace(cm)/sum(cm(:))
```

```
cm =

    39     9
    12    29
```

```
accurarcy =

    0.7640
```

Confusion Matrix		True status: Yes	True status: No
Predicted status	Yes	**True Positive** (TP)	False Positive (FP) Type I error
	No	False Negative (FN) Type II error	**True Negative** (TN)

MLmaterials_L7\Ex_ClassificationTree.m

Tree-based Methods

Pros

- Trees are very easy to explain to people. (High interpretability)
- Decision trees are more closely mirror human decision-making strategy than other approaches.
- Trees can be displayed graphically, and are easily interpreted even by a non-expert (especially if they are small).
- Trees can easily handle qualitative predictors without the need to create dummy variables.

Cons

- Unfortunately, trees generally do not have the same level of predictive accuracy as some of the other regression and classification approaches.

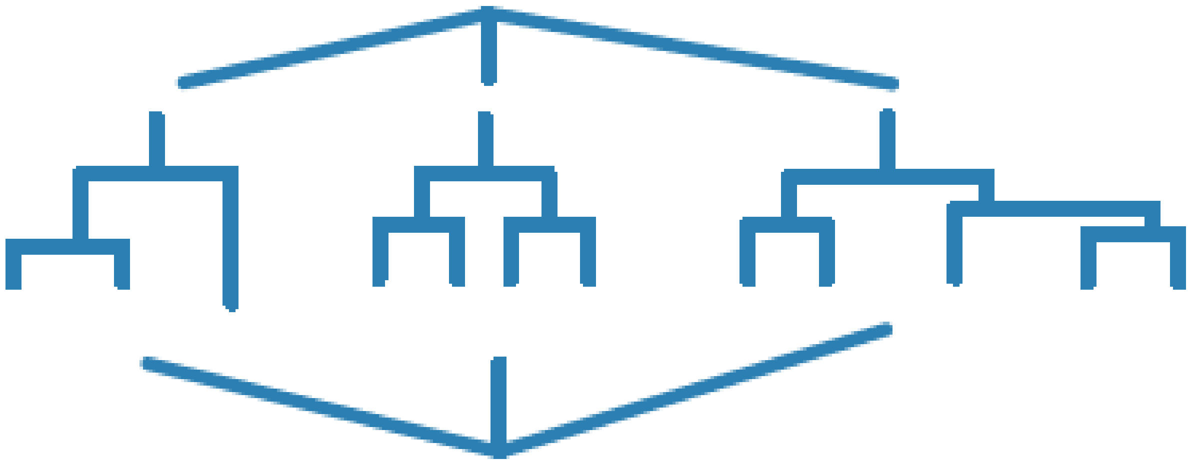
2 Bagging, Random Forests, Boosting

Tree voting and sequential learning

Ensemble Methods of Tree

- Use trees as building blocks to construct more powerful prediction models.
- Several “weaker” decision trees (**weak learner**) are combined into a “stronger” ensemble.

- Bagging
- Random forests
- Boosting



High Variance of Decision Trees

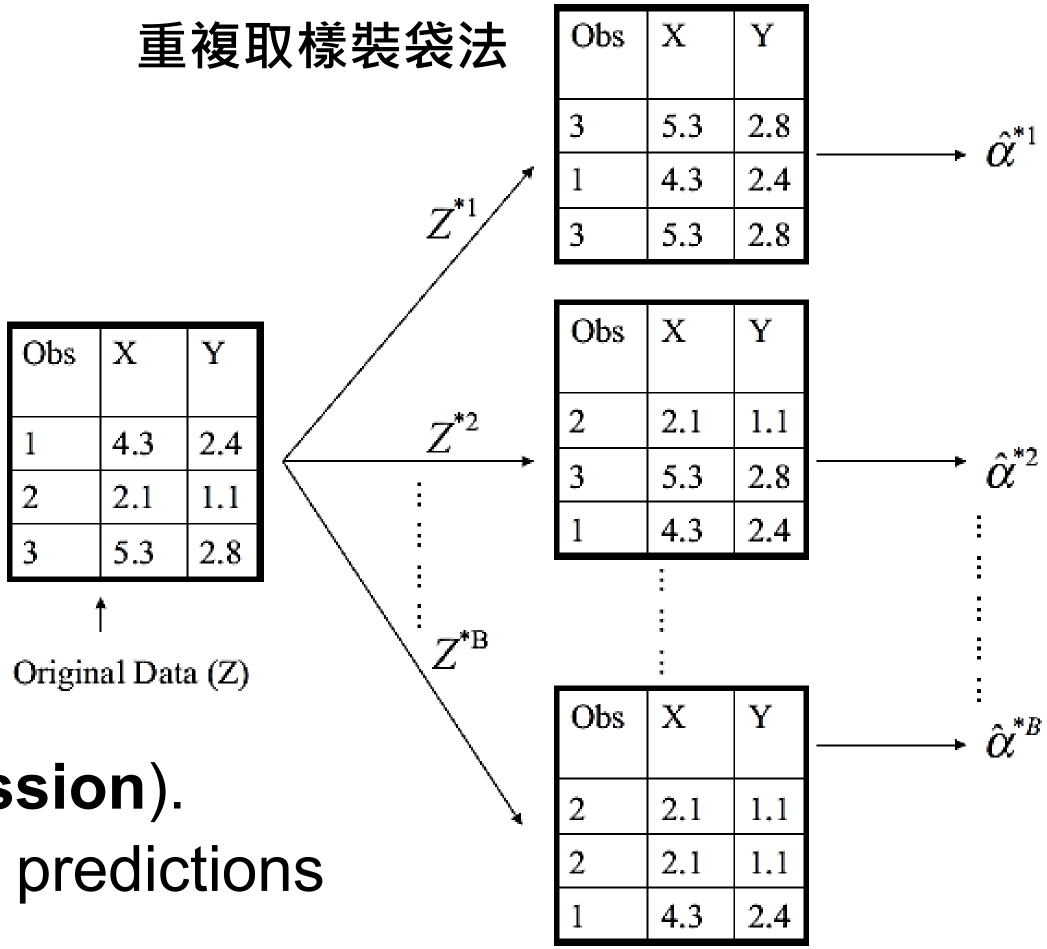
- The decision trees suffer from *high variance*.
  - if we split the training data into two parts at random, and fit a decision tree to both halves, the results that we get could be quite different.

- *Bootstrap aggregation*, or *bagging*, is a general-purpose procedure for reducing the variance of a statistical learning method.
  - To take many training sets from the population, build a separate prediction model using each training set, and average the resulting predictions.

Bagging

- Rather than repeatedly obtaining independent data sets from the population, bootstrap method instead obtains distinct data sets by repeatedly sampling observations from the original (single) data set with replacement.

重複取樣裝袋法



- We then train decision trees on each bootstrapped training set
  - Average all the predictions (**regression**).
  - Take a majority vote among all the predictions (**classification**).

Out-of-Bag Error Estimation

- A very straightforward way to estimate the test error of a bagged model.

- One can show that on average, each bagged tree makes use of around two-thirds of the observations.
- The remaining one-third of the observations not used to fit a given bagged tree are referred to as the out-of-bag (OOB) observations.

- We can predict the response for an observation using each of the trees in which that observation was OOB.

Exercise – Bagging Tree

- Predict “AHD" (Yes or No)

- Should remove patients with missing data.
- Perform cvpartition to hold out 30% data.

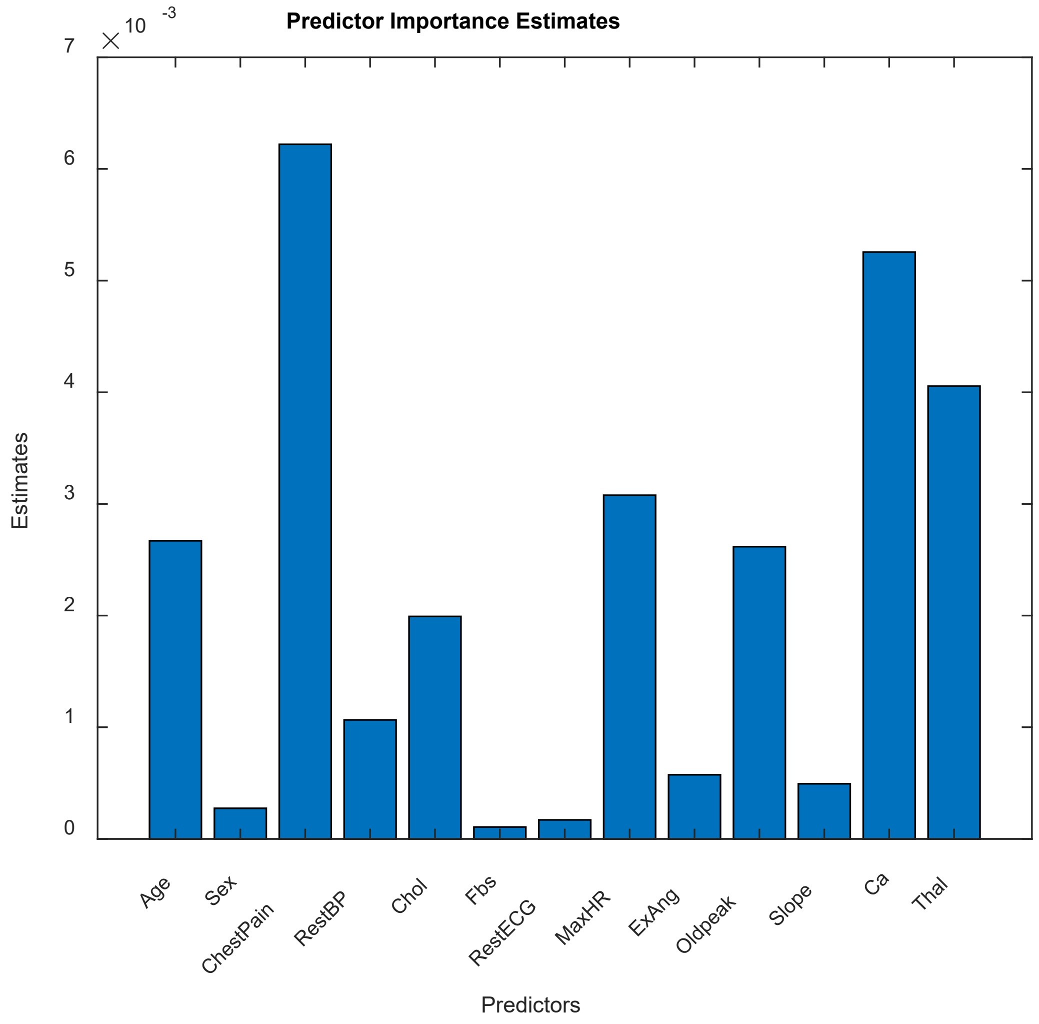
```
B=100;  % the number of bootstrapped/bagged training sets used
t = templateTree('SplitCriterion', 'deviance', 'NumVariablesToSample','all');
tree_bag = fitcensemble(dataTrain,'AHD','PredictorNames',predictors,...
    'Method','Bag',...
    'NumLearningCycles',B,'Learners',t,...
     'OptimizeHyperparameters',{'MinLeafSize','MaxNumSplits'},...
    'HyperparameterOptimizationOptions',...
    struct('UseParallel',true,'AcquisitionFunctionName','expected-improvement-plus','kfold',5));
```

Lines 34 to 42 in MLmaterials_L7\Ex_BaggedTree.m

Variable Importance Measures

- Bagging improves prediction accuracy at the expense of interpretability.
  - It can be difficult to interpret the resulting model.

- Instead, one can obtain an overall summary of the importance of each predictor using the RSS (for bagging regression trees) or the Gini index (for bagging classification trees).



Exercise – Classification Tree

```
AHD_predict=predict(tree_bag,dataTest);
[cm,order] = confusionmat(dataTest.AHD,AHD_predict)
accurarcy = trace(cm)/sum(cm(:))
```

```
cm =

    42     6
    11    30
```

```
accurarcy =

    0.8090
```

Confusion Matrix		True status	
		Yes	No
Predicted status	Yes	**True Positive** (TP)	False Positive (FP) Type I error
	No	False Negative (FN) Type II error	**True Negative** (TN)

MLmaterials_L7\Ex_BaggedTree.m

Random Forests

- Provide an improvement over bagged trees by way of a small tweak that ***decorrelates*** the trees.

- As in bagging, we build a number of decision trees on bootstrapped training samples.
- But when building these decision trees, each time a split in a tree is considered, only a random sample of ***m*** predictors is chosen as split candidates from the full set of ***p*** predictors.
  - Typically we choose $\boldsymbol{m} \approx \sqrt{\boldsymbol{p}}$.

Random Sample: Decorrelation

- The algorithm is *not even allowed* to consider a majority of the available predictors (ex: 4 out of the 13 predictors for the Heart data set).
- This process can avoid a strong predictors dominating the most of bagged trees.

- We can think of this process as *decorrelating the trees*, thereby making the average of the resulting trees less variable and hence more reliable.

Exercise – Bagging Tree

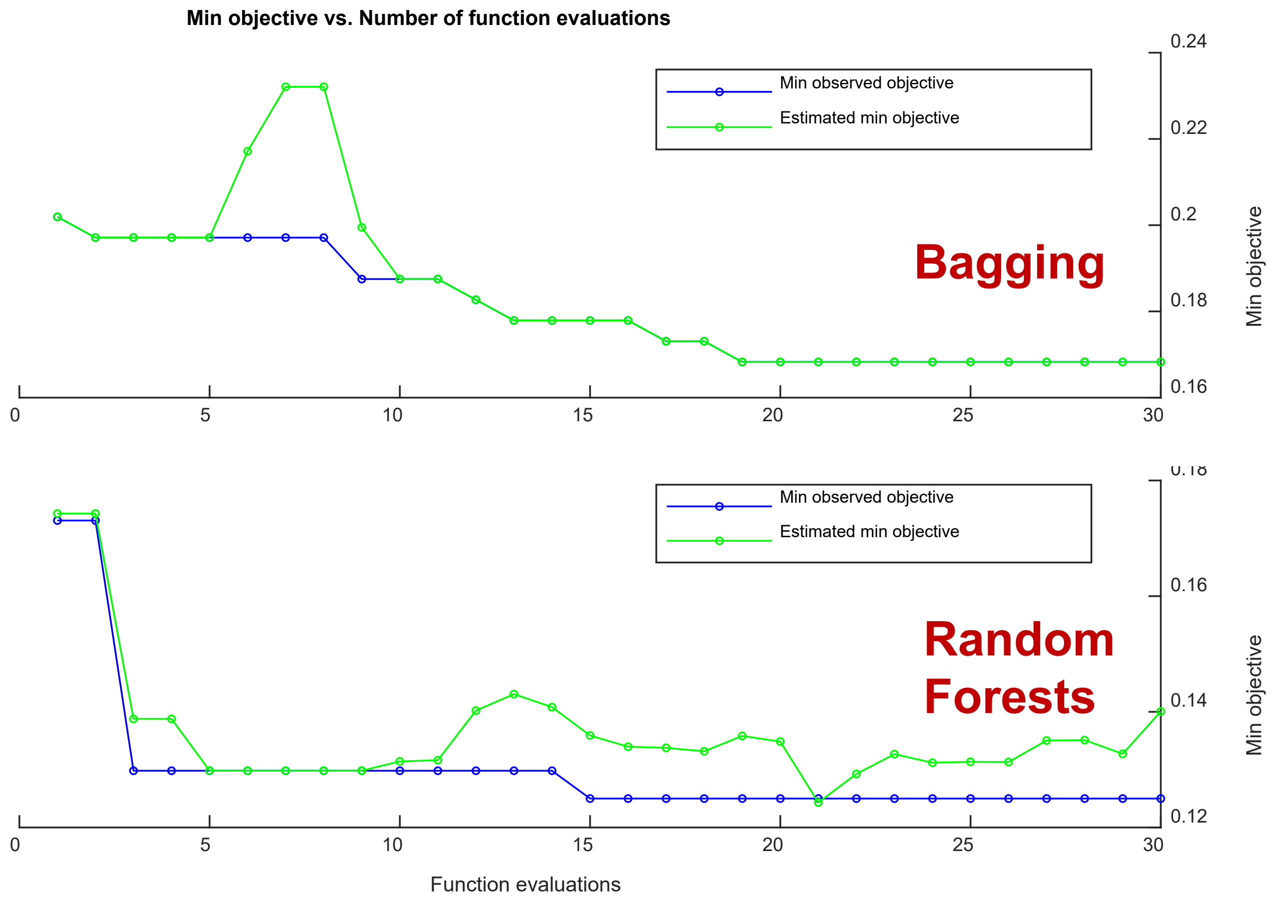
- Predict “AHD" (Yes or No)

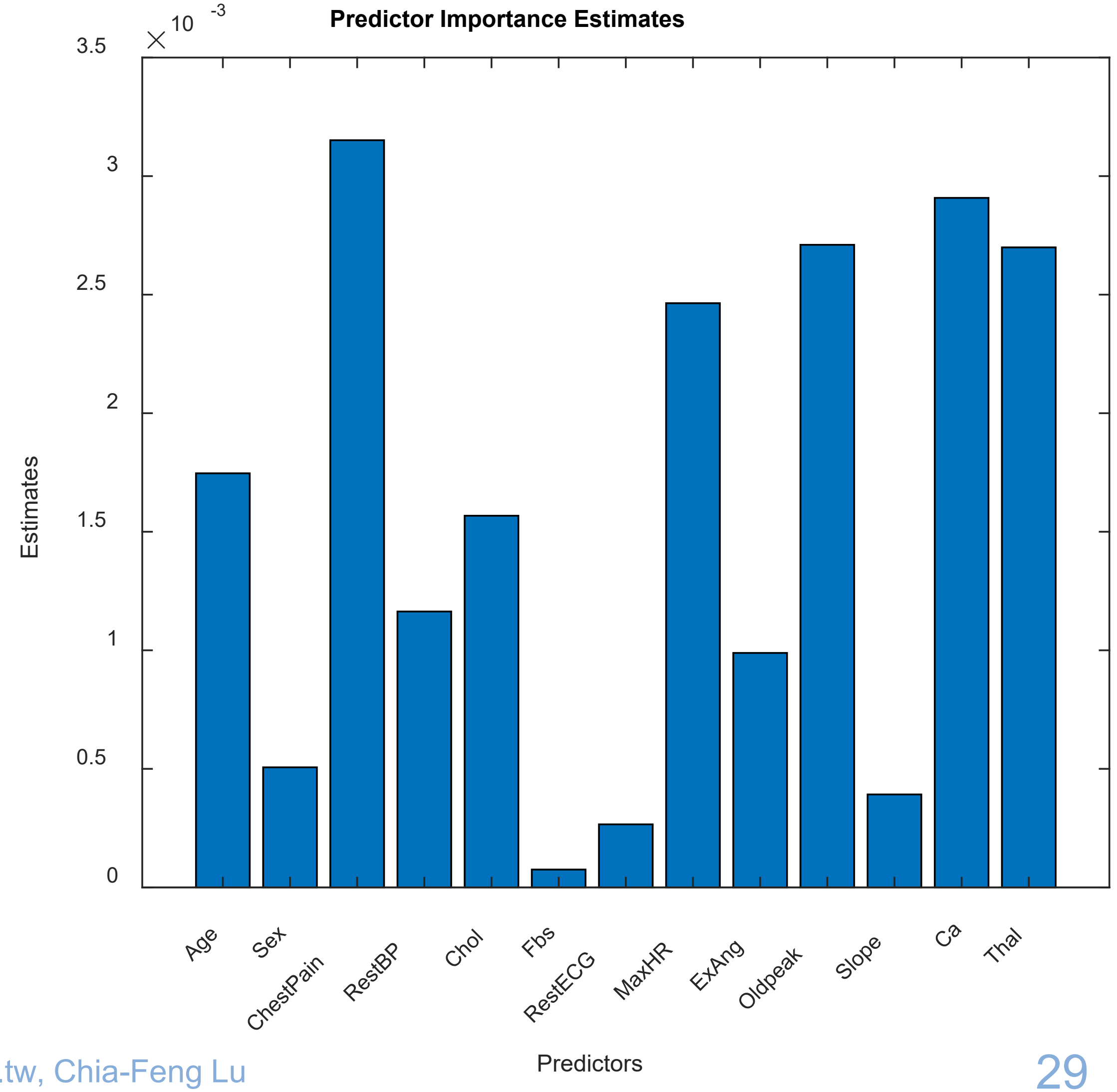
- Should remove patients with missing data.
- Perform cvpartition to hold out 30% data.

```
t = templateTree('SplitCriterion', 'deviance',…
    'NumVariablesToSample', sqrt(size(dataTest,2)-1),…
    'Reproducible',true); % For reproducibility of random predictor selections
tree_RF = fitcensemble(dataTrain,'AHD','PredictorNames',predictors,...
    'Method','Bag',...
    'NumLearningCycles',B,'Learners',t,...
     'OptimizeHyperparameters',{'MinLeafSize','MaxNumSplits'},...
    'HyperparameterOptimizationOptions',...
    struct('UseParallel',true,'AcquisitionFunctionName','expected-improvement-
plus','kfold',5));
```

Lines 66 to 73 in MLmaterials_L7\Ex_BaggedTree.m

Variable Importance Measures





Exercise – Classification Tree

```
AHD_predict=predict(tree_RF,dataTest);
[cm,order] = confusionmat(dataTest.AHD,AHD_predict)
accurarcy = trace(cm)/sum(cm(:))
```

```
cm =

    41     7
    10    31
```

```
accurarcy =

    0.8090
```

Confusion Matrix		True status: Yes	True status: No
Predicted status	Yes	**True Positive** (TP)	False Positive (FP) Type I error
	No	False Negative (FN) Type II error	**True Negative** (TN)

MLmaterials_L7\Ex_BaggedTree.m

Boosting

- Boosting does not involve bootstrap sampling; instead each tree is fit on a modified version of the original data set.
- The trees are grown *sequentially*: each tree is grown using information from previously grown trees.
- The samples that classified incorrectly in the previous tree will be re-examed/emphasized in the next tree.

fitcensemble 'Method'

'AdaBoostM1'	Adaptive boosting
'AdaBoostM2'	Adaptive boosting
'GentleBoost'	Gentle adaptive boosting
'LogitBoost'	Adaptive logistic regression
'LPBoost'	Linear programming boosting — Requires Optimization Toolbox™
'RobustBoost'	Robust boosting — Requires Optimization Toolbox
'RUSBoost'	Random undersampling boosting
'TotalBoost'	Totally corrective boosting — Requires Optimization Toolbox

MLmaterials_L7\Ex_OptimizeEnsembleTree.m

THE END

Contact:
盧家鋒 alvin4016@nycu.edu.tw

Carseats Dataset

Including simulated sales of child car seats at 400 different stores.

- **Sales**
  - Unit sales (in thousands)
- **CompPrice**
  - Price charged by competitor
- **Income**
  - Community income level (in thousands of dollars)
- **Advertising**
  - Local advertising budget for company (in thousands of dollars)
- **Population**
  - Population size in region (in thousands)
- **Price**
  - Price company charges for car seats
- **ShelveLoc**
  - Bad, Good and Medium indicating the quality of the shelving location
- **Age**
  - Average age of the local population
- **Education**
  - Education level at each location
- **Urban**
  - Whether the store is in an urban or rural location
- **US**
  - Whether the store is in the US or not

MLmaterials_L7\Carseats.csv

https://rdrr.io/cran/ISLR/man/Carseats.html