Practical Machine Learning with R

Lecture 3 Classification: Decision Trees and Rules

Objectives:

- ❖ Comprehend the "greedy" partition algorithm implementation for decision trees and rules.
- ❖ The usage of algorithms like C 5.0, Ripper,1R for classification
- ❖ Usage of Entropy for evaluating the Information Gain with regards to the classification
- ❖ Comparison of Decision Trees and Rule learners.
- ❖ Juxtaposing pre-pruning and post-pruning.

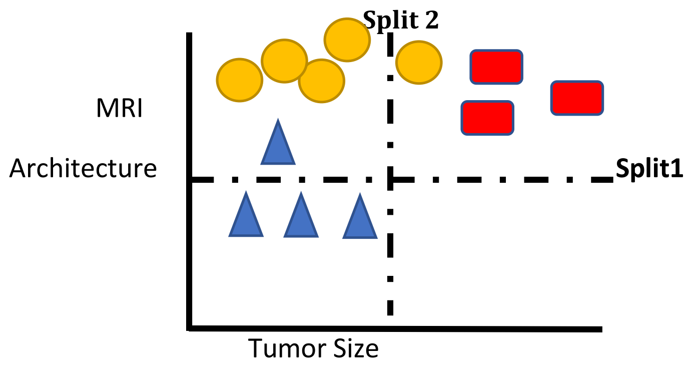
Decision Tree Algorithm:

- Decision Trees are eager learner, non-parametric ML techniques which makes no assumptions regarding the underlying data distribution. Tree are easily interpretable therefore have a broad scope of applicability specially within domain areas of Agriculture, Loan Categorization etc.
- The spectrum of applications additionally range from credit scoring to marketing applications as well as medical diagnosis.
- Trees can handle both quantitative(discrete and continuous) data as well as categorical data. This ML technique can be used both for classification and prediction.
- The decision tree learner algorithm implements classification by emulating an invertible tree structure, initiating the decision making from the root node and terminating at the leaf nodes.
- The initial segmentation is performed based on the feature/attribute considered at the root node then according to the choices made the tree is partitioned upon a feature (most predictive of the targeted class) iteratively into nested decision nodes , traversing down the various branches based on the best feature at that level until the terminal leaf nodes provide the final outcome for the related pathways. The algorithm determines the terminal node by a stopping criterion based on the homogeneity of the final classes or a predefined stopping criterion.
- The algorithm will stop when all the examples are addressed or there are no more features to differentiate examples with or a specified homogeneity has been achieved or the tree has grown to a pre-established size.
- The traversal into the partitioned pathways is determined by a heuristic (follows sound principle of logical reasoning) This recursive partitioning is a perspective that is also called Divide and Conquer(top down).
- The Classification model can be replaced by a Predictive model if a more concrete action has to be taken for instance does an individual qualify for a loan or not.
- The holistic visualization of the partitions makes the tree classification interpretable and intuitive.
- Tree classification encompasses segmenting most kinds of data like categorical and quantitative except the scenarios where the data has multitude of numeric features or nominal features with various levels which would finally result in overfitting the data. This can be overcome by adjusting some parameter. Juxtaposing Tree classification in contrast to other ML techniques , the aforementioned characteristics showcase the leverage that Trees have over the other techniques.

- Trees become computationally intensive, inefficient with regards time as well as complexity with regards to interpretability if there are too many sub branches.
- The popular algorithms for tree classification and rule based classification include C5.0,1R an RIPPER algorithms.
- For Decision trees the features to separate on are decided by the tree algorithm but for Naïve Bayes all features specified will be used. KNN is a lazy learner whereas decision trees are eager learners. Scenarios are disparate therefore the best strategy is to implement them and then compare.

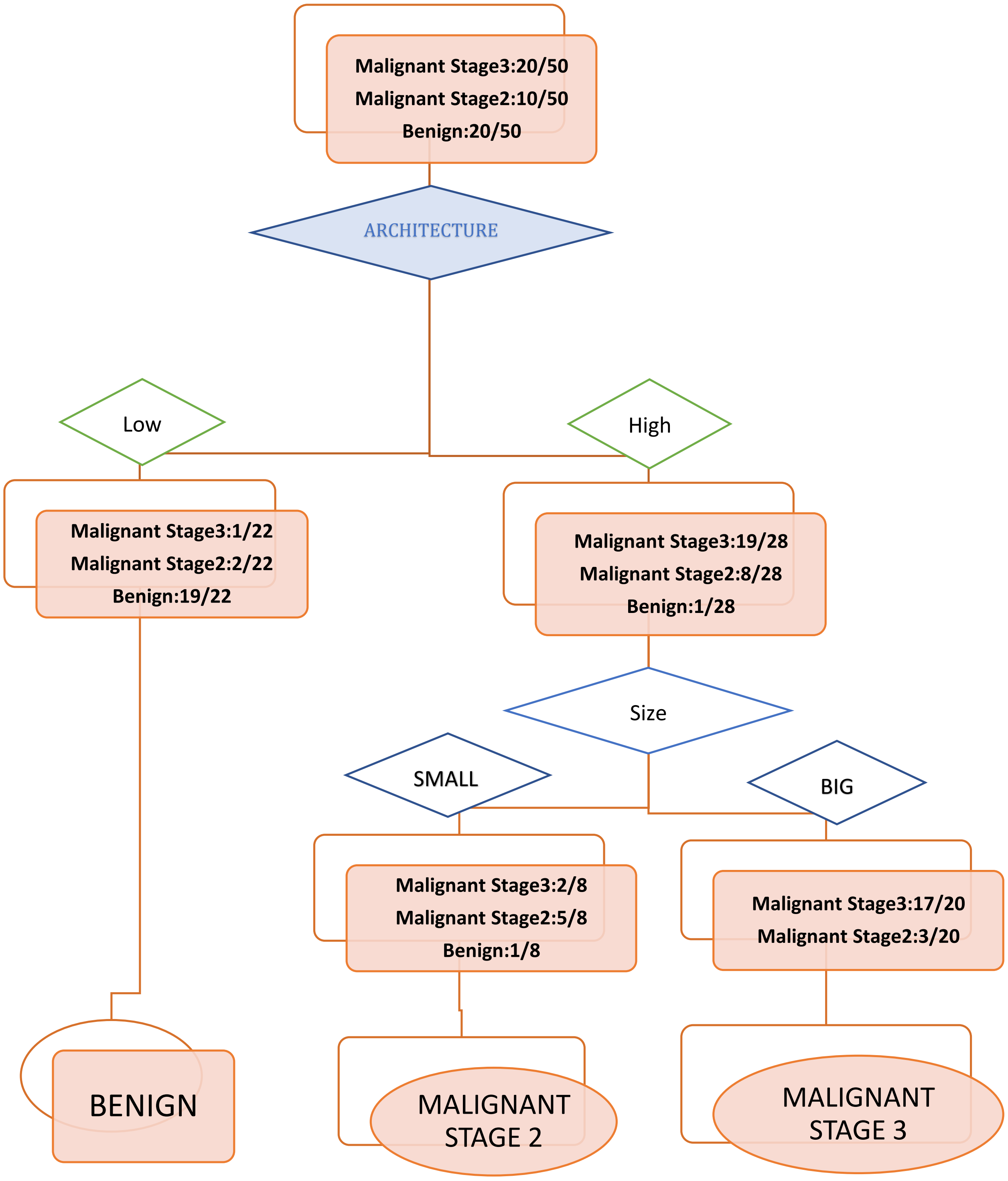
Decision Tree Algorithm Implementation:

Hypothetically if we wish to classify a tumor as malignant-stage3, malignant-Stage4or benign and we have numerous cases to classify we can use classification tree algorithm. The classification partition is implemented by axis parallel splits since the tree algorithm divides and conquers based on a specific feature at a time. The insight into the cancer categorization can be obtained by features like tumor size(area) and tumor architecture(volume) using MRI.



The top right column defines the individuals with Malignant-Stage 4 tumor , top left defines the individuals with Malignant-Stage 3 tumor and the bottom left are the ones with benign tumor.

As a rule of thumb by experiential evidence if more than 80% of the data is classified correctly we can stop splitting the data because if we partition the features further into bin ranges and continue the splitting then we are liable to overfit the data which will result in a model not generalizing well for the test data. This can be used as the stopping criterion. There can be numerous features in context to this scenario but to understand the classification technique visually we will the split the tree by two features:


Malignant Stage3:20/50
Malignant Stage2:10/50
Benign:20/50
ARCHITECTURE
Low
High
Malignant Stage3:1/22
Malignant Stage2:2/22
Benign:19/22
Malignant Stage3:19/28
Malignant Stage2:8/28
Benign:1/28
Size
SMALL
BIG
Malignant Stage3:2/8
Malignant Stage2:5/8
Benign:1/8
Malignant Stage3:17/20
Malignant Stage2:3/20
BENIGN
MALIGNANT STAGE 2
MALIGNANT STAGE 3

<u>C5 Algorithm</u>

- This white box tree algorithm is simple to comprehend and interpret and its performance is as good as the Black box methods like Neural Networks and Support Vector Machine.
- C5 uses the paradigm of Entropy to split the data iteratively into segments. Entropy is a measure that ascertain the randomness in the data therefore high entropy is an indication of less homogeneity within the groups and does not provide us with information regarding numerous other items in the set. Purity is also a measure of degree to which a subgroup contains one unique class.
- For 2 classes within a segment S entropy ranges from 0 to 1.
- For n classes the value of Entropy is measured in bits ranges from 0 to $\log_2$ n.
- Mathematically:

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

c number of class levels and p_i proportion of values falling in class level i.

- For instance if we have a data segment S containing 2 classes with 65% in one group and 35% in the other group then the entropy is calculated as:

$$\boldsymbol{Entropy(S) = -.65 * log_2 .65 + -.35 * log_2 .35 = (-0.621) * -.65 + (-1.514) * -.35 = .93355}$$

- The proportion values for each class are close to 50% therefore the Entropy close to 1.
- To make a decision regarding which feature to split on, a numerical measure called Information Gain which utilizes Entropy is calculated for each feature. Information gain on a specific feature F is computed by calculating the difference between entropy of the segment before the split(S1) and entropy of the partitions created after the split(S2).

  InfoGain(F) = Entropy(S1)-Entropy(S2)

- If S2 split creates numerous partitions then to ensure that the resultant split S2 calculates the Entropy(S2) equitably across the partitions , weightages are assigned to the Entropy as per the number of records in each of the partitions and these are then cumulated:

$$\boldsymbol{Entropy(S) = \sum_{i=1}^{c} w_i\, \mathbf{Entropy}\, p_i}$$

- The higher the Information gain the better the particular feature or attribute is in creating more homogeneous groups. Maximum Entropy is the Entropy achieved prior to the split. If the Information gain is zero it implies that split did not reduce the entropy.
- By default the above mathematical formula assumes the data categories to be nominal consequently the numeric data is transposed by divided it into 2 groups , each group incorporating values within a specified range. The numerical split that provides the largest information gain is selected for nested classification.
- C5.0 is a classification algorithm performs well but has the weakness that it is biased towards features that have a large number of levels.

- It is an algorithm that can handle both numeric as well as nominal features, it can handle unimportant features and missing data though it is prone to underfit or overfit the model.
- It can be implemented on a large data set as well as small dataset though does not represent a dataset that requires **non-parallel axis split.**
- The tree structure does not require a significant amount of mathematical interpretation.
- Trees can model complex structures though large tree can some times present conflicting decisions that are not as intuitive and logical as expected
- The small alteration to the training data might cause a significant change to the trees logical structure.

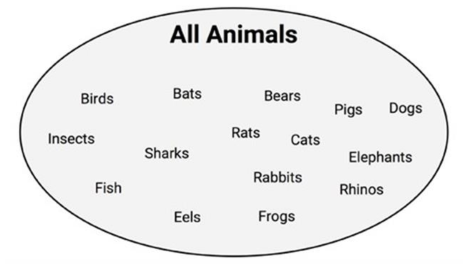
Pruning:

- Data is partitioned recursively as per the feature splitting criterion consequent to which a tree that is formulated is sometimes overly large. If the tree is too exhaustive then the problem of overfitting emerges and the tradeoff between specificity in the training phase and generalizability during the testing phase is encountered.
- Data naturally partitions until all the examples are perfectly partitioned or if the data runs out of splitting features. There are two strategies to resolve this issue , namely pre-pruning and post-pruning. Pre-pruning entails cutting off the tree once a certain depth (depicted by number of decisions) is reached or each decision node reaches a predefined minimum threshold of the number of examples encapsulated within node. The disadvantage of this strategy is that some critical learning patterns might get eliminated due to the pruning.
- Post pruning is a better strategy to implement since the data tree is allowed to grow to its natural length therefore ensuring all the patterns that are plausible have been explored and investigated before pruning the leaf nodes that have insignificant patterns or structures.
- One of the algorithms prevalently utilized is the C5.0 Algorithm which post-prunes the tree using various contextual mechanisms of eliminating the branches that have no effect on classification as well as replacing some upper level hierarchical structures by the lower level more significant branches. This technique is called subtree raising or sub tree replacement.

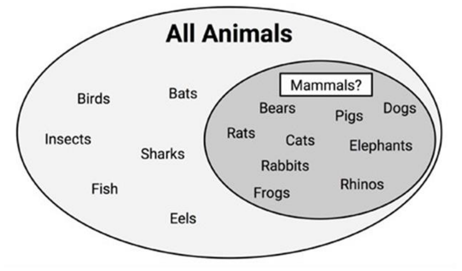
Classification Rules:

- Classification rules follow the paradigm of "if then else" protocol with the antecedent stating the if part comprising of a feature or a combination of features and the else part comprising of the class value if the conditions specified in the rule is achieved.
- Trees are a trunk to leaf traversal through the decision tree whereas Rule learners are propositions therefore less complex and easy to understand.
- Rule learners are applicable for data that has features that are mostly nominal and they are able to differentiate rare events based on interplay between features.
- Rule learner applies a rule on to the training data and creates a subgroup of examples that satisfies the condition stipulated by the rule. This subgroup is segmented from the data. Recursively rules are applied on the data and additional subgroups are formed and separated until the examples obtain full coverage.
- The rule learner implements the Separate and Conquer (bottom up)heuristic which is different from the Divide and conquer heuristic implemented by Decision trees. The rule learner algorithm are also called covering algorithms.

Example: Identifying mammals:



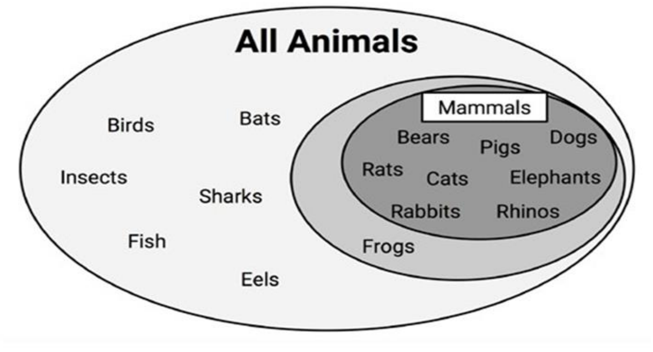
First rule = using the feature which identifies which animals are land based.



Some examples or instances are separated.

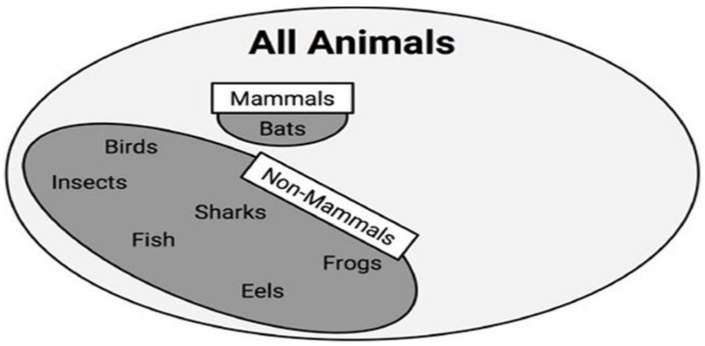
But frogs are amphibians therefore we need to make our rule more specific:

Mammals must walk on land and have tails



Now the new set of examples are conquered again and separated.

The only other additional rule required to separate bats from the remining animals is presence of fur.



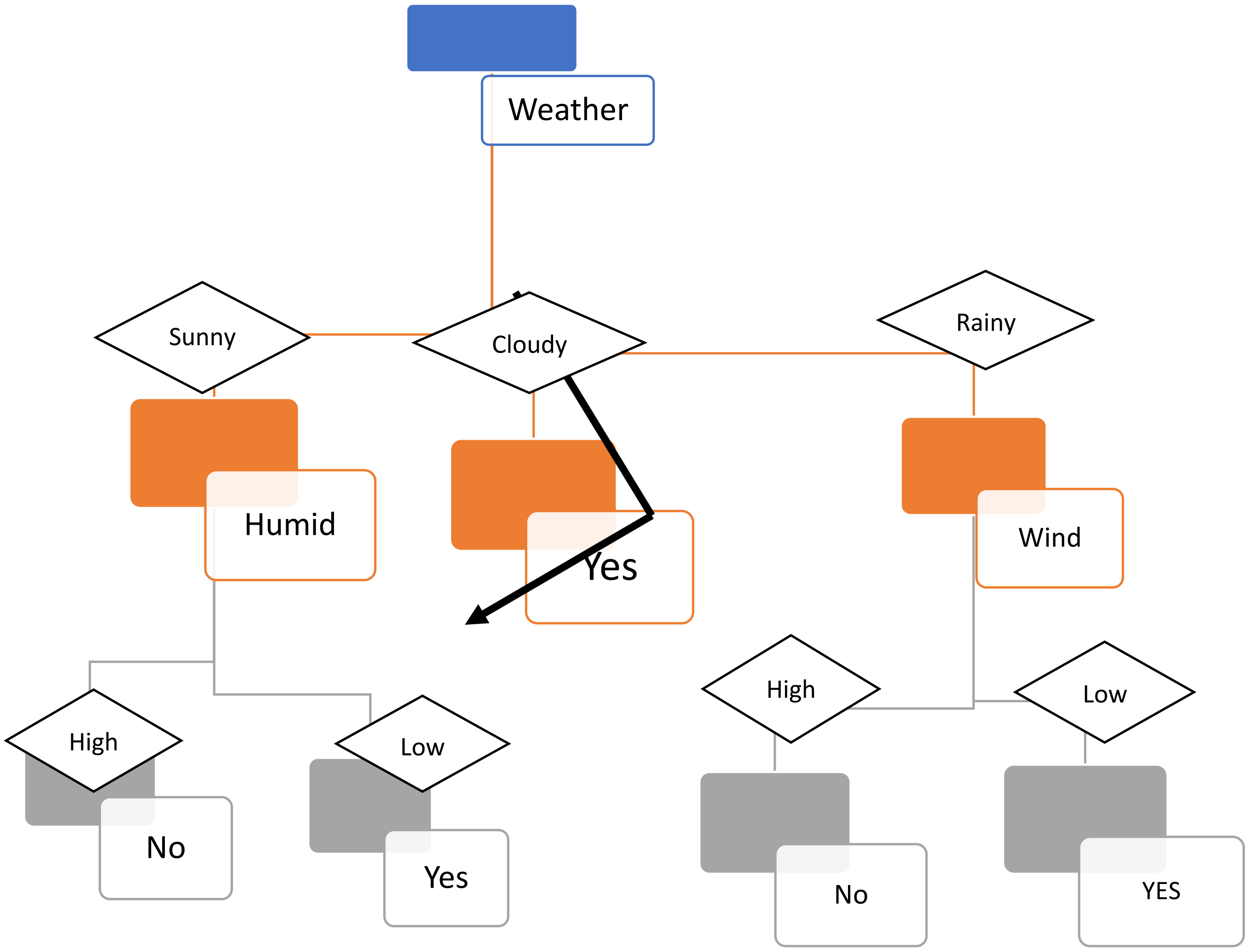
There are many famous rule learning algorithms like OneR, Incremental Reduced Error Pruning, RIPPER algorithm. OneR is an incremental formulation of the ZeroR that just classifies each unlabeled example by the majority class in the data. One R exclusively uses only one rule.

Although OneR algorithm only uses one rule its performance is good. It is simplistic can be used as a benchmark for complex algorithms. The alternate algorithm IREP handled the issues of slow performance in the face of large number examples faced by rule learner algorithm. IREP has a a set of strategies for pre and post pruning the complicated rules before separating the examples following the rule.

RIPPER algorithm grows, prunes and optimizes in context of rules implemented. RIPPER algorithm creates a simpler model, generates interpretable rules that work well with large and noisy data. RIPPER algorithms might not perform as well as some of the more complicated decision tree algorithms and RIPPER does not work with numeric data. RIPPER algorithm partitions the data by recursively adding the conditions to the rule until either a subset is perfectly classified or the attributes are exhausted.

Information gain is used to check if the specificity of the rule would reduce entropy. If the entropy is no longer reduced then rule is pruned. The algorithm grows, prunes till the stopping condition is reached and then the rules are optimized with a variety of heuristic. Rules can be created from decision trees.

Converting Trees to Rules



Tree Implementation Algorithm: 5 branches(5 leaves)

If the weather =sunny and humid =high then No
If the weather = sunny and the humid = low then Yes
If the weather = cloudy then Yes
If the weather = Rainy and wind = high then No
If the weather= Rainy and wind =low then Yes

Rule Implementation: In order to implement rules we can delete some of these clauses

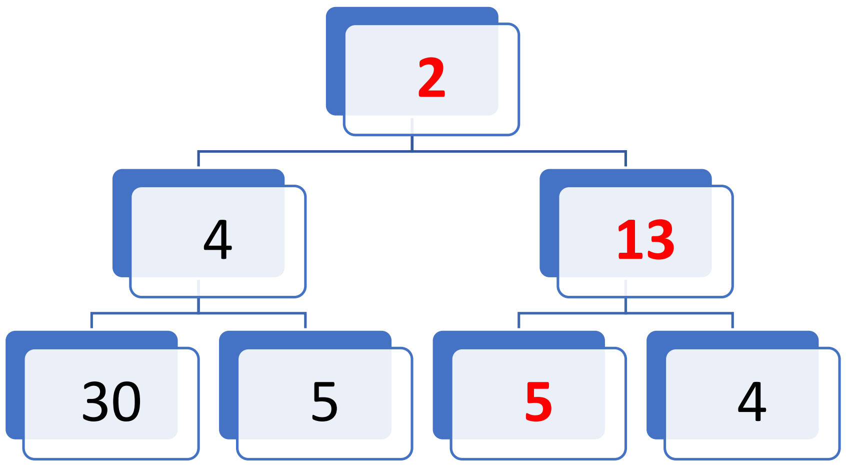
If the weather =sunny and humid =high then No
~~***If the weather = sunny and the humid = low then Yes***~~
else Yes
If the weather = cloudy then Yes
If the weather = Rainy and wind = high then No
~~***If the weather= Rainy and wind =low then Yes***~~
else Yes

Simplifying this even further

If the weather = Sunny and humid =high then No
If the weather = Rainy and wind = high then No
else Yes

Rules Versus Decisions Trees

- Rules and decision trees are equivalent , as they can describe the same phenomenon ie give a tree you can create a set of rules and given a set of rules you can describe a tree .The tree and rules provide the same predictions.
- They differ in their implementation. Rules are expressed as a decision list, they are executed in order and they can be much simpler as compared to trees.
- Rules seem to be independent set of decision list but infact they are executed sequentially and they are contextually dependent on their predecessors.
- Rules can be created either by first creating a decision tree , using the Divide and conquer strategy and then read every rule for each traversal from the root to a leaf, subsequent to which the extra rules can be eliminated.
- Rules can also be created by paradigm of Separate and Conquer (bottom up) strategy ie the covering strategy. Every class is taken sequentially subsequent a useful rule is identified that covers the examples(instances) of that class, separate these examples and then carry on and conquer the remaining examples in that class by finding and identifying more rules.
- For most cases Rules are more perspicuous versus trees . Tree implementation have repeated subtrees.
- If the data has multi class framework then rules deal with classes sequentially whereas trees address all the classes concurrently.
- Both the decision tree learner as well as the rule learners are greedy learners. The heuristic of divide and conquer as well as separate and conquer implement a paradigm of creating partitions sequentially discovering the most homogeneous partition followed by the next optimal one till the algorithm runs out of examples.
- The disadvantage of greedy learning is that the algorithm takes a more marginalized view of the data while implementing a rule or partition and therefore the this might not be optimal. If the entire data is considered then the set of rules implemented would be more streamlined.



- If we need to find the largest sum, the greedy algorithm will take the route 2---13---5 rather than 2----4------30
- If the tree is very large and if greedy algorithm is not used then the whole procedural framework is infeasible.
- If we implement the decision tree the partition that are disaggregated cannot be revisited therefore the tree is constrained by the history of decisions whereas decision rules covers specific examples but the examples that are not encompassed by all the conditions stipulated by the rule can be revisited and covered.