Financial Data Science

Lecture 6
Decision Tree and Random Forest

Video tutorial:
https://youtu.be/bjiQuZudoJU

Liu Peng
liupeng@smu.edu.sg

Strict Assumptions for Regression Analysis

Assumptions for **Linear Regression**	Assumptions for **Logistic Regression**
• **None** of the input variables is constant • **No perfect linear relationships** among the input variables	• The observations (rows) are independent of each other, and their target outcome follow the same **Bernoulli distribution** • **Little or no collinearity** (i.e., low correlation) among the input variables • No linear relationship between the target y and input variables • **Log odds** of the probability of a target value y being 1 is **linearly** related to input variables

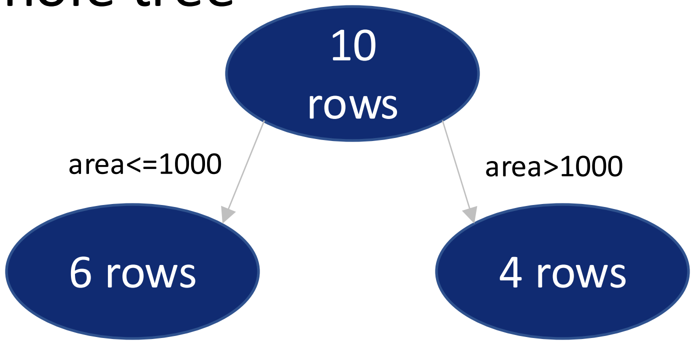
Pros and Cons of Regression

Pros	Cons
• Clear mathematical equations • Training algorithm of regression is easy to implement	▪ Often difficult to determine whether an input variable should become non-linear term ▪ Even harder to determine how to transform an input variable non-linearly to have a good fit for the given data set, i.e., x^2, x^3 or $\log x$ or something else? ▪ Also need to test whether the input variable’s influence on the target variable is significant ▪ Might not converge to a final solution if no good fit can be found with a reasonable no. of iterations

Decision Tree

Generic mechanism of Decision Trees

- Recursively split the data set into branches based on values of input variables
- Maximize reduction of loss function at each split
- End goal is to minimize loss function of the whole tree

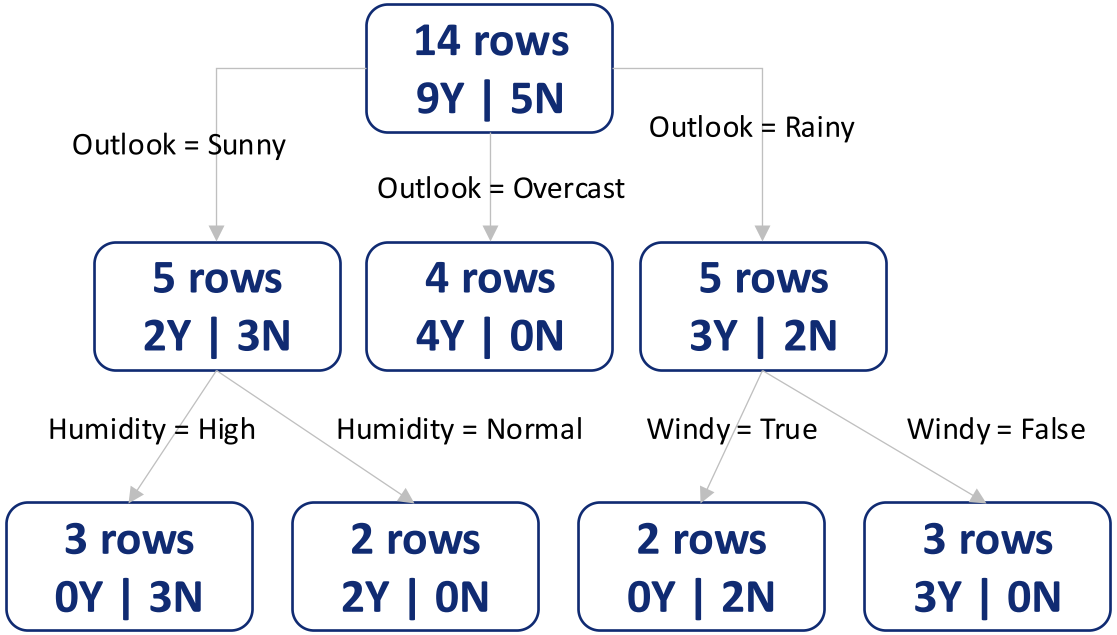


Advantages of Decision Trees

- Can deal with **both regression and classification**
- **No** assumption of the data set's **distribution** explicitly
- **Robust** and do not easily fail to produce a solution
- Able to deal with **large dataset**; the tree aims to split large data set into smaller and homogeneous subsets at each leaf node of the tree

Example: Play tennis based on weather

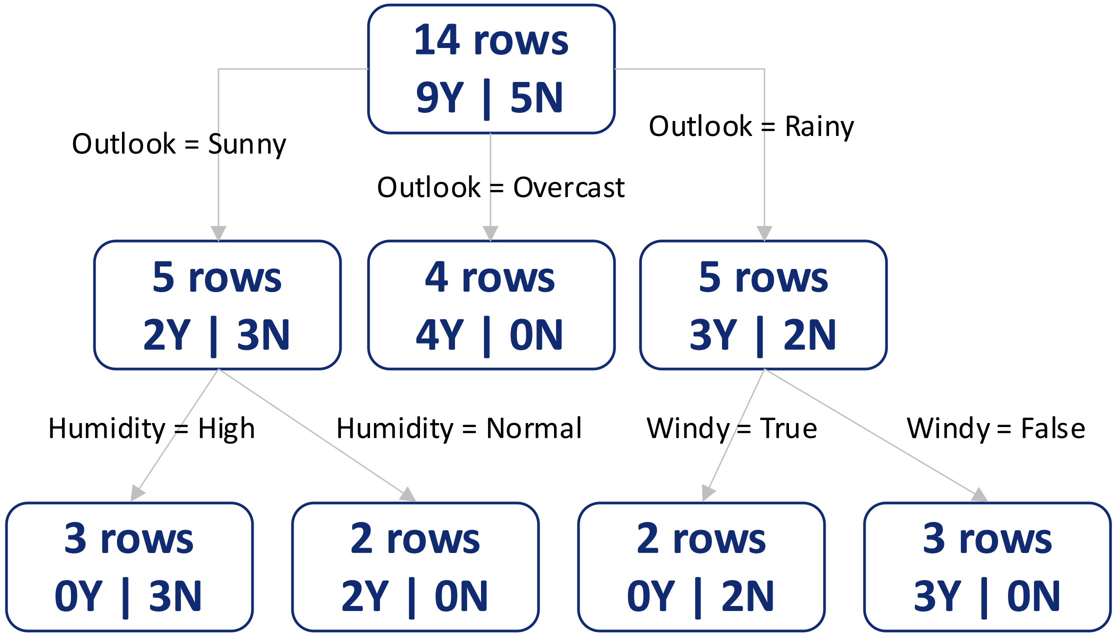
Outlook	Temp	Humidity	Windy	Play
Sunny	Hot	High	FALSE	No
Sunny	Hot	High	TRUE	No
Overcast	Hot	High	FALSE	Yes
Rainy	Mild	High	FALSE	Yes
Rainy	Cool	Normal	FALSE	Yes
Rainy	Cool	Normal	TRUE	No
Overcast	Cool	Normal	TRUE	Yes
Sunny	Mild	High	FALSE	No
Sunny	Cool	Normal	FALSE	Yes
Rainy	Mild	Normal	FALSE	Yes
Sunny	Mild	Normal	TRUE	Yes
Overcast	Mild	High	TRUE	Yes
Overcast	Hot	Normal	FALSE	Yes
Rainy	Mild	High	TRUE	No



Group discussion:

- How do we determine the split decision?
- When do we stop splitting?
- How do we use the model for prediction?

Terminology



- **Root Node**: represents entire data set and can be further divided into smaller subsets
- **Splitting**: dividing a node into two or more sub-nodes
- **Decision Node**: a node that can be split into sub-nodes; the node is **Parent Node** while the sub-nodes are **Child Nodes**
- **Leaf/Terminal Node**: no more splitting
- **Branch/Sub-tree**: a sub section of the entire tree

How do we interpret the tree as rules?
Rule 1: if outlook is sunny and humidity is high, then no tennis
Rule 2: …

Classification And Regression Tree (CART)

CART's overall mechanism

- **Binary split** at each branch
- **Impurity measures**
  - A way to quantify the impurity of nodes; must be reduced maximally at each split
  - A pure node contains observations of identical y (class label or value), and further split is not needed
- Categorical y (classification): CART aims to **minimise misclassification error theoretically**
- Continuous y (regression): CART aims to **minimise sum of squared residuals (SSR)**

CART's advantages

- Input and output variables can be categorical and continuous (**Note**: sklearn implementation of CART can **NOT** support **categorical input** variables)
- Produces a useful tree model with a few important input variables

Overall process of CART

Train-validation-test

1. **Train (Construction)** set: Build the tree
2. **Validation** set: Prune the tree
3. **Test** set: Assess the re-constructed optimal final solution

- Split a variable at all its binary split points
- Select the best split of the variable, i.e., one with highest reduction in impurity
- Rank all variables based on best splits; select the variable that achieves the highest impurity reduction
- Assign observations to the two sub-nodes based on the chosen input variable with chosen split point
- **Repeat above steps for each non-terminal node**

How to split an input variable?

For each categorical input

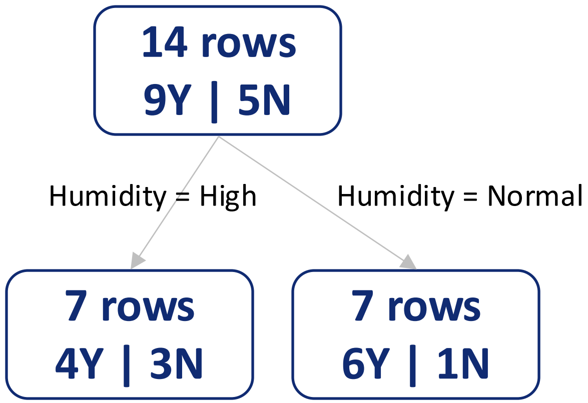
- All possible binary split points are identified
- E.g., x_1 has 5 categories to be split into two buckets; both buckets cannot be empty
- How many splits? $2^5/2 - 1 = 15$ Why?

For each continuous input

- Sort the values in ascending order
- Possible split points = mid-points between pairs of consecutive values
- E.g., continuous input x_2 has values {1, 2, 3, 4, 5, 6}, possible split points: {1.5, 2.5, 3.5, 4.5, 5.5}

- Not too sensitive to outliers or erroneous data
- If one outlier 20 is added, what are the mid-points of {1, 2, 3, 4, 5, 6, 20}? {1.5, 2.5, 3.5, 4.5, 5.5, 13}
- Max mid-point is less influenced by the outlier 20

How to measure the split's impact?



What is the prediction for each node?
Majority wins

- Let's try percentage of misclassification

$$\frac{1}{n}\sum_{i=1}^{n} 1_{y_i \neq pred_i}$$

- Before split
  - root node: 5/14
- After split
  - left leaf: 3/7; right leaf: 1/7; total $= \frac{3}{7} \times \frac{7}{14} + \frac{1}{7} \times \frac{7}{14} = \frac{2}{7}$
- Reduction $= \frac{5}{14} - \frac{2}{7} = \frac{1}{14}$

Issues

- What is the prediction of a node if it has a tie between two values of y?
- Extremely hard problem to optimize
- Computationally costly

Alternative measures to find the best split for each variable

Surrogate loss measures

Categorical output (classification)

- Gini impurity index (not Gini coefficient which measures income/wealth inequality)
- Entropy

Continuous output (regression)

- Sum of squared errors (SSE): L2 loss
- Absolute_error: L1 loss
- Friedman_mse
- Poisson

The best split point for each input…

- Change impure to pure
- Measure the impurity of the nodes before vs. after splitting
- The split point with the highest reduction in the surrogate loss is the best split point
- No splitting if any node is pure

CART (classification): Gini impurity index

- The Gini impurity index for a node A is:

$$I(\mathrm{A}) = 1 - \sum_{c=1}^{k} p_c^2$$

- p_c : the proportion of observations in node A that belong to category *c* of the target variable
- The larger the Gini impurity index is, the more impure the node is

Node A

y=0: 200 | y=1: 800

- Total 1,000 data observations in node A
- I(A) = 1 – [(0.2)^2 + (0.8)^2] = 0.32

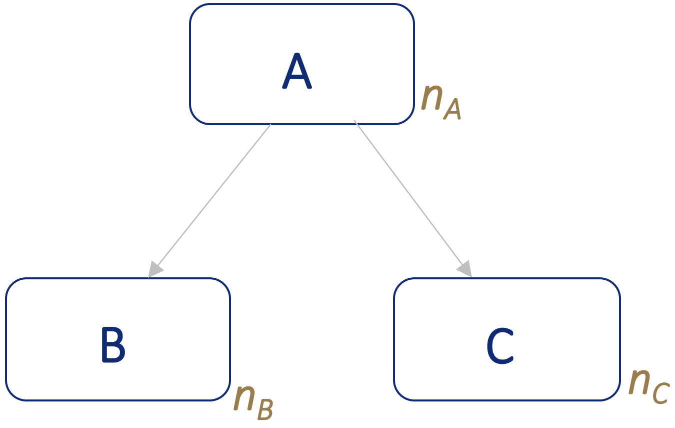
Node B

y=0: 200 | y=1: 500 | y=2: 300

- The categorical target has three values
- I(B) = 1 – [(0.2)^2 + (0.5)^2 + (0.3)^2] = 0.62

Finding the best split point based on reduction of Gini Impurity Index

- Node A is split into node B and node C by a certain condition of an input variable
- Combined Gini impurity index of node B and C:
  - $I(\mathrm{B},\mathrm{C}) = (n_B/n_A)I(\mathrm{B}) + (n_C/n_A)I(\mathrm{C})$
  - n_A, n_B, n_C: no. of observations in the node



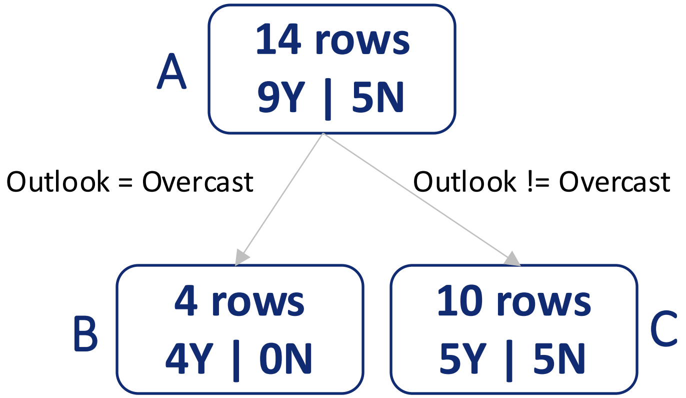
Based on the reduction in Gini impurity indices

$$\Delta I(\mathrm{A},\mathrm{BC}) = I(\mathrm{A}) - I(\mathrm{B},\mathrm{C})$$

A split point is the best if the reduction is the largest among all the possible split points

Exercise: Build a decision tree classifier with Gini as impurity measure

Group discussion: which split is better for ‘Outlook’?

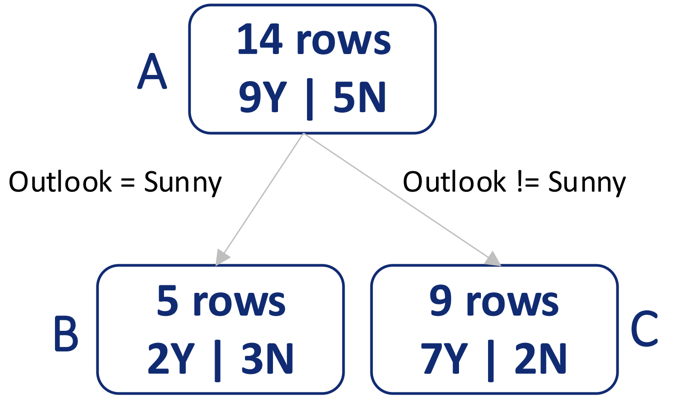


- $I_A = 1 - [(9/14)^2 + (5/14)^2] = 0.459$
- $I_B = 1 - [(4/4)^2 + (0/4)^2] = 0$
- $I_C = 1 - [(5/10)^2 + (5/10)^2] = 0.5$
- $n_A = 14$; $n_B = 4$; $n_C = 10$
- $I_{(B,C)} = (n_B/n_A)I_B + (n_C/n_A)I_C$

$$= \frac{4}{14} \times 0 + \frac{10}{14} \times 0.5$$

$$= \frac{5}{14}$$

- Reduction: $0.459 - 5/14 = 0.102$



- $I_A = 1 - [(9/14)^2 + (5/14)^2] = 0.459$
- $I_B = 1 - [(2/5)^2 + (3/5)^2] = 0.48$
- $I_C = 1 - [(7/9)^2 + (2/9)^2] = 0.346$
- $n_A = 14$; $n_B = 5$; $n_C = 9$
- $I_{(B,C)} = (n_B/n_A)I_B + (n_C/n_A)I_C$

$$= \frac{5}{14} \times 0.48 + \frac{9}{14} \times 0.346$$

$$= 0.39$$

Reduction: $0.459 - 0.39 = 0.069$

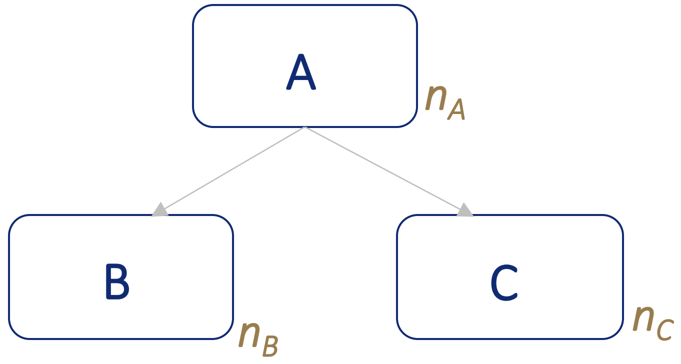
CART (classification): Entropy & Information Gain

- Measures the amount of uncertainty in a dataset
- The entropy value of a node A is:

Recall BCE in logistic regression

$$E(\mathrm{A}) = -\sum_{c=1}^{k} p_C \log_2(p_C)$$

- Two classes in y : $-p_0 \log_2(p_0) - p_1 \log_2(p_1)$
- More classes in y : $-p_0 \log_2(p_0) - p_1 \log_2(p_1) - p_2 \log_2(p_2) - \cdots$
- Lowest Entropy? $p_{y=i} = 1$ and $p_{y=others} = 0 \Rightarrow E = 0$
- Entropy = 0 → no uncertainty → purest
- Highest Entropy for binary y? $p_0 = p_1 = 0.5 \Rightarrow E = 1$
- High entropy → large degree of impurity



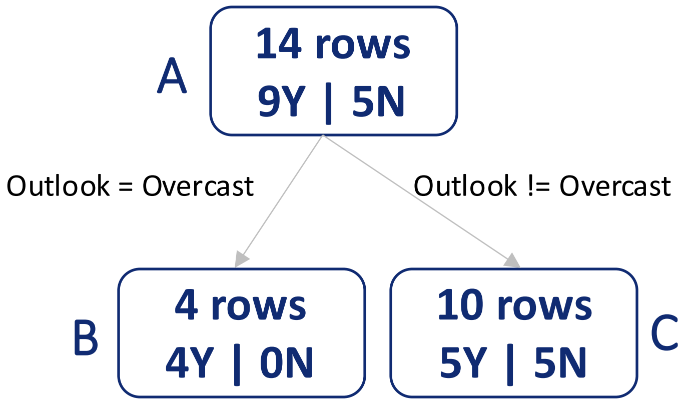
Like Gini, based on the reduction in Entropy, a.k.a. **Information Gain**

$$\begin{aligned} &\Delta E(\mathrm{A}, \mathrm{BC}) \\ &= E(\mathrm{A}) - E(\mathrm{B}, \mathrm{C}) \\ &= E(\mathrm{A}) - [(n_B/n_A)E(\mathrm{B}) + (n_C/n_A)E(\mathrm{C})] \end{aligned}$$

A split point is the best if the **Information Gain** is the largest among all the possible split points

More Reading: https://stackoverflow.com/questions/1859554/what-is-entropy-and-information-gain

Example: Information Gain on a binary split for 'Outlook'



$$\mathrm{E(A)} = -(\frac{9}{14}\log_2\frac{9}{14}) - (\frac{5}{14}\log_2\frac{5}{14}) = 0.94$$

$$\mathrm{E(B)} = -(\frac{4}{4}\log_2\frac{4}{4}) - (\frac{0}{5}\log_2\frac{0}{5}) = 0$$

$$\mathrm{E(C)} = -(\frac{5}{10}\log_2\frac{5}{10}) - (\frac{5}{10}\log_2\frac{5}{10}) = 1$$

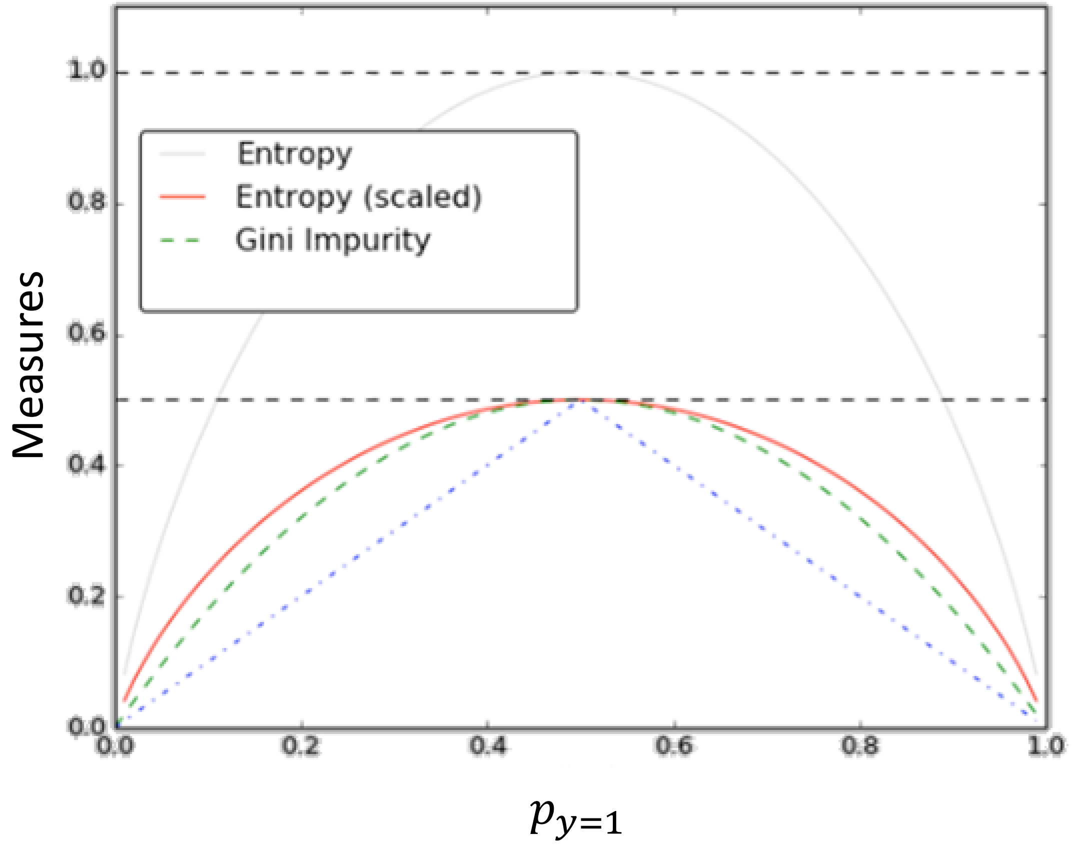
$$E(\mathrm{A}) - [(n_B/n_A)E(\mathrm{B}) + (n_C/n_A)E(\mathrm{C})]$$

$$= \mathrm{E(A)} - [\frac{4}{14}\mathrm{E(B)} + \frac{10}{14}\mathrm{E(C)}]$$

$$= 0.94 - \frac{10}{14}$$

$$= 0.226$$

Difference: Gini impurity index vs. Entropy



- Left plot is for binary y, impurity measures against $p_{y=1} \in [0,1]$
- Both have similar shapes
- Low value indicates low impurity
- Not much difference in terms of what split point will be selected
- Gini is faster computationally than Entropy

Exercise: Build a decision tree classifier with entropy as impurity measure

CART (Regression): sum of squared errors (SSE)

- For **each mid-point** of an input x_i, we have a potential split from node A to nodes B and C
- We calculate the **sum of squared errors** (SSE) of node B and C

$$SSE = \sum [y_j^B - \bar{y}^B]^2 + \sum [y_k^C - \bar{y}^C]^2$$

- y_j^B: the target variable's values in node B
- $\bar{y}^B$: the target variable's mean in node B
- Similarly for C

The **mid-point** with **the smallest SSE** is the best split point for the current input x_i

What is the goal? Maximize reduction in variance of y after the split

CART step 2: Pruning

Q: why do we need pruning?

Train-validation-test

1 **Train (Construction)** set: Build the tree

2 **Validation** set: Prune the tree

3 **Test** set: Assess the re-constructed optimal final solution

- After constructing an exhaustively large tree, we prune away nodes which do not improve overall prediction performance
- Cost-complexity pruning is based on the function of the tree's misclassification rate plus a penalty for the no. of leaf nodes

  $$CC(T) = MisRate(T) + \alpha \times Leaf(T)$$

- The function increases as the size of tree increases
- Sklearn library's Decision Tree implementation does not perform pruning by default; if pruning is enabled, the model takes a part of training set as validation set automatically

Evolution of Decision Tree

ID3 (Iterative Dichotomiser 3)

- Simple decision tree algorithm invented by Ross Quinlan in 1986
- Uses entropy and information gain for splitting
- Chooses the input with the highest reduction in Entropy, i.e., highest information gain
- Handles categorical input variables only
- Continuous input variables are binned as categorical inputs; hence **loss of information**
- Favors inputs with many potential splits; hence could be **overfitting**

C4.5

- Improvement from ID3
- Handles both categorical and continuous inputs
- Applies **Pruning**
- Tree converted to **rule sets**

CART (Similar to C4.5)

- Available in most Analytics tools; usually supports categorical & continuous input
- However, sklearn implementation cannot handle categorical input

C5.0

- Improvement from C4.5
- **Proprietary license** of Ross Quinlan; details of the algorithm are not published
- Uses **information gain ratio** to avoid choosing inputs with many split points, i.e., many values; **avoids overfitting**; constructs smaller trees
- Processes faster; handles large data set
- Includes boosting to improve predictive accuracy (to be covered in 'Ensemble' topic)

C5.0: Information Gain Ratio to avoid choosing inputs with many split points (optional)

- Bias of ID3 and C4.5: Information gain tends to choose input variables with many split points, i.e., many values
- C5.0 uses information gain ratio to address this problem
- The information gain ratio of input X for splitting a node A:

$$InfoGain(\mathrm{A}, X) \;/\; SplitInfo(X)$$

$$SplitInfo(X) = -\sum_{i=1}^{r} \frac{n_{iA}}{n_A} \log_2\left(\frac{n_{iA}}{n_A}\right)$$

  - n_{iA}: the number of observations for value i in X
  - n_A: the total number of observations in node A
- Input with many values have a bigger split information; hence gain ratio is penalized

Example

Information Gain Ratio if we choose 'Outlook' as the first split variable (C5.0 is not limited to binary split)

Outlook	Yes	No	Count
Sunny	2	3	5
Overcast	4	0	4
Rainy	3	2	5

$InfoGain(A,X) = 0.247$

$SplitInfo(X)$

$$= -\left(\frac{5}{14}\log_2\frac{5}{14}\right) - \left(\frac{4}{14}\log_2\frac{4}{14}\right) - \left(\frac{5}{14}\log_2\frac{5}{14}\right)$$

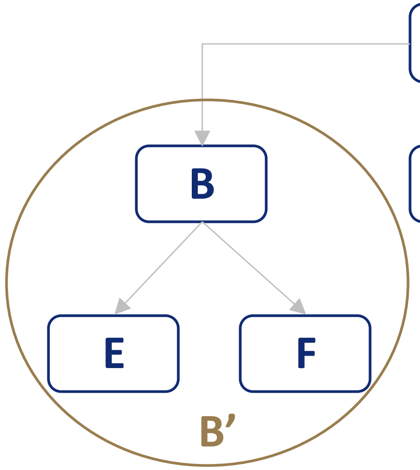
$$= 1.577$$

$$Gain\ Ratio = \frac{0.247}{1.577} = 0.1566$$

C5.0: Two pruning strategies (optional)

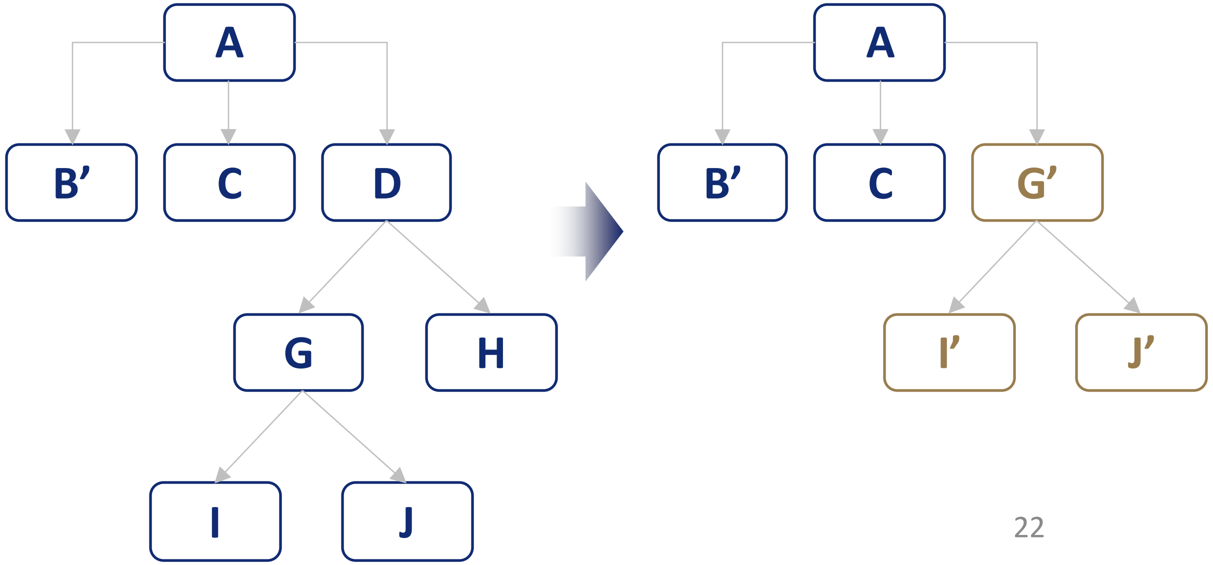
Subtree replacement

- Replace entire subtree B with a leaf node B'
- If overall error of the leaf node B' is close to that of the subtree B, i.e., sum of error for E and F



Subtree raising

- Replace a subtree D by its most used subtree G
- Raise Subtree G from its current location to a higher node
- Merge observations inside H into G' and redistributed between I' and J'



Recap of Decision Tree

- Tree structure:
  - Easy to interpret into rule sets
  - Easy for business to understand and implement
  - No need to think about nonlinear and interaction terms in regression

- Less data preprocessing compared to linear and logistic regression:
  - No standardization
  - No dummy variables
  - No $\log$ transformation

Exercise: Build a decision tree regressor

Random Forest

- Build multiple decision trees that serve as sub-models
- Use random sampling (both observation and features) to approximate independent and uncorrelated trees; the bootstrap step
- Use ensembling to make a final decision based on multiple trees; the aggregation step
- A random forest thus involves the bagging process
- More advanced boosting techniques exist, such as adaptive boosting in XgBoost, etc.

Coding session