

EE2211 Pre-Tutorial 9

Dr Feng LIN

feng_lin@nus.edu.sg

Agenda

- Recap
- Self-learning
- Tutorial 9

Recap

Module III Contents

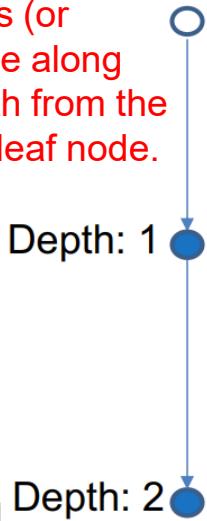
- Overfitting, underfitting and model complexity
- Bias-variance trade-off
- Regularization
- Loss function
- Optimization
- Gradient descent
- Decision Trees
- Random forest

Recap

- Classification tree learning
 - Impurity measures: Gini, Entropy, Misclassification rate
 - For each leaf node, find best feature and threshold to split node to minimize impurity
- Regression tree predicts continuous target
 - Minimize MSE instead of impurity
- Random forest
 - Generate multiple bootstrapped training sets
 - Train on each bootstrapped training set & average

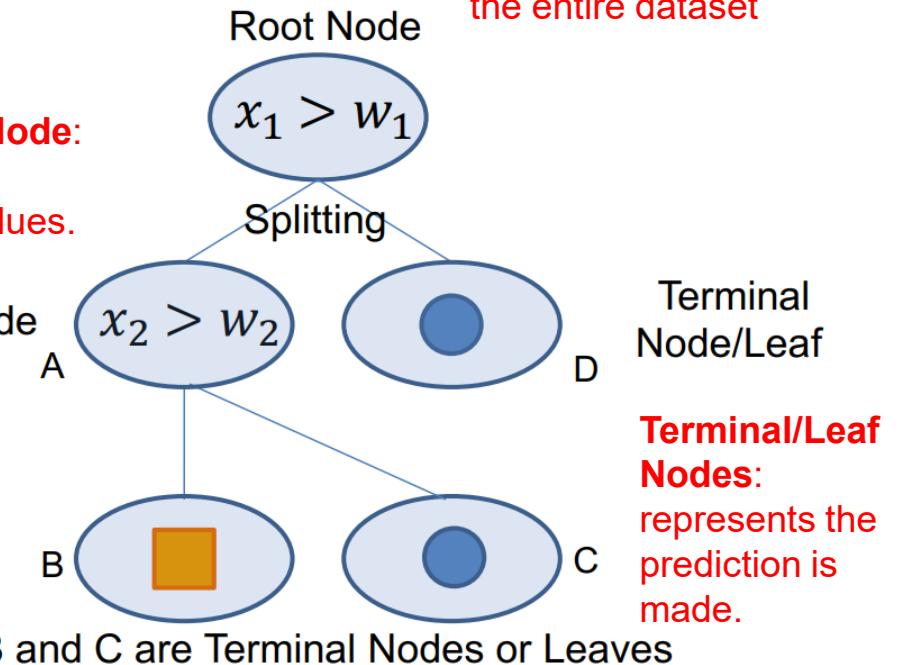
Decision Trees

Depth measures how many decisions (or splits) are made along the longest path from the root node to a leaf node.



Decision/Internal Node: represent decisions based on feature values.

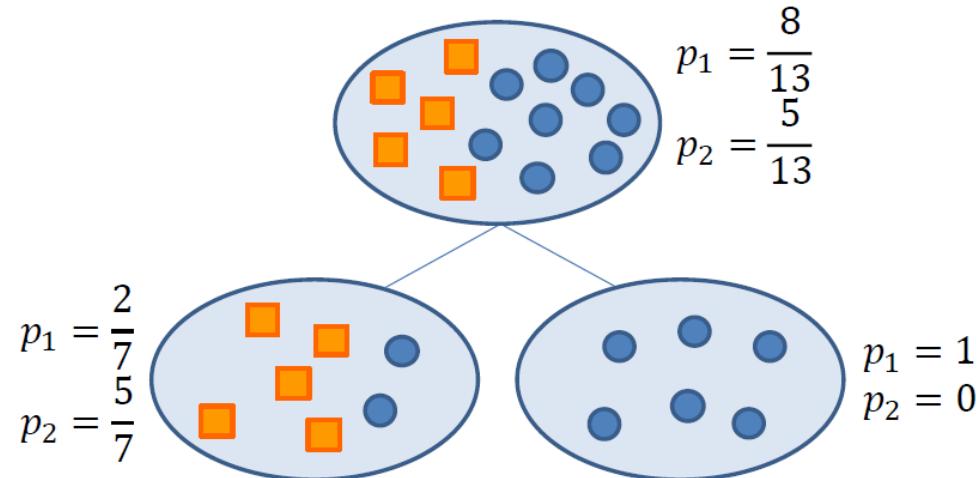
Decision Node



Root Node: represents the entire dataset

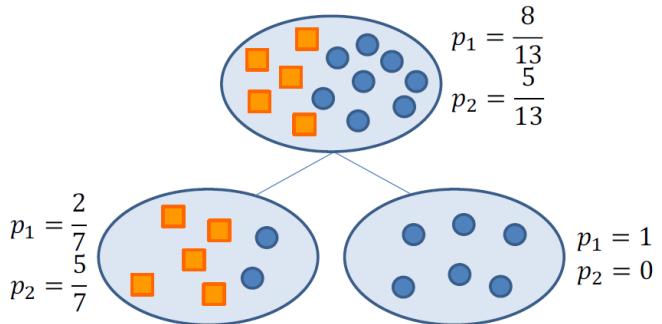
Node Impurity Measures

- Let ● be class 1 & ■ be class 2
- For particular node m , let p_i be the fraction (or probability) of data samples in node m belonging to class i
- Let Q_m be impurity of node m
- 3 impurity measures: Gini, entropy, misclassification rate

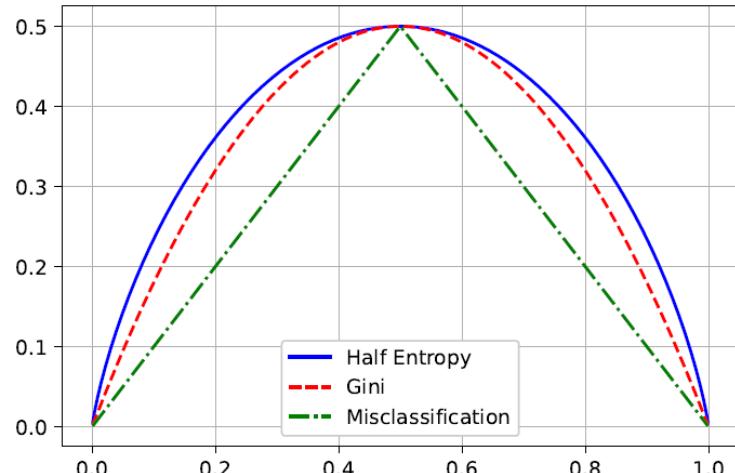


Node Impurity Measures

For particular node m , let p_i be the fraction (or probability) of data samples in node m belonging to class i



- Gini Impurity, $Q_g = 1 - \sum_{i=1}^K p_i^2$
 - Overall gini impurity is $\frac{B}{A} Q_{g,B} + \frac{C}{A} Q_{g,C} \dots$
 - Impurity is lower at lower depth ($Q_{g@2} < Q_{g@1}$)
- Entropy, $Q_e = -\sum_{i=1}^K p_i \log_2 p_i$
 - Entropy always > 0
 - Overall entropy $\frac{B}{A} Q_{e,B} + \frac{C}{A} Q_{e,C} \dots$
- Misclassification rate, $Q_m = 1 - \max(p_1, p_2)$
 - Overall misclassification rate, $\frac{B}{A} Q_{m,B} + \frac{C}{A} Q_{m,C} \dots$



Comparison of impurity measures
for the two-class case

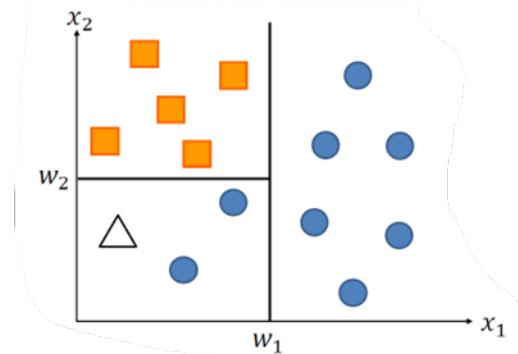
Decision Tree for Classification

Algorithm: Classification Tree Learning

Input: Impurity measure Q , parameter max_depth & training set

Output: Tree

- 1 root \leftarrow all training samples
 - 2 **for** $d \leftarrow 1$ **to** max_depth **do**
 - 3 **for** each leaf node m at depth $d - 1$ **do**
 - 4 Find best feature & best threshold, so splitting node m into two reduces the most impurity
 - 5 Use decision rule to distribute training samples from node m across two new leaf nodes
 - 6 **return** tree
-



Advantages / Disadvantages

Advantages

- Easy to visualize & understand tree
- Can work with a mix of continuous and discrete data
- Less data cleaning required
- Makes less assumptions about the relationship between features & target



- No Need to Normalize or Scale Features
- Tolerant to Missing Values
- Outliers Have Less Impact

Disadvantages

- Trees can become overly complex resulting in overfitting
- Trees can be unstable, e.g., small changes in training data can result very different trees

To reduce overfitting...

- One or more of the following can help reduce overfitting
 - Set maximum depth for the tree
 - Set minimum number of samples for splitting a leaf node, e.g., if leaf node has less than 10 samples, then do not split node
 - Set minimum decrease in impurity, e.g., if selecting the best feature & threshold does not improve impurity by at least 1%, then do not split the leaf node
 - Instead of looking at all features when considering how to split a leaf node, we can randomly look at a subset (e.g., square root of the total number of features)

Regression Trees

- Classification trees seek to predict discrete variables (i.e., classification)
- Regression trees seek to predict **continuous** variables (i.e., regression)
- Can use same approach as before, but instead of minimizing impurity, we can try to **minimize mean square error (MSE)**
- Suppose there are J_m training samples in a leaf node m of the regression tree with target values y_1, y_2, \dots, y_{J_m}
 - We can predict $\hat{y}_m = \frac{1}{J_m} \sum_{j=1}^{J_m} y_j$
 - Then MSE of node m is given by $S_m = \frac{1}{J_m} \sum_{j=1}^{J_m} (y_j - \hat{y}_m)^2$
 - Across all leaf nodes, **total MSE** $S = \sum_m \frac{J_m}{N} S_m$, where N is the total number of data samples

Regression Tree Learning

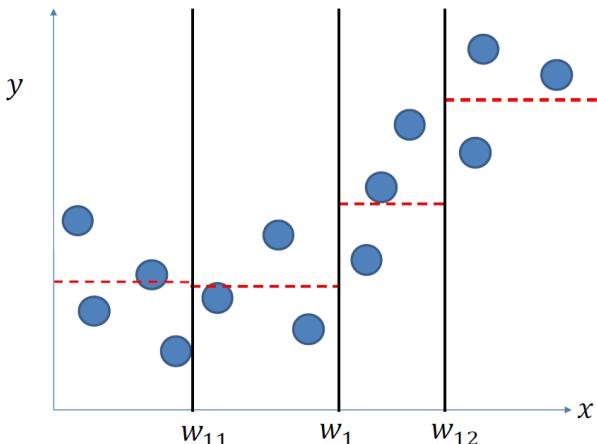
- Algorithm is basically the same as classification tree learning
- Various approaches to reduce overfitting also apply here

Algorithm: Regression Tree Learning

Input: parameter max_depth & training set

Output: Tree

```
1 root  $\leftarrow$  all training samples
2 for  $d \leftarrow 1$  to  $\text{max\_depth}$  do
3   for each leaf node  $m$  at depth  $d - 1$  do
4     Find best feature & best threshold, so splitting
        node  $m$  into two reduces MSE the most
5     Use decision rule to distribute training samples
        from node  $m$  across two new leaf nodes
6 return tree
```



Node impurity and MSE

Classification trees

Gini

$$Q_m = 1 - \sum_{i=1}^K p_i^2$$

Entropy

$$Q_m = - \sum_{i=1}^K p_i \log_2 p_i$$

Misclassification Rate

$$Q_m = 1 - \max(p_i)$$

Regression trees

Mean Squared Error

$$S_m = \frac{1}{J_m} \sum_{j=1}^{J_m} (y_j - \hat{y}_m)^2$$

Example

Example of Regression Tree

- Consider house prices in Singapore.
- Target variable is Price P .
- Attributes are House Size S and Number of Rooms R .

	House Size ('000 sq ft)	Num of Rooms	Price ('000,000 SGD)
1	0.5	2	0.19
2	0.6	1	0.23
3	1.0	3	0.28
4	2.0	5	0.42
5	3.0	4	0.53
6	3.2	6	0.75
7	3.8	7	0.80

- Note that I have arranged the data points in increasing order of P , which so happens to be increasing order of S as well. However, this is not the same order as that of R .

Example

0.75

Calculation of MSE for House Size Split

- Focus first on the House Size attribute S . If we set the threshold at $\tau = 0.75$, then the targets of the two classes are $\{0.19, 0.23\}$ and $\{0.28, 0.42, 0.53, 0.75, 0.80\}$. The individual conditional MSEs are

$$\text{MSE}_{P|S<0.75} = 4 \times 10^{-4} \quad \text{and} \quad \text{MSE}_{P|S \geq 0.75} = 0.0385.$$

- Thus, the averaged conditional MSE with a split of S at 0.75 is

$$\text{MSE}_{P|S(0.75)} = \frac{2}{7}\text{MSE}_{P|S<0.75} + \frac{5}{7}\text{MSE}_{P|S \geq 0.75} = 0.0276.$$

- Sweep through all possible thresholds τ to determine the best threshold for attribute S .

$\text{MSE}_{P S(0.55)}$	$\text{MSE}_{P S(0.75)}$	$\text{MSE}_{P S(1.5)}$	$\text{MSE}_{P S(2.5)}$	$\text{MSE}_{P S(3.1)}$	$\text{MSE}_{P S(3.5)}$
0.0402	0.0276	0.0145	0.0102	0.0116	0.0325

	House Size ('000 sq ft)	Num of Rooms	Price ('000,000 SGD)
1	0.5	2	0.19
2	0.6	1	0.23
3	1.0	3	0.28
4	2.0	5	0.42
5	3.0	4	0.53
6	3.2	6	0.75
7	3.8	7	0.80

$$\hat{y}_{P|S<0.75} = \frac{0.19 + 0.23}{2} = 0.21$$

$$\text{MSE}_{P|S<0.75} = \sum_{j=1}^2 (y_j - \hat{y}_{P|S<0.75})^2 = 4 \times 10^{-4}$$

$$\hat{y}_{P|S>0.75} = \frac{0.28 + 0.42 + 0.53 + 0.75 + 0.80}{5} = 0.556$$

$$\text{MSE}_{P|S>0.75} = \sum_{j=1}^5 (y_j - \hat{y}_{P|S>0.75})^2 = 0.0385$$

$$S_m = \frac{1}{J_m} \sum_{j=1}^{J_m} (y_j - \hat{y}_m)^2$$

$$\text{MSE } S = \sum_m \frac{J_m}{N} S_m$$

Example

	House Size ('000 sq ft)	Num of Rooms	Price ('000,000 SGD)
1	0.5	2	0.19
2	0.6	1	0.23
3	1.0	3	0.28
4	2.0	5	0.42
5	3.0	4	0.53
6	3.2	6	0.75
7	3.8	7	0.80

Calculation of MSE for # Rooms Split

- Rearrange the target variables in order of the house sizes. Doing so we get (0.23, 0.19, 0.28, 0.53, 0.42, 0.75, 0.80). Now we sweep through all possible thresholds τ for R to get the following averaged conditional MSEs.
- We get the following table.

MSE $_{P R(1.5)}$	MSE $_{P R(2.5)}$	MSE $_{P R(3.5)}$	MSE $_{P R(4.5)}$	MSE $_{P R(5.5)}$	MSE $_{P R(6.5)}$
0.0435	0.0276	0.0145	0.0222	0.0116	0.0325

Example

Where is the First Split?

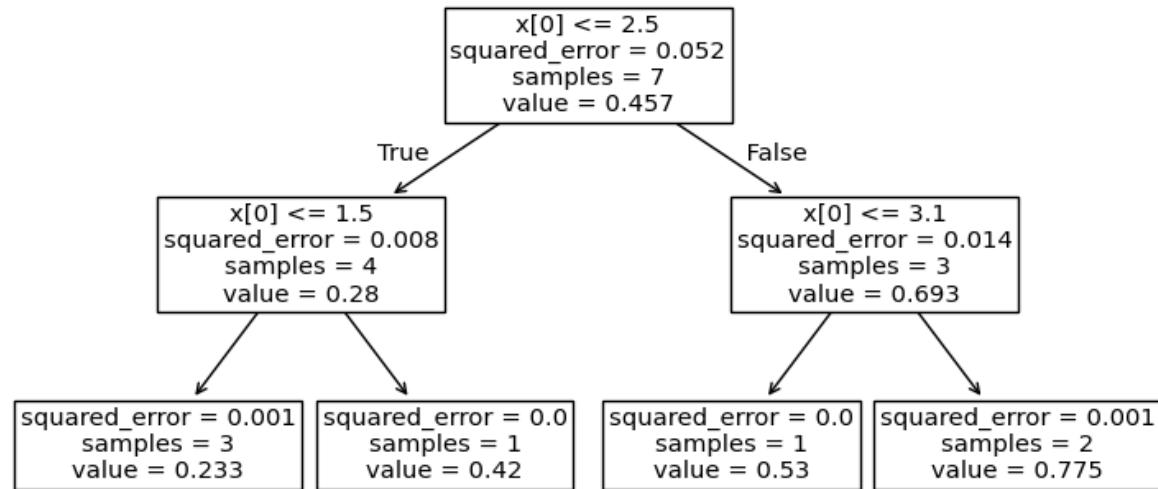
- We should first split the dataset into two branches, the left branch indicating $S < 2.5$ and the right with $S \geq 2.5$.
- Split the dataset into two sub-datasets and we may decide to stop or split the R feature.
- If we decide to stop, then for any new/test house with a house size of < 2.5 , we will predict that its price is the average of the houses in our training set whose size is < 2.5 , i.e.,

$$(0.19 + 0.23 + 0.28 + 0.42)/4 = 0.28.$$

- For a new/test house with a house size of ≥ 2.5 , we will predict that its price is the average of the houses in our training set whose size is ≥ 2.5 , i.e.,

$$(0.53 + 0.75 + 0.80)/3 = 0.6933.$$

Example

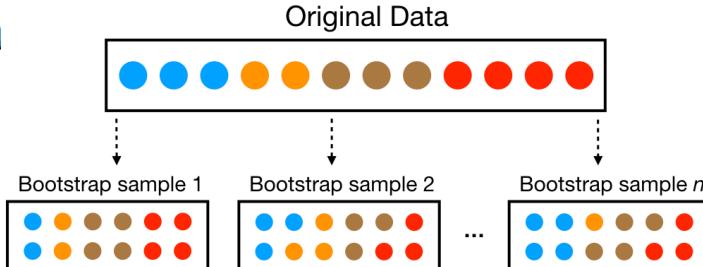


To reduce instability...

- For both classification & regression trees, small perturbations to data can result in very different trees => low bias, high variance
- To reduce variance, we can perturb training set to generate M perturbed training sets
 - Train one tree for each perturbed training set
 - Average predictions across the M trees?
- For example, if we have 100 regression trees (trained from 100 perturbed training sets)
 - Given features x from new test sample, the i -th tree predicts $f_i(x)$
 - Then final prediction is $\frac{1}{100} \sum_{i=1}^{100} f_i(x)$
- For example, if we have 100 classification trees (trained from 100 perturbed training sets)
 - Given features x from new test sample, the i -th tree predicts $g_i(x)$
 - Then final prediction is the most frequent class among 100 predictions $g_1(x), g_2(x), \dots, g_{100}(x)$

Random Forest

- To perturb data, we can apply “bootstrapping” to the training set to create a new “bootstrapped” dataset
- Bootstrapping is procedure in which we sample data with replacement
 - For example, given training set with 3 data samples $\{x_1, y_1\}$, $\{x_2, y_2\}$, $\{x_3, y_3\}$, a bootstrapped training set might comprise sample 1 $\{x_2, y_2\}$, sample 2 $\{x_2, y_2\}$ and sample 3 $\{x_1, y_1\}$
 - Bootstrapped dataset is the same size as original dataset
 - Bootstrapped dataset might contain repeated samples
 - Bootstrapped dataset might not contain some samples from original data



Random Forest

Algorithm: Random Forest Learning

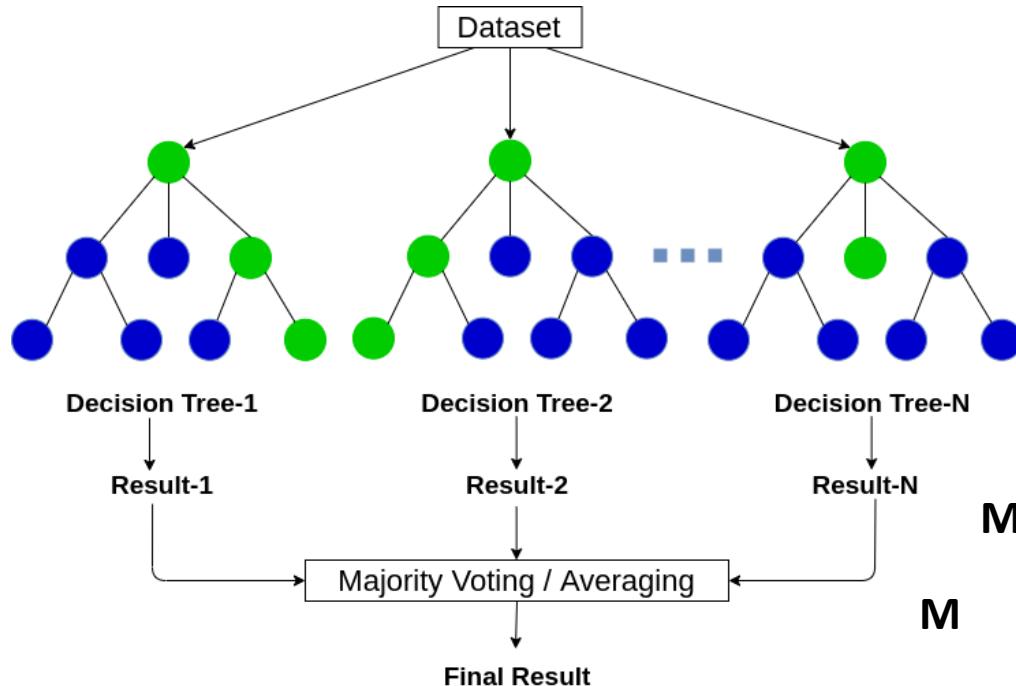
Input: parameter max_trees & N training samples

Output: Forest

```
1 for  $t \leftarrow 1$  to  $max\_trees$  do
2   dataset  $\leftarrow$  bootstrap( $N$  training samples)
3   trees[t]  $\leftarrow$  TreeLearning(dataset)
4 forest  $\leftarrow$  average(trees)
5 return  $forest$ 
```

- To increase randomness, when training the trees, instead of looking at all features when considering how to split a node, we can randomly look at a subset (e.g., square root of the total number of features)

Random Forest



Summary

- Decision tree: series of binary decisions to arrive at prediction of target variable
- Classification tree predicts discrete target class
- Classification tree learning
 - Impurity measures: Gini, Entropy, Misclassification rate
 - For each leaf node, find best feature and threshold to split node to minimize impurity
- Regression tree predicts continuous target
 - Minimize MSE instead of impurity
- Random forest
 - Generate multiple bootstrapped training sets
 - Train on each bootstrapped training set & average



THANK YOU