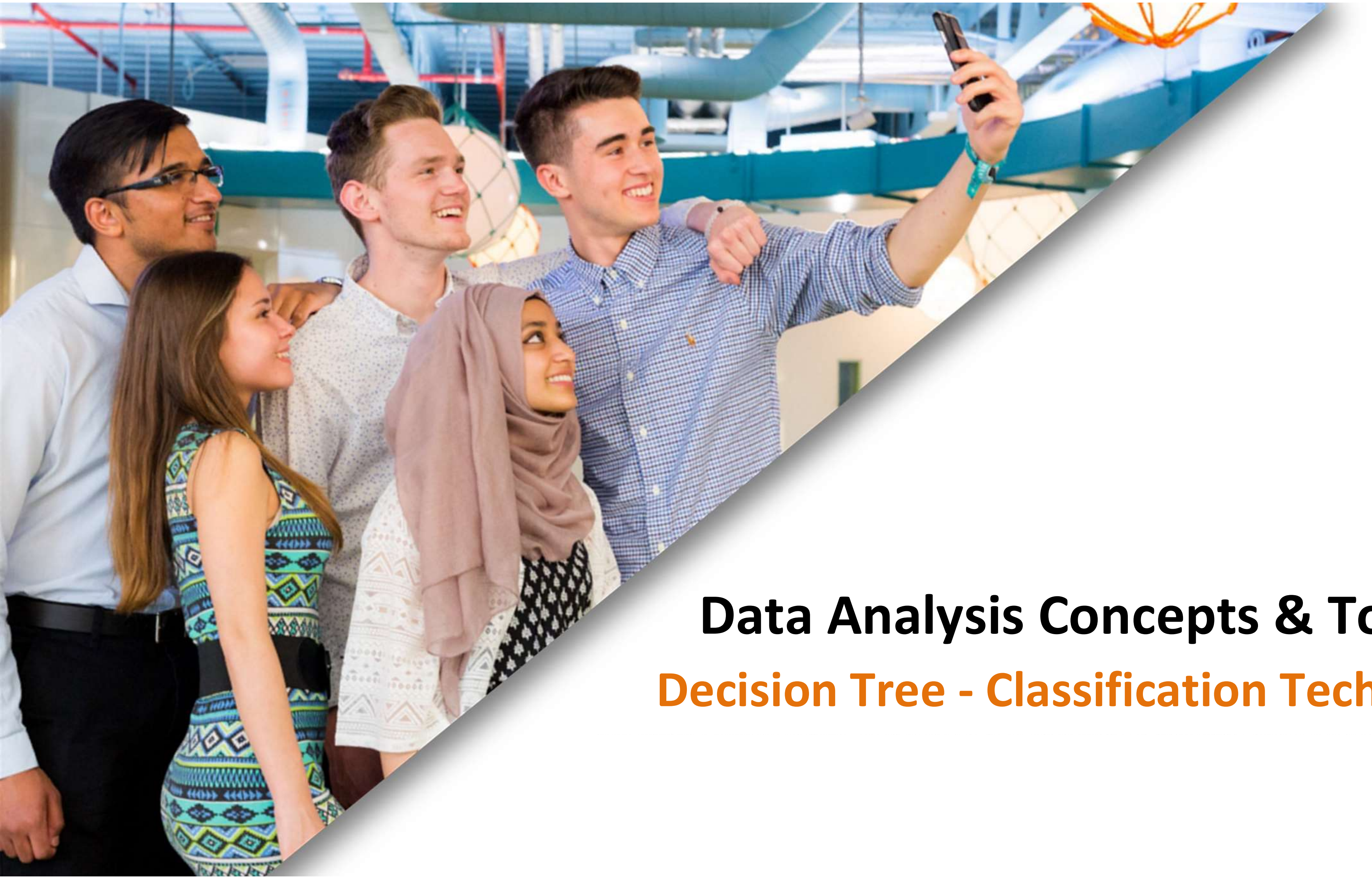


Data Analysis Concepts & Tools 2

Decision Tree - Classification Technique

Decision Tree - Classification Technique

- There are two types of trees; Classification Trees and Regression (or Prediction) Trees.
- **Classification Trees** – are used to segment observations into more homogenous groups (assign class labels). They usually apply to outcomes that are binary or categorical in nature.
- **Regression Trees** – are variations of regression and what is returned in each node is the average value at each node. Regression trees can be applied to outcomes that are continuous (like account spend or personal income).
- The input values can be continuous or discrete.
- Decision Tree models output a tree that describes the decision flow. The leaf nodes return class labels and, in some implementations, they also return the probability scores. In theory the tree can be converted into decision rules.
- Decision Trees are a popular method because they can be applied to a variety of situations.

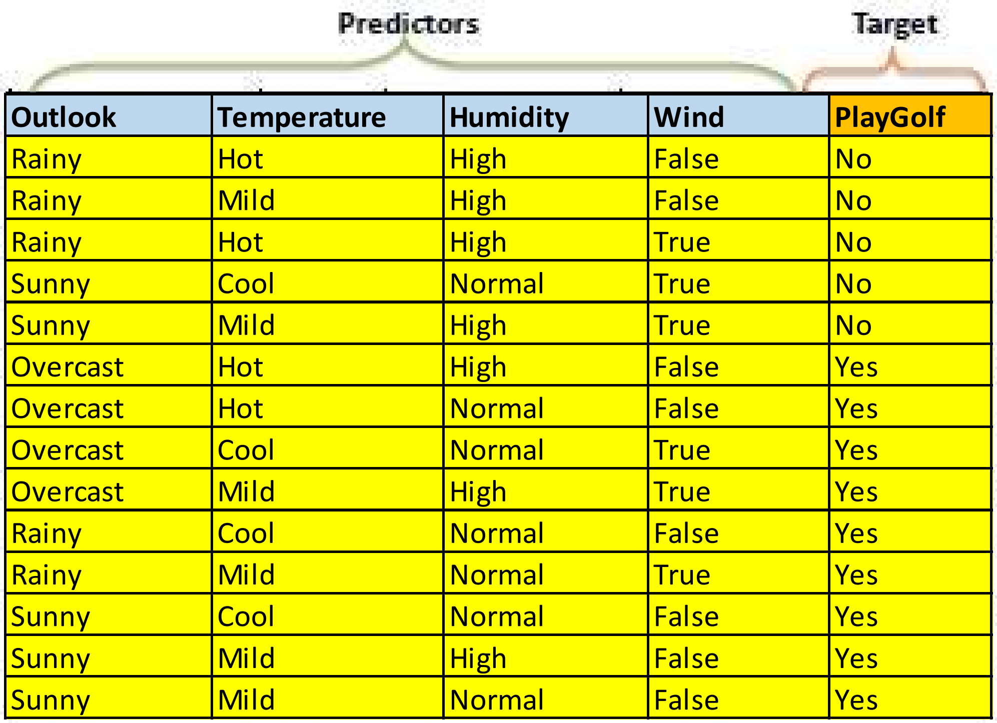
Decision Tree - Classification Technique

- The rules of Decision tree classification are very straight forward and the results can easily be presented visually.

- Additionally, because the end result is a series of logical "**if-then" statements**, there is no underlying assumption of a linear (or non-linear) relationship between the predictor variables and the dependent variable.

- It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with **Decision nodes** and **Leaf nodes**.

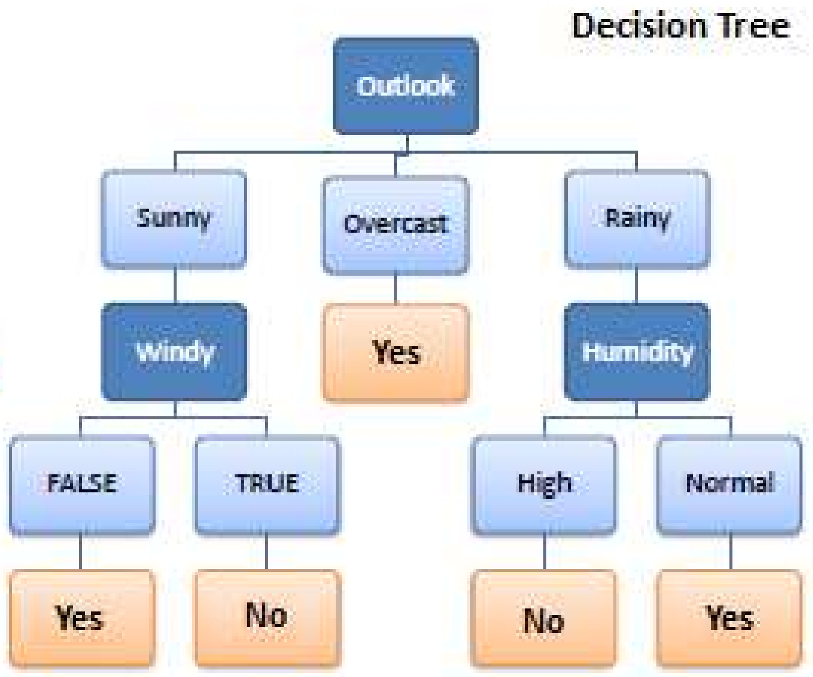
Decision Tree - Classification Technique



Predictors

Target

Outlook	Temperature	Humidity	Wind	PlayGolf
Rainy	Hot	High	False	No
Rainy	Mild	High	False	No
Rainy	Hot	High	True	No
Sunny	Cool	Normal	True	No
Sunny	Mild	High	True	No
Overcast	Hot	High	False	Yes
Overcast	Hot	Normal	False	Yes
Overcast	Cool	Normal	True	Yes
Overcast	Mild	High	True	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Mild	Normal	False	Yes



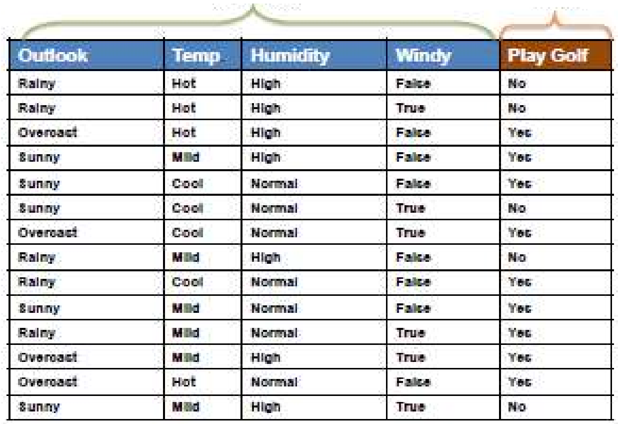
Decision Tree - Classification Technique

- Decision trees can handle both Categorical and Numerical data.

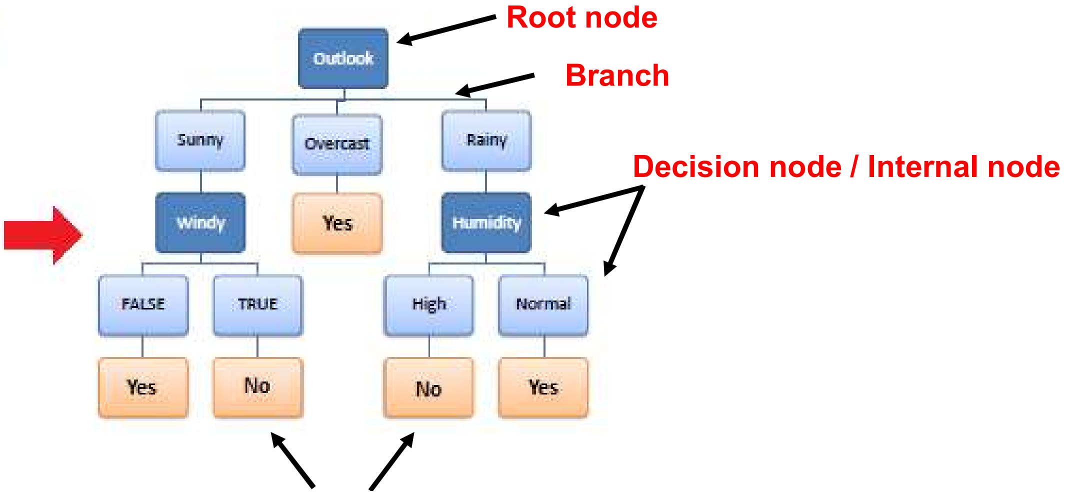
- A **Decision node** (e.g., **Outlook**) has two or more branches (e.g., Sunny, Overcast and Rainy). **Leaf node** (e.g., **PlayGolf-Yes/No**) represents a classification or decision. The top most decision node in a tree which corresponds to the **best predictor** called **Root node**.

x: attribute, predictor, independent variable, input

y: class, response, dependent variable, output



Outlook	Temp	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No



Root node

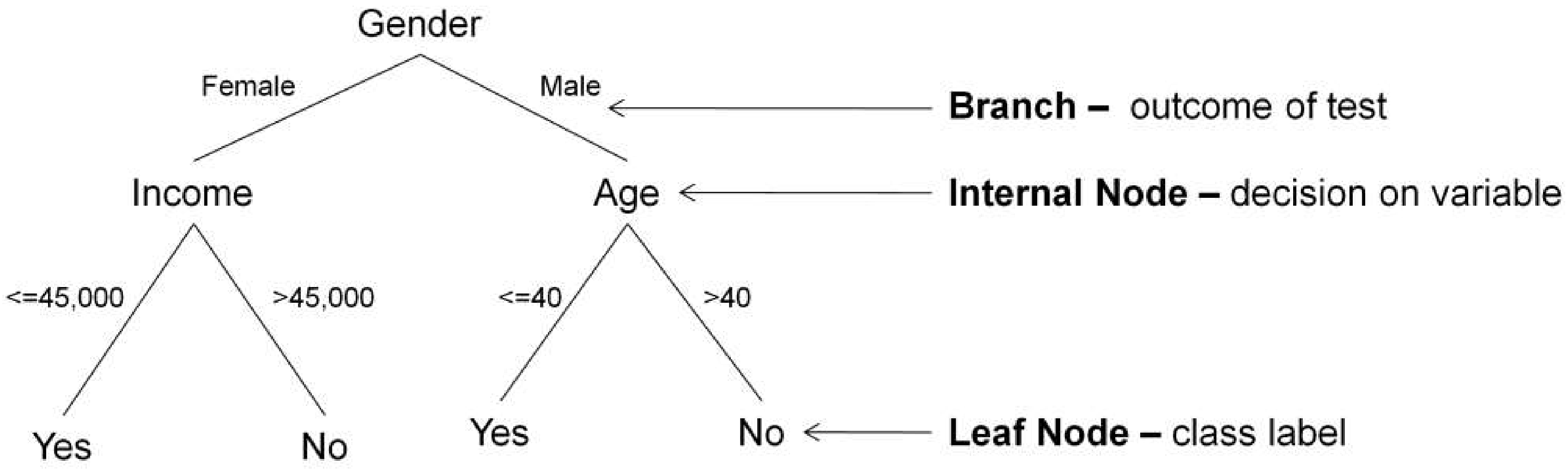
Branch

Decision node / Internal node

Decision Tree - Classification Technique – Another Example

Female

Male



Branch – outcome of test

Internal Node – decision on variable

Leaf Node – class label

Income

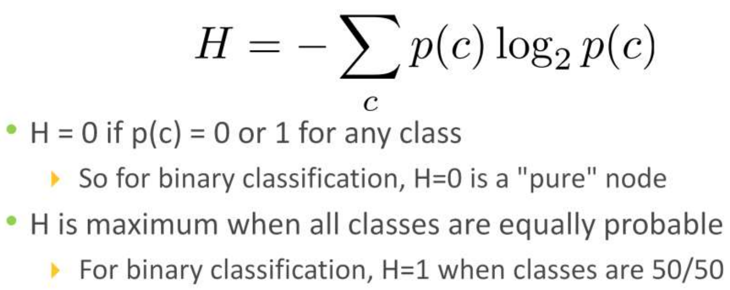
Age

Decision Tree - Algorithm

- Decision Tree algorithm uses **Entropy** and **Information Gain** to construct a decision tree. It does this by finding the single best predictor (most informative attribute), and making it as the root node.
- Some widely used Decision tree algorithms:
  - ID3
  - CART
  - C4.5

Decision Tree - Entropy

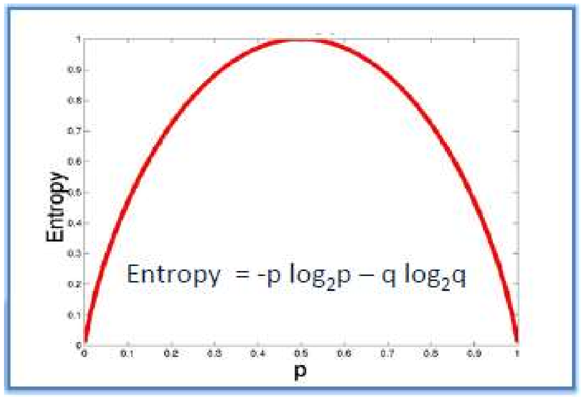
- Word **Entropy** is used to explain how similar the values in a specific group are.
- Entropy controls how a Decision Tree decides to split the data. A Decision Tree is built top-down from a **root node** and involves partitioning the data into subsets that contain instances with similar values (homogenous).
- Decision Tree algorithm uses **Entropy to calculate the homogeneity of a sample**.
- If the sample is completely homogeneous the entropy is zero, means it is a "**pure**". If the sample is equally divided (50/50) it has entropy of one, mean it is "**impure**".



$$H = -\sum_{c} p(c) \log_2 p(c)$$

- H = 0 if p(c) = 0 or 1 for any class
  - So for binary classification, H=0 is a "pure" node
- H is maximum when all classes are equally probable
  - For binary classification, H=1 when classes are 50/50

Entropy. Sometimes also denoted using the letter 'H'



Entropy = $-0.5 \log_2 0.5 - 0.5 \log_2 0.5 = 1$

Decision Tree - Entropy Formula

Both formulas are same!

Entropy. Sometimes also denoted using the letter ‘H’

$$H = -\sum_{c} p(c) \log_2 p(c)$$

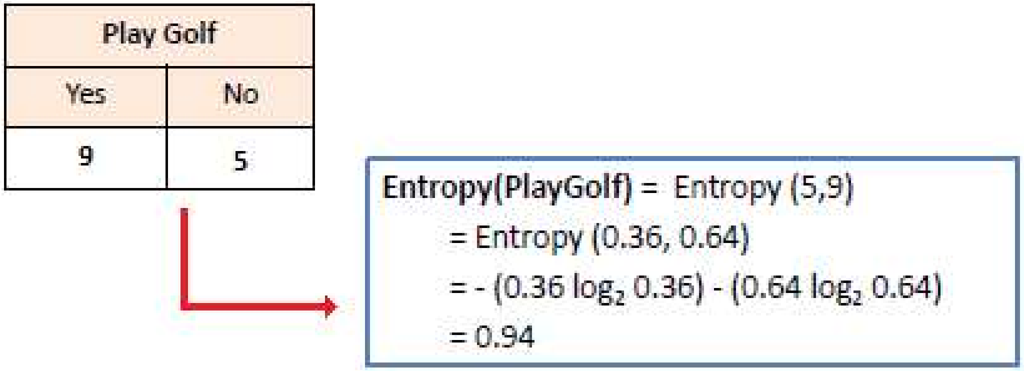
$$E(S) = \sum_{i=1}^{c} -p_i \log_2 p_i$$

Decision Tree - Calculation of Base Entropy

To build a decision tree, we need to calculate two types of entropy using frequency tables as follows:

- a) The process starts by Calculating entropy of the target/class (**PlayGolf**): **Calculation of the Base Entropy**
- **This is also know as Class Entropy**

$$E(S) = \sum_{i=1}^{c} - p_i \log_2 p_i$$



Play Golf	
Yes	No
9	5

Entropy(PlayGolf) = Entropy (5,9)

= Entropy (0.36, 0.64)

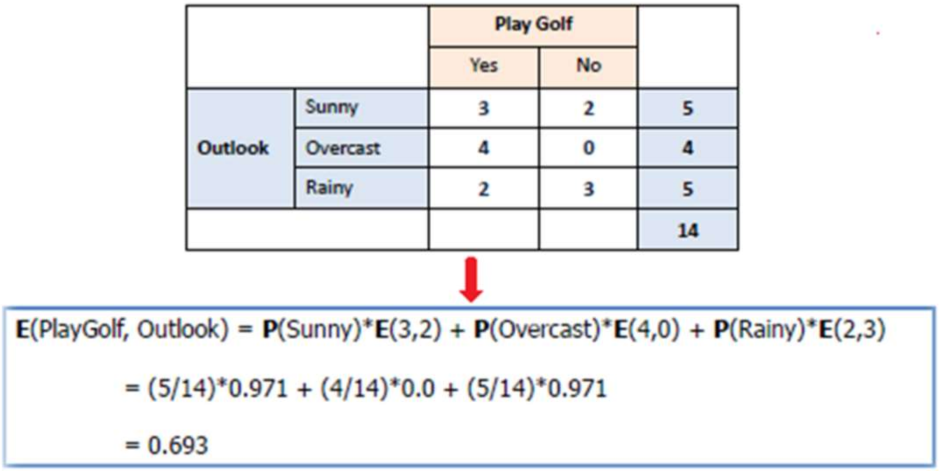
= - (0.36 $\log_2$ 0.36) - (0.64 $\log_2$ 0.64)

= 0.94

Decision Tree - Calculation of Conditional Entropy

- **b)** The dataset is then split on the different attributes. **The Entropy for each attribute is calculated. This is known as Conditional Entropy, aka Attribute Entropy**
- Calculation for the Outlook attribute: which contain three events: Sunny, Overcast, Rainy.
- This process will be repeated for every other attribute.

$$E(T,X) = \sum_{c \in X} P(c)E(c)$$



		Play Golf		
		Yes	No	
Outlook	Sunny	3	2	5
	Overcast	4	0	4
	Rainy	2	3	5
				14

E(PlayGolf, Outlook) = **P**(Sunny)***E**(3,2) + **P**(Overcast)***E**(4,0) + **P**(Rainy)***E**(2,3)

= (5/14)*0.971 + (4/14)*0.0 + (5/14)*0.971

= 0.693

Decision Tree - Information Gain

What is **Information Gain** and why it is matter in Decision Tree?

- Information Gain (IG) measures how much "information" a attribute/predictor gives us about the class.

Why it matter ?

- Information Gain is the main key that is used by Decision Tree Algorithms to construct a Decision Tree.
- Decision Tree algorithm will always tries to **maximize Information Gain**.
- So the most informative attribute is the attribute with most Information Gain (Pure)
- An attribute/predictor with highest Information Gain will split first. (Root Node)
- **Information Gain is defined as the difference between the Base Entropy and the Conditional Entropy of the attribute.**

The Equation of Information Gain → This process will be repeated for every attribute

$$Gain(T,X) = Entropy(T) - Entropy(T,X)$$

$$\text{InfoGain}_{attr} = H - Hattr$$

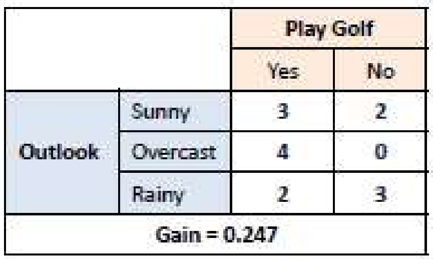
→

$$\mathbf{G}(\text{PlayGolf, Outlook}) = \mathbf{E}(\text{PlayGolf}) - \mathbf{E}(\text{PlayGolf, Outlook})$$

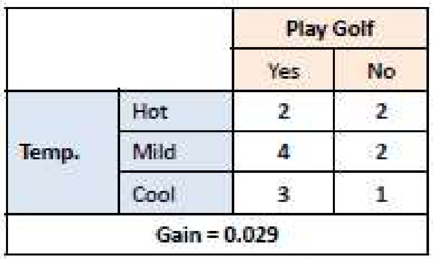
$$= 0.940 - 0.693 = 0.247$$

Decision Tree - Information Gain

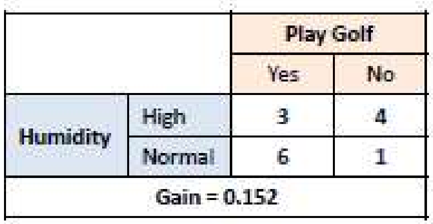
- The dataset is then split on the different attributes. The entropy for each branch is calculated. Then it is added proportionally, to get total entropy for the split. The resulting entropy is subtracted from the entropy before the split. **The result is the Information Gain, or decrease in entropy.**



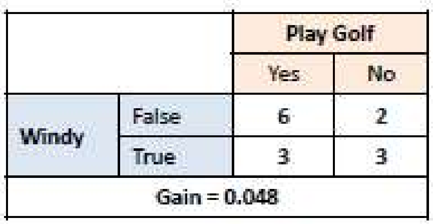
		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			



		Play Golf	
		Yes	No
Temp.	Hot	2	2
	Mild	4	2
	Cool	3	1
Gain = 0.029			



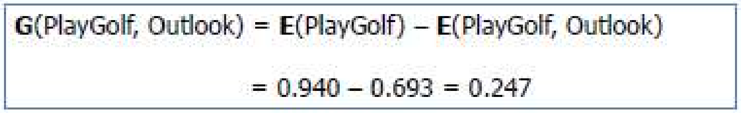
		Play Golf	
		Yes	No
Humidity	High	3	4
	Normal	6	1
Gain = 0.152			



		Play Golf	
		Yes	No
Windy	False	6	2
	True	3	3
Gain = 0.048			

$$Gain(T,X) = Entropy(T) - Entropy(T,X)$$

$$\text{InfoGain}_{attr} = H - Hattr$$

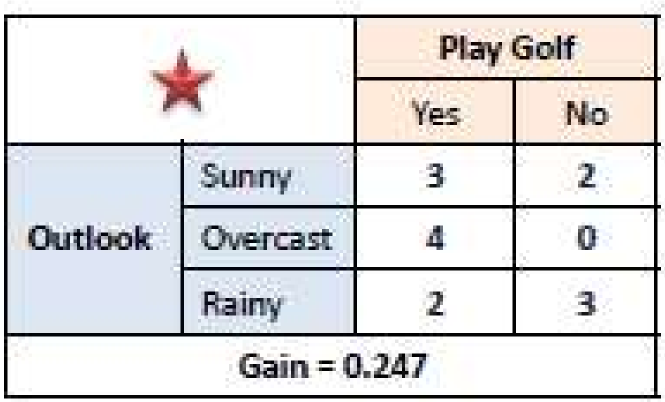


G(PlayGolf, Outlook) = **E**(PlayGolf) – **E**(PlayGolf, Outlook)

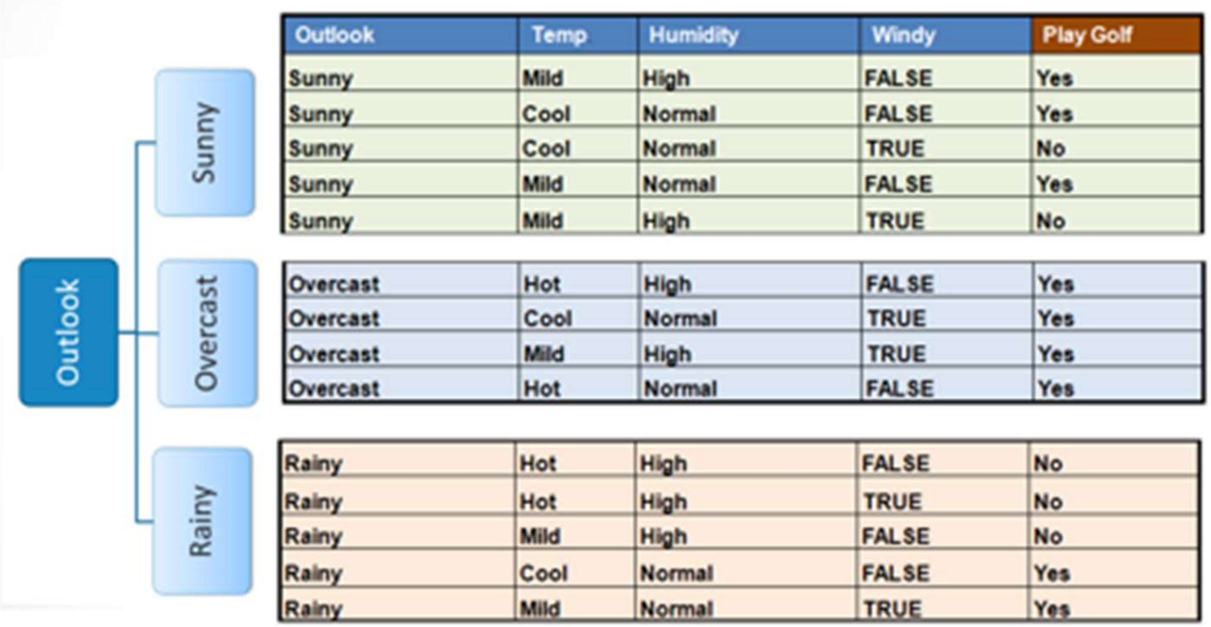
= 0.940 – 0.693 = 0.247

Decision Tree - Information Gain

- **Choose attribute with the largest information gain as the Root Node/Decision Node**, divide the dataset by its branches and repeat the same process on every branch



★		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			



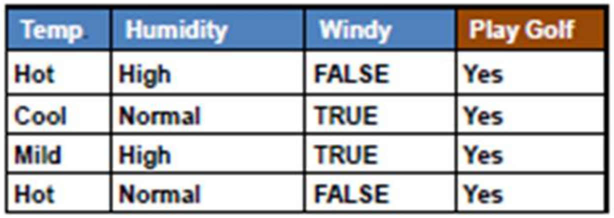
Outlook	Temp	Humidity	Windy	Play Golf
Sunny	Mild	High	FALSE	Yes
Sunny	Cool	Normal	FALSE	Yes
Sunny	Cool	Normal	TRUE	No
Sunny	Mild	Normal	FALSE	Yes
Sunny	Mild	High	TRUE	No

Overcast	Hot	High	FALSE	Yes
Overcast	Cool	Normal	TRUE	Yes
Overcast	Mild	High	TRUE	Yes
Overcast	Hot	Normal	FALSE	Yes

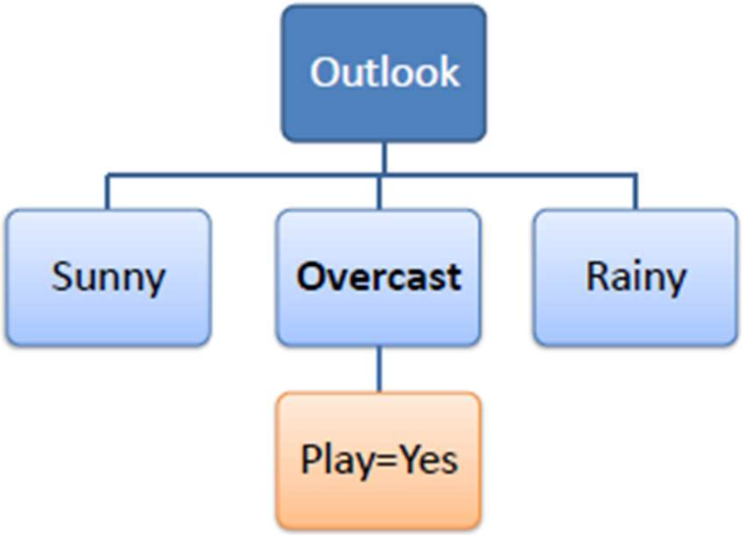
Rainy	Hot	High	FALSE	No
Rainy	Hot	High	TRUE	No
Rainy	Mild	High	FALSE	No
Rainy	Cool	Normal	FALSE	Yes
Rainy	Mild	Normal	TRUE	Yes

Decision Tree - Information Gain

- A branch with entropy of 0 is a leaf node.

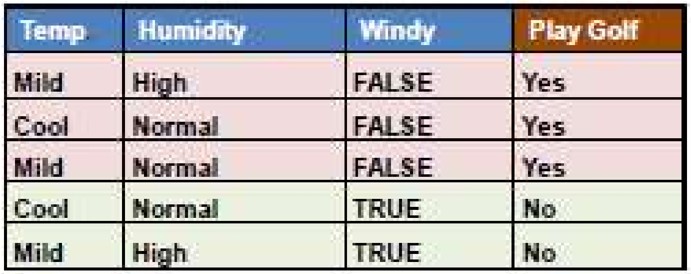


Temp	Humidity	Windy	Play Golf
Hot	High	FALSE	Yes
Cool	Normal	TRUE	Yes
Mild	High	TRUE	Yes
Hot	Normal	FALSE	Yes

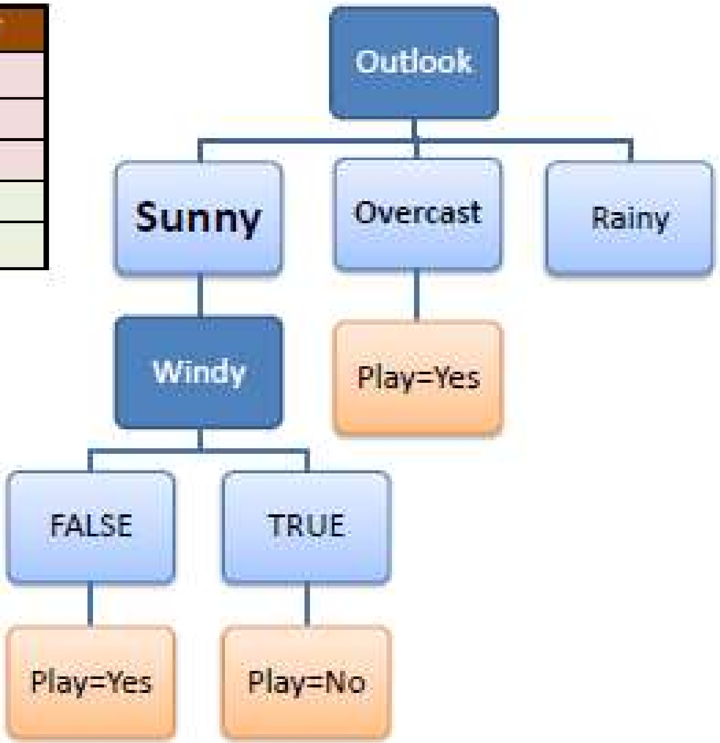


Decision Tree - Information Gain

- A branch with entropy more than 0 needs further splitting.

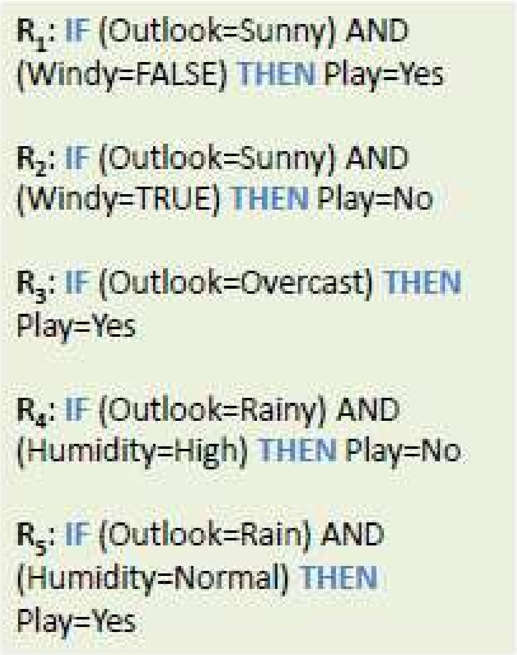


Temp	Humidity	Windy	Play Golf
Mild	High	FALSE	Yes
Cool	Normal	FALSE	Yes
Mild	Normal	FALSE	Yes
Cool	Normal	TRUE	No
Mild	High	TRUE	No



Decision Tree to Decision Rules

- A Decision Tree can easily be transformed to a set of rules by mapping from the root node to the leaf nodes one by one.



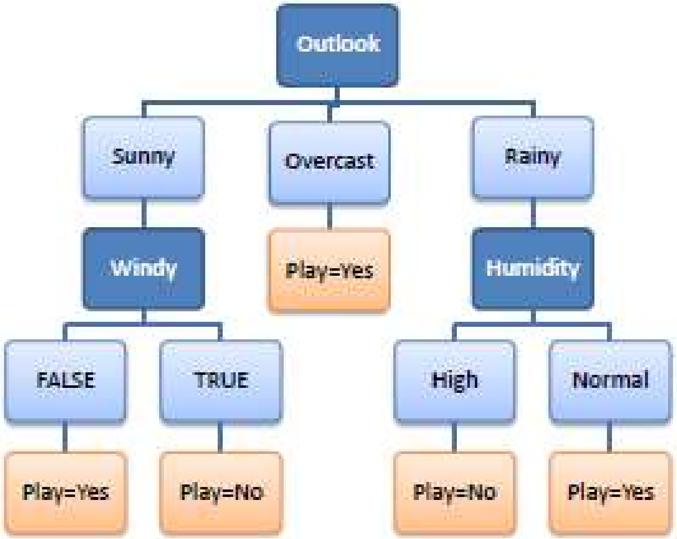
R_1: IF (Outlook=Sunny) AND (Windy=FALSE) THEN Play=Yes

R_2: IF (Outlook=Sunny) AND (Windy=TRUE) THEN Play=No

R_3: IF (Outlook=Overcast) THEN Play=Yes

R_4: IF (Outlook=Rainy) AND (Humidity=High) THEN Play=No

R_5: IF (Outlook=Rain) AND (Humidity=Normal) THEN Play=Yes



Decision Tree - Reason to Choose (+) and Cautions (-)

Reasons to Choose (+)	Cautions (-)
Takes any input type (numeric, categorical) In principle, can handle categorical variables with many distinct values (ZIP code)	Decision surfaces can only be axis-aligned
Robust with redundant variables, correlated variables	Tree structure is sensitive to small changes in the training data
Naturally handles variable interaction	A "deep" tree is probably over-fit Because each split reduces the training data for subsequent splits
Handles variables that have non-linear effect on outcome	Not good for outcomes that are dependent on many variables Related to over-fit problem, above
Computationally efficient to build	Doesn't naturally handle missing values; However most implementations include a method for dealing with this
Easy to score data	In practice, decision rules can be fairly complex
Many algorithms can return a measure of variable importance	
In principle, decision rules are easy to understand	

Decision Tree - Reason to Choose (+) and Cautions (-)

- Decision Trees take both numerical and categorical variables. They can handle many distinct values such as the zip code in the data.
- Unlike Naïve Bayesian the Decision Tree method is robust with redundant or correlated variables. Decision Trees handles variables that are non-linear. Linear/logistic regression computes the value as b1*x1 + b2*x2 .. And so on.
- If two variables interact and say the value y depends on x1*x2, linear regression does not model this type of data correctly.
- Naïve Bayes also does not do variable interactions (by design). Decision Trees handle variable interactions naturally. Every node in the tree is in some sense an interaction.
- Decision Tree algorithms are computationally efficient and it is easy to score the data. The outputs are easy to understand. Many algorithms return a measure of variable importance. Basically the information gain from each variable is provided by many packages.

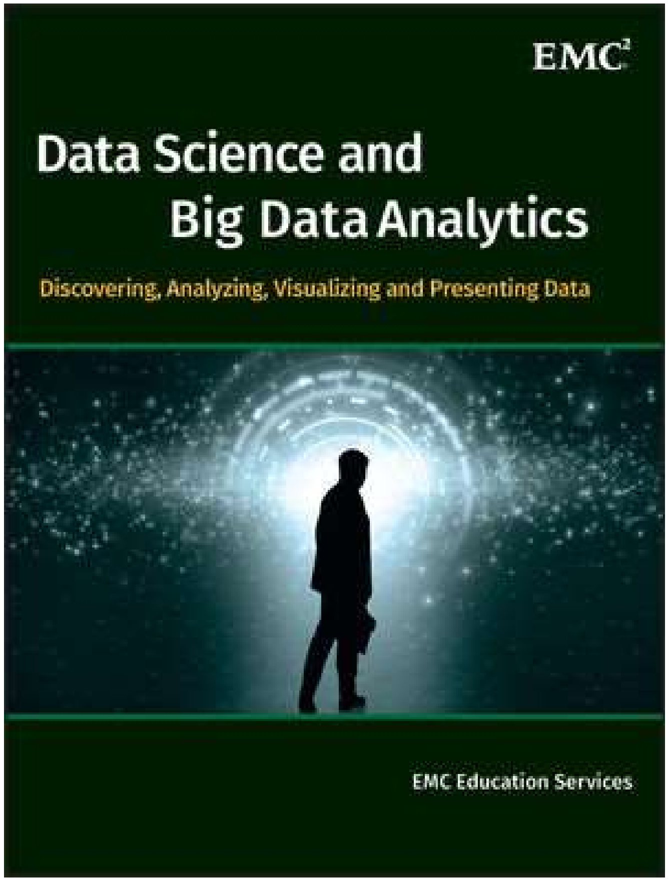
Decision Tree - Reason to Choose (+) and Cautions (-)

- In terms of Cautions (-), decision surface is axis aligned and the decision regions are rectangular surfaces. However, if the decision surface is not axis aligned (say a triangular surface), the Decision Tree algorithms do not handle this type of data well.
- Tree structure is sensitive to small variations in the training data. If you have a large data set and you build a Decision Tree on one subset and another Decision Tree on a different subset the resulting trees can be very different even though they are from the same data set. If you get a deep tree you are probably over fitting as each split reduces the training data for subsequent splits.
- Decision Trees are not good for outcomes that are dependent on many variables. This may contradict the notion that they are robust with redundant variables and correlated variables.
- If you have redundant variables, Decision Trees ignore them as the algorithm cannot detect any information gain. If there variables are important and if you split on these variables you will end up with less data with every split.

Decision Tree - Reason to Choose (+) and Cautions (-)

- So if you are dependent on too many variables, Decision Trees may not work well. You will end up with over fit trees. You can compensate for the instability and potential over-fitting of deep trees by combining the decisions of several randomized shallow Decision Trees. Or other "weak learners" – but usually trees use an ensemble model for classification. This has been shown to improve predictive power compared to a single model.
- If you are modelling with logistic regression and you have 500 variables and you really do not know which ones to choose. You can use Decision Trees to determine which variables to select based on information gain. Then choose those variables for the logistic regression model. Decision Trees can be used to prune redundant variables.
- Decision Trees don’t naturally handle missing values though many implementations include a method for dealing with this. Even though we mentioned that decision rules are easy to understand, in practice they can be very complex.

Recommended Reading Materials



Data Science and Big Data Analytics: Discovering, Analyzing, Visualizing and Presenting Data

Recommended Reading Materials

- http://www.saedsayad.com/data_mining_map.htm
- http://www.saedsayad.com/decision_tree.htm
- http://www.saedsayad.com/decision_tree_reg.htm

Any Questions?