DATA ANALYTICS

(Professional Elective - I)
Subject Code: CS513PE

UNIT 4

NOTES MATERIAL

OBJECT SEGMENTATION
TIME SERIES METHODS

For B. TECH (CSE)

3rd YEAR – 1st SEM (R18)

Faculty:

B. RAVIKRISHNA

DEPARTMENT OF CSE

VIGNAN INSTITUTE OF TECHNOLOGY & SCIENCE

DESHMUKHI

UNIT - IV
Object Segmentation & Time Series Methods

Syllabus:

Object Segmentation: Regression Vs Segmentation – Supervised and Unsupervised Learning, Tree Building – Regression, Classification, Overfitting, Pruning and Complexity, Multiple Decision Trees etc.

Time Series Methods: Arima, Measures of Forecast Accuracy, STL approach, Extract features from generated model as Height, Average Energy etc and analyze for prediction

Topics:

Object Segmentation:

- Supervised and Unsupervised Learning
- Segmentaion & Regression Vs Segmentation
- Regression, Classification, Overfitting,
- Decision Tree Building
- Pruning and Complexity
- Multiple Decision Trees etc.

Time Series Methods:

- Arima, Measures of Forecast Accuracy
- STL approach
- Extract features from generated model as Height Average Energy etc. and analyze for prediction

Unit-4 Objectives:

1. To explore the Segmentaion & Regression Vs Segmentation
2. To learn the Regression, Classification, Overfitting
3. To explore Decision Tree Building, Multiple Decision Trees etc.
4. To Learn the Arima, Measures of Forecast Accuracy
5. To understand the STL approach

Unit-4 Outcomes:

After completion of this course students will be able to

1. To Describe the Segmentaion & Regression Vs Segmentation
2. To demonstrate Regression, Classification, Overfitting
3. To analyze the Decision Tree Building, Multiple Decision Trees etc.
4. To explore the Arima, Measures of Forecast Accuracy
5. To describe the STL approach

Supervised and Unsupervised Learning

Supervised Learning:

- Supervised learning is a machine learning method in which models are trained using labeled data. In supervised learning, models need to find the mapping function to map the input variable (X) with the output variable (Y).
- We find a relation between x & y, suchthat y=f(x)
- Supervised learning needs supervision to train the model, which is similar to as a student learns things in the presence of a teacher. Supervised learning can be used for two types of problems: Classification and Regression.
- Example: Suppose we have an image of different types of fruits. The task of our supervised learning model is to identify the fruits and classify them accordingly. So to identify the image in supervised learning, we will give the input data as well as output for that, which means we will train the model by the shape, size, color, and taste of each fruit. Once the training is completed, we will test the model by giving the new set of fruit. The model will identify the fruit and predict the output using a suitable algorithm.

Unsupervised Machine Learning:

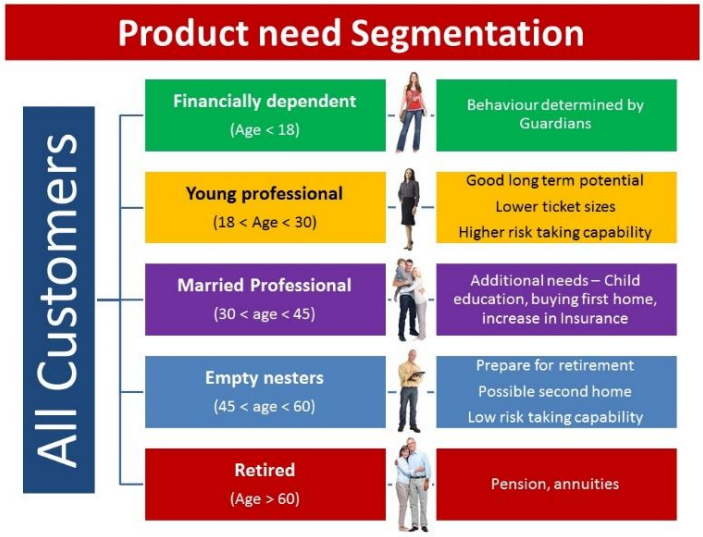
- Unsupervised learning is another machine learning method in which patterns inferred from the unlabeled input data. The goal of unsupervised learning is to find the structure and patterns from the input data. Unsupervised learning does not need any supervision. Instead, it finds patterns from the data by its own.
- Unsupervised learning can be used for two types of problems: Clustering and Association.
- Example: To understand the unsupervised learning, we will use the example given above. So unlike supervised learning, here we will not provide any supervision to the model. We will just provide the input dataset to the model and allow the model to find the patterns from the data. With the help of a suitable algorithm, the model will train itself and divide the fruits into different groups according to the most similar features between them.

The main differences between Supervised and Unsupervised learning are given below:

Supervised Learning	Unsupervised Learning
Supervised learning algorithms are trained using labeled data.	Unsupervised learning algorithms are trained using unlabeled data.
Supervised learning model takes direct feedback to check if it is predicting correct output or not.	Unsupervised learning model does not take any feedback.
Supervised learning model predicts the output.	Unsupervised learning model finds the hidden patterns in data.
In supervised learning, input data is provided to the model along with the output.	In unsupervised learning, only input data is provided to the model.
The goal of supervised learning is to train the model so that it can predict the output when it is given new data.	The goal of unsupervised learning is to find the hidden patterns and useful insights from the unknown dataset.
Supervised learning needs supervision to train the model.	Unsupervised learning does not need any supervision to train the model.
Supervised learning can be categorized in Classification and Regression problems.	Unsupervised Learning can be classified in Clustering and Associations problems.
Supervised learning can be used for those cases where we know the input as well as corresponding outputs.	Unsupervised learning can be used for those cases where we have only input data and no corresponding output data.
Supervised learning model produces an accurate result.	Unsupervised learning model may give less accurate result as compared to supervised learning.
Supervised learning is not close to true Artificial intelligence as in this, we first train the model for each data, and then only it can predict the correct output.	Unsupervised learning is more close to the true Artificial Intelligence as it learns similarly as a child learns daily routine things by his experiences.
It includes various algorithms such as Linear Regression, Logistic Regression, Support Vector Machine, Multi-class Classification, Decision tree, Random Forest, Decision Trees , Bayesian Logic, etc.	In Unsupervised Learing we have K-means clustering. KNN (k-nearest neighbors), Hierarchal clustering, Anomaly detection, Neural Networks, Principle Component Analysis, Independent Component Analysis, Apriori algorithms, etc.

Segmentation

- Segmentation refers to the act of segmenting data according to your company's needs in order to refine your analyses based on a defined context. It is a technique of splitting customers into separate groups depending on their attributes or behavior.
- The purpose of segmentation is to better understand your customers(visitors), and to obtain actionable data in order to improve your website or mobile app. In concrete terms, a segment enables you to filter your analyses based on certain elements (single or combined).
- Segmentation can be done on elements related to a visit, as well as on elements related to multiple visits during a studied period.



Steps:

- Define purpose – Already mentioned in the statement above
- Identify critical parameters – Some of the variables which come up in mind are skill, motivation, vintage, department, education etc. Let us say that basis past experience, we know that skill and motivation are most important parameters. Also, for sake of simplicity we just select 2 variables. Taking additional variables will increase the complexity, but can be done if it adds value.
- Granularity – Let us say we are able to classify both skill and motivation into High and Low using various techniques.

There are two broad set of methodologies for segmentation:

- Objective (supervised) segmentation
- Non-Objective (unsupervised) segmentation

Objective Segmentation

- Segmentation to identify the type of customers who would respond to a particular offer.
- Segmentation to identify high spenders among customers who will use the e-commerce channel for festive shopping.
- Segmentation to identify customers who will default on their credit obligation for a loan or credit card.

Non-Objective Segmentation

https://www.yieldify.com/blog/types-of-market-segmentation/

- Segmentation of the customer base to understand the specific profiles which exist within the customer base so that multiple marketing actions can be personalized for each segment
- Segmentation of geographies on the basis of affluence and lifestyle of people living in each geography so that sales and distribution strategies can be formulated accordingly.
- Hence, it is critical that the segments created on the basis of an objective segmentation methodology must be different with respect to the stated objective (e.g. response to an offer).
- However, in case of a non-objective methodology, the segments are different with respect to the "generic profile" of observations belonging to each segment, but not with regards to any specific outcome of interest.
- The most common techniques for building non-objective segmentation are cluster analysis, K nearest neighbor techniques etc.

Regression Vs Segmentation

- Regression analysis focuses on finding a relationship between a dependent variable and one or more independent variables.
- Predicts the value of a dependent variable based on the value of at least one independent variable.
- Explains the impact of changes in an independent variable on the dependent variable.
- We use linear or logistic regression technique for developing accurate models for predicting an outcome of interest.
- Often, we create separate models for separate segments.
- Segmentation methods such as CHAID or CRT is used to judge their effectiveness.

- Creating separate model for separate segments may be time consuming and not worth the effort. But, creating separate model for separate segments may provide higher predictive power.

Decision Tree Classification Algorithm:

- Decision Tree is a supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems.
- Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand.
- A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.
- It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.
- It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.
- In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
- Basic Decision Tree Learning Algorithm:
- Now that we know what a Decision Tree is, we'll see how it works internally. There are many algorithms out there which construct Decision Trees, but one of the best is called as ID3 Algorithm. ID3 Stands for Iterative Dichotomiser 3.

There are two main types of Decision Trees:

1. Classification trees (Yes/No types)

What we've seen above is an example of classification tree, where the outcome was a variable like 'fit' or 'unfit'. Here the decision variable is Categorical.

2. Regression trees (Continuous data types)

Here the decision or the outcome variable is Continuous, e.g. a number like 123.

Decision Tree Terminologies

Root Node: Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.

Leaf Node: Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.

Splitting: Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.

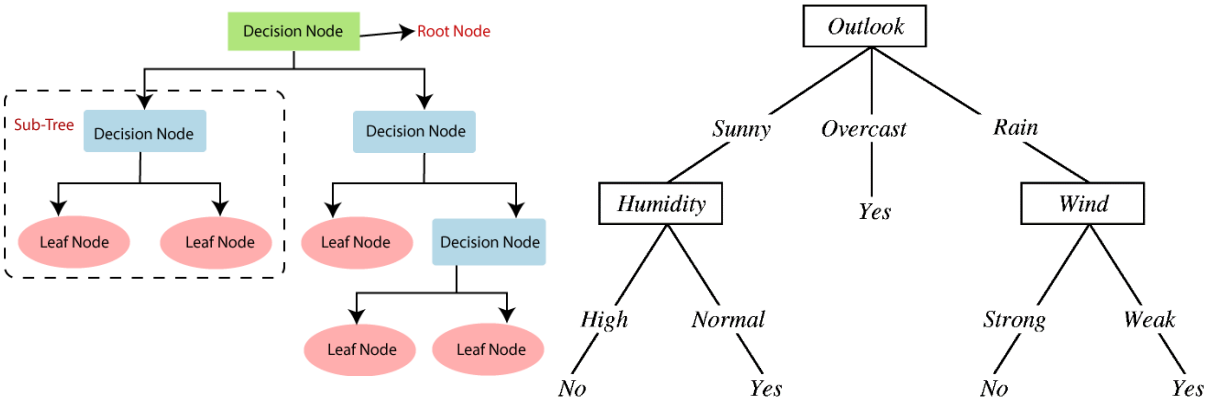
Branch/Sub Tree: A tree formed by splitting the tree.

Pruning: Pruning is the process of removing the unwanted branches from the tree.

Parent/Child node: The root node of the tree is called the parent node, and other nodes are called the child nodes.

Decision Tree Representation:

- Each non-leaf node is connected to a test that splits its set of possible answers into subsets corresponding to different test results.
- Each branch carries a particular test result's subset to another node.
- Each node is connected to a set of possible answers.
- Below diagram explains the general structure of a decision tree:



- A decision tree is an arrangement of tests that provides an appropriate classification at every step in an analysis.
- "In general, decision trees represent a disjunction of conjunctions of constraints on the attribute-values of instances. Each path from the tree root to a leaf corresponds to a conjunction of attribute tests, and the tree itself to a disjunction of these conjunctions" (Mitchell, 1997, p.53).
- More specifically, decision trees classify instances by sorting them down the tree from the root node to some leaf node, which provides the classification of the instance. Each node in the tree specifies a test of some attribute of the instance,

and each branch descending from that node corresponds to one of the possible values for this attribute.

- An instance is classified by starting at the root node of the decision tree, testing the attribute specified by this node, then moving down the tree branch corresponding to the value of the attribute. This process is then repeated at the node on this branch and so on until a leaf node is reached.

Appropriate Problems for Decision Tree Learning

Decision tree learning is generally best suited to problems with the following characteristics:

- Instances are represented by attribute-value pairs.
    - There is a finite list of attributes (e.g. hair colour) and each instance stores a value for that attribute (e.g. blonde).
    - When each attribute has a small number of distinct values (e.g. blonde, brown, red) it is easier for the decision tree to reach a useful solution.
    - The algorithm can be extended to handle real-valued attributes (e.g. a floating point temperature)
- The target function has discrete output values.
    - A decision tree classifies each example as one of the output values.
        - Simplest case exists when there are only two possible classes (Boolean classification).
        - However, it is easy to extend the decision tree to produce a target function with more than two possible output values.
    - Although it is less common, the algorithm can also be extended to produce a target function with real-valued outputs.
- Disjunctive descriptions may be required.
    - Decision trees naturally represent disjunctive expressions.
- The training data may contain errors.
    - Errors in the classification of examples, or in the attribute values describing those examples are handled well by decision trees, making them a robust learning method.
- The training data may contain missing attribute values.
    - Decision tree methods can be used even when some training examples have unknown values (e.g., humidity is known for only a fraction of the examples).

After a decision tree learns classification rules, it can also be re-represented as a set of if-then rules in order to improve readability.

How does the Decision Tree algorithm Work?

The decision of making strategic splits heavily affects a tree's accuracy. The decision criteria are different for classification and regression trees.

Decision trees use multiple algorithms to decide to split a node into two or more sub-nodes. The creation of sub-nodes increases the homogeneity of resultant sub-nodes. In other words, we can say that the purity of the node increases with respect to the target variable. The decision tree splits the nodes on all available variables and then selects the split which results in most homogeneous sub-nodes.

Tree Building: Decision tree learning is the construction of a decision tree from class-labeled training tuples. A decision tree is a flow-chart-like structure, where each internal (non-leaf) node denotes a test on an attribute, each branch represents the outcome of a test, and each leaf (or terminal) node holds a class label. The topmost node in a tree is the root node. There are many specific decision-tree algorithms. Notable ones include the following.

ID3 → (extension of D3)

C4.5 → (successor of ID3)

CART → (Classification And Regression Tree)

CHAID → (Chi-square automatic interaction detection Performs multi-level splits when computing classification trees)

MARS → (multivariate adaptive regression splines): Extends decision trees to handle numerical data better

Conditional Inference Trees → Statistics-based approach that uses non-parametric tests as splitting criteria, corrected for multiple testing to avoid over fitting.

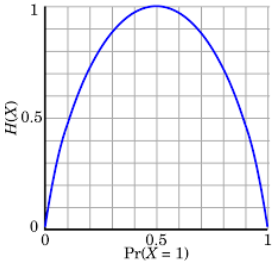
The ID3 algorithm builds decision trees using a top-down greedy search approach through the space of possible branches with no backtracking. A greedy algorithm, as the name suggests, always makes the choice that seems to be the best at that moment.

In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node.

For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree. The complete process can be better understood using the below algorithm:

- Step-1: Begin the tree with the root node, says S, which contains the complete dataset.
- Step-2: Find the best attribute in the dataset using Attribute Selection Measure (ASM).
- Step-3: Divide the S into subsets that contains possible values for the best attributes.
- Step-4: Generate the decision tree node, which contains the best attribute.
- Step-5: Recursively make new decision trees using the subsets of the dataset created in
- Step -6: Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

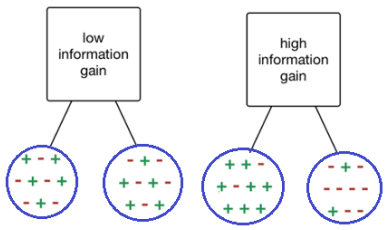
Entropy:



Entropy is a measure of the randomness in the information being processed. The higher the entropy, the harder it is to draw any conclusions from that information. Flipping a coin is an example of an action that provides information that is random.

From the graph, it is quite evident that the entropy H(X) is zero when the probability is either 0 or 1. The Entropy is maximum when the probability is 0.5 because it projects perfect randomness in the data and there is no chance if perfectly determining the outcome.

Information Gain



Information gain or IG is a statistical property that measures how well a given attribute separates the training examples according to their target classification. Constructing a decision tree is all about finding an attribute that returns the highest information gain and the smallest entropy.

ID3 follows the rule — A branch with an entropy of zero is a leaf node and A branch with entropy more than zero needs further splitting.

Hypothesis space search in decision tree learning:

In order to derive the Hypothesis space, we compute the Entropy and Information Gain of Class and attributes. For them we use the following statistics formulae:

Entropy of Class is:

$$Entropy(Class) = -\frac{P}{P+N}\log_2\left(\frac{P}{P+N}\right) - \frac{N}{P+N}\log_2\left(\frac{N}{P+N}\right)$$

For any Attribute,

$$InformationGain(Attribute)$$

$$I(p_i, n_i) = -\frac{p_i}{p_i+n_i}\log_2\left(\frac{p_i}{p_i+n_i}\right) - \frac{n_i}{p_i+n_i}\log_2\left(\frac{n_i}{p_i+n_i}\right)$$

Entropy of an Attribute is:

$$Entropy(Attribute) = \frac{\sum(p_i+n_i)}{P+N} I(p_i+n_i)$$

$$Gain = Entropy(Class) - Entropy(Attribute)$$

Illustrative Example: Concept: "Play Tennis":

Data set:

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Basic algorithm for inducing a decision tree from training tuples:

Algorithm:
Generate decision tree. Generate a decision tree from the training tuples of data
partition *D*.
Input:
Data partition, *D*, which is a set of training tuples and their associated class labels;
attribute list, the set of candidate attributes;
Attribute selection method, a procedure to determine the splitting criterion that "best" partitions the data tuples into individual classes. This criterion consists of a *splitting attribute* and, possibly, either a *split point* or *splitting subset*.
Output: A decision tree.
Method:

```
(1) create a node N;
(2) if tuples in D are all of the same class, C then
        return N as a leaf node labeled with the class C;
(3) if attribute list is empty then
        return N as a leaf node labeled with the majority class in D;
              // majority voting
(4) apply Attribute selection method(D, attribute list) to find the
"best"
    splitting criterion;
(5) Label node N with splitting criterion;
(6) if splitting attribute is discrete-valued and multiway splits
allowed
    then // not restricted to binary trees
(7) attribute list= attribute list - splitting attribute
(8) for each outcome j of splitting criterion
                          // partition the tuples and grow subtrees for
each partition
(9) let Dj be the set of data tuples in D satisfying outcome j;
                                                    // a partition
(10) if Dj is empty then
        attach a leaf labeled with the majority class in D to node N;
     else
        attach the node returned by Generate decision tree(Dj,
        attribute list) to node N;
(11) return N;
```

Advantages of Decision Tree:

- Simple to understand and interpret. People are able to understand decision tree models after a brief explanation.

- Requires little data preparation. Other techniques often require data normalization, dummy variables need to be created and blank values to be removed.
- Able to handle both numerical and categorical data. Other techniques are usually specialized in analysing datasets that have only one type of variable. (For example, relation rules can be used only with nominal variables while neural networks can be used only with numerical variables.)
- Uses a white box model. If a given situation is observable in a model the explanation for the condition is easily explained by Boolean logic. (An example of a black box model is an artificial neural network since the explanation for the results is difficult to understand.)
- Possible to validate a model using statistical tests. That makes it possible to account for the reliability of the model.
- Robust: Performs well with large datasets. Large amounts of data can be analyzed using standard computing resources in reasonable time.

Tools used to make Decision Tree:

Many data mining software packages provide implementations of one or more decision tree algorithms. Several examples include:

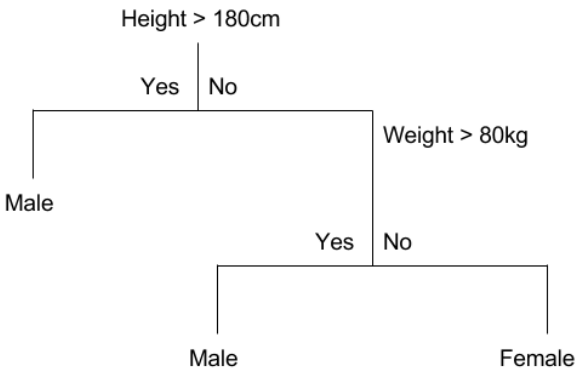
- SAS Enterprise Miner
- Matlab
- R (an open source software environment for statistical computing which includes several CART implementations such as rpart, party and random Forest packages)
- Weka (a free and open-source data mining suite, contains many decision tree algorithms)
- Orange (a free data mining software suite, which includes the tree module orngTree)
- KNIME
- Microsoft SQL Server
- Scikit-learn (a free and open-source machine learning library for the Python programming language).
- Salford Systems CART (which licensed the proprietary code of the original CART authors)
- IBM SPSS Modeler
- Rapid Miner

Multiple Decision Trees:
Classification & Regression Trees:

- ✓ Classification and regression trees is a term used to describe decision tree algorithms that are used for classification and regression learning tasks.
- ✓ The Classification and Regression Tree methodology, also known as the CART were introduced in 1984 by Leo Breiman, Jerome Friedman, Richard Olshen, and Charles Stone.

Classification Trees:

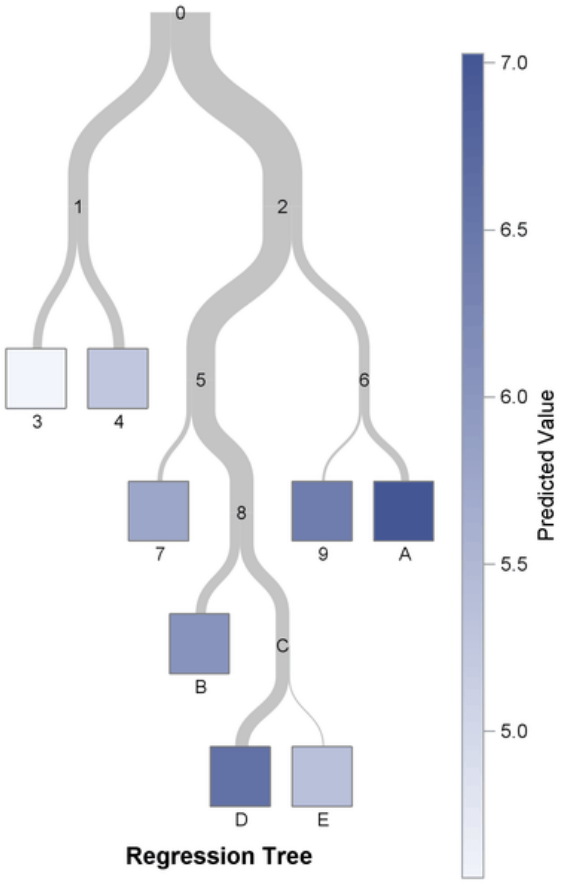
A classification tree is an algorithm where the target variable is fixed or categorical. The algorithm is then used to identify the "class" within which a target variable would most likely fall.



- ✓ An example of a classification-type problem would be determining who will or will not subscribe to a digital platform; or who will or will not graduate from high school.
- ✓ These are examples of simple binary classifications where the categorical dependent variable can assume only one of two, mutually exclusive values.

Regression Trees

- ✓ A regression tree refers to an algorithm where the target variable is and the algorithm is used to predict its value which is a continuous variable.
- ✓ As an example of a regression type problem, you may want to predict the selling prices of a residential house, which is a continuous dependent variable.
- ✓ This will depend on both continuous factors like square footage as well as categorical factors.



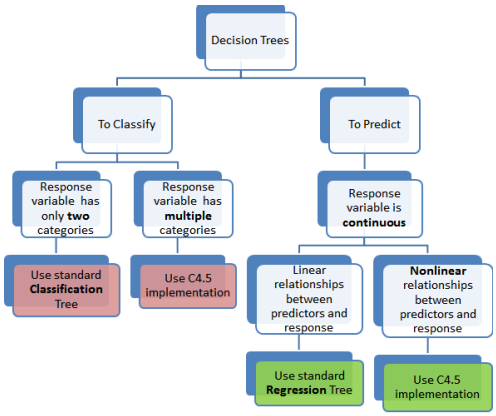
Difference Between Classification and Regression Trees

- ✓ Classification trees are used when the dataset needs to be split into classes that belong to the response variable. In many cases, the classes Yes or No.
- ✓ In other words, they are just two and mutually exclusive. In some cases, there may be more than two classes in which case a variant of the classification tree algorithm is used.
- ✓ Regression trees, on the other hand, are used when the response variable is continuous. For instance, if the response variable is something like the price of a property or the temperature of the day, a regression tree is used.
- ✓ In other words, regression trees are used for prediction-type problems while classification trees are used for classification-type problems.

CART: CART stands for Classification And Regression Tree.

- ✓ CART algorithm was introduced in Breiman et al. (1986). A CART tree is a binary decision tree that is constructed by splitting a node into two child nodes repeatedly, beginning with the root node that contains the whole learning sample. The CART growing method attempts to maximize within-node homogeneity.
- ✓ The extent to which a node does not represent a homogenous subset of cases is an indication of impurity. For example, a terminal node in which all cases have the same

value for the dependent variable is a homogenous node that requires no further splitting because it is "pure." For categorical (nominal, ordinal) dependent variables the common measure of impurity is Gini, which is based on squared probabilities of membership for each category. Splits are found that maximize the homogeneity of child nodes with respect to the value of the dependent variable.



Decision tree pruning:

Pruning is a data compression technique in machine learning and search algorithms that reduces the size of decision trees by removing sections of the tree that are non-critical and redundant to classify instances. Pruning reduces the complexity of the final classifier, and hence improves predictive accuracy by the reduction of overfitting.

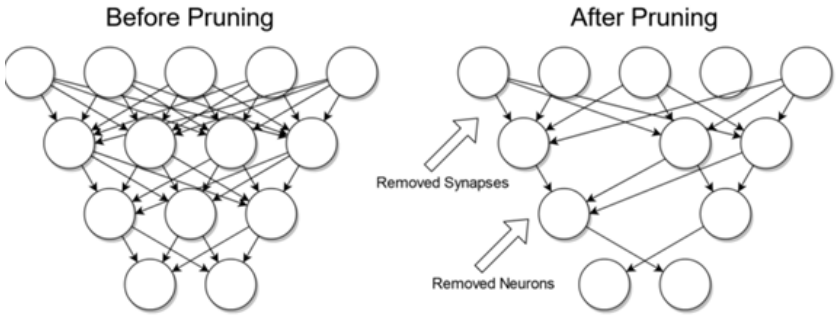


Fig: Before and After pruning

One of the questions that arises in a decision tree algorithm is the optimal size of the final tree. A tree that is too large risks overfitting the training data and poorly generalizing to new samples. A small tree might not capture important structural information about the sample space. However, it is hard to tell when a tree algorithm should stop because it is impossible Before and After pruning to tell if the addition of a single extra node will dramatically decrease error. This problem is known as the horizon effect. A common strategy is to grow the tree until each node contains a small number of instances then use pruning to remove nodes that do not provide additional information. Pruning should reduce the size of a learning tree without reducing predictive accuracy as measured by a cross-

validation set. There are many techniques for tree pruning that differ in the measurement that is used to optimize performance.

Pruning Techniques:

Pruning processes can be divided into two types: PrePruning & Post Pruning

- Pre-pruning procedures prevent a complete induction of the training set by replacing a stop () criterion in the induction algorithm (e.g. max. Tree depth or information gain (Attr)> minGain). They considered to be more efficient because they do not induce an entire set, but rather trees remain small from the start.
- Post-Pruning (or just pruning) is the most common way of simplifying trees. Here, nodes and subtrees are replaced with leaves to reduce complexity.

The procedures are differentiated on the basis of their approach in the tree: Top-down approach & Bottom-Up approach

Bottom-up pruning approach:

- These procedures start at the last node in the tree (the lowest point).
- Following recursively upwards, they determine the relevance of each individual node.
- If the relevance for the classification is not given, the node is dropped or replaced by a leaf.
- The advantage is that no relevant sub-trees can be lost with this method.
- These methods include Reduced Error Pruning (REP), Minimum Cost Complexity Pruning (MCCP), or Minimum Error Pruning (MEP).

Top-down pruning approach:

- In contrast to the bottom-up method, this method starts at the root of the tree. Following the structure below, a relevance check is carried out which decides whether a node is relevant for the classification of all n items or not.
- By pruning the tree at an inner node, it can happen that an entire sub-tree (regardless of its relevance) is dropped.
- One of these representatives is pessimistic error pruning (PEP), which brings quite good results with unseen items.

CHAID:

- CHAID stands for CHI-squared Automatic Interaction Detector. Morgan and Sonquist (1963) proposed a simple method for fitting trees to predict a quantitative variable.
- Each predictor is tested for splitting as follows: sort all the n cases on the predictor and examine all n-1 ways to split the cluster in two. For each possible split, compute the within-cluster sum of squares about the mean of the cluster on the dependent variable.
- Choose the best of the n-1 splits to represent the predictor's contribution. Now do this for every other predictor. For the actual split, choose the predictor and its cut point which yields the smallest overall within-cluster sum of squares. Categorical predictors require a different approach. Since categories are unordered, all possible splits between categories must be considered. For deciding on one split of k categories into two groups, this means that 2k-1 possible splits must be considered. Once a split is found, its suitability is measured on the same within-cluster sum of squares as for a quantitative predictor.
- It has to do instead with conditional discrepancies. In the analysis of variance, interaction means that a trend within one level of a variable is not parallel to a trend within another level of the same variable. In the ANOVA model, interaction is represented by cross-products between predictors.
- In the tree model, it is represented by branches from the same nodes which have different splitting predictors further down the tree. Regression trees parallel regression/ANOVA modeling, in which the dependent variable is quantitative. Classification trees parallel discriminant analysis and algebraic classification methods. Kass (1980) proposed a modification to AID called CHAID for categorized dependent and independent variables. His algorithm incorporated a sequential merge and split procedure based on a chi-square test statistic.
- Kass's algorithm is like sequential cross-tabulation. For each predictor:
    1) cross tabulate the m categories of the predictor with the k categories of the dependent variable.
    2) find the pair of categories of the predictor whose 2xk sub-table is least significantly different on a chi-square test and merge these two categories.
    3) if the chi-square test statistic is not "significant" according to a preset critical value, repeat this merging process for the selected predictor until no non-significant chi-square is found for a sub-table, and pick the predictor variable

whose chi-square is largest and split the sample into subsets, where l is the number of categories resulting from the merging process on that predictor.

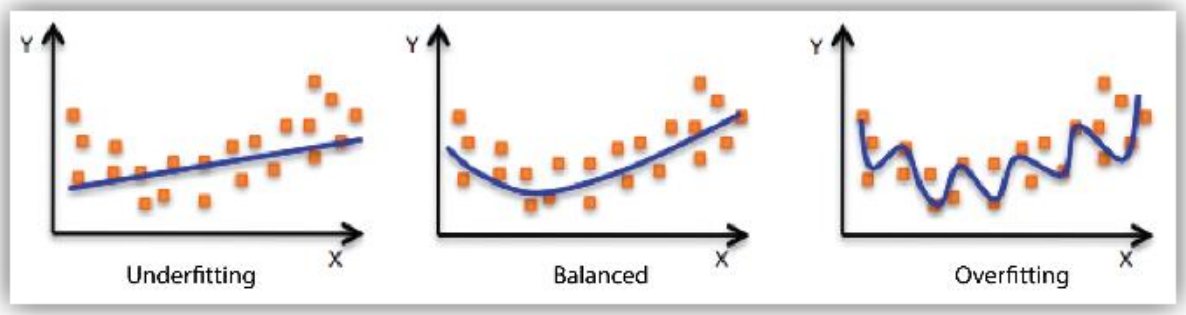
4) Continue splitting, as with AID, until no "significant" chi-squares result. The CHAID algorithm saves some computer time, but it is not guaranteed to find the splits which predict best at a given step.

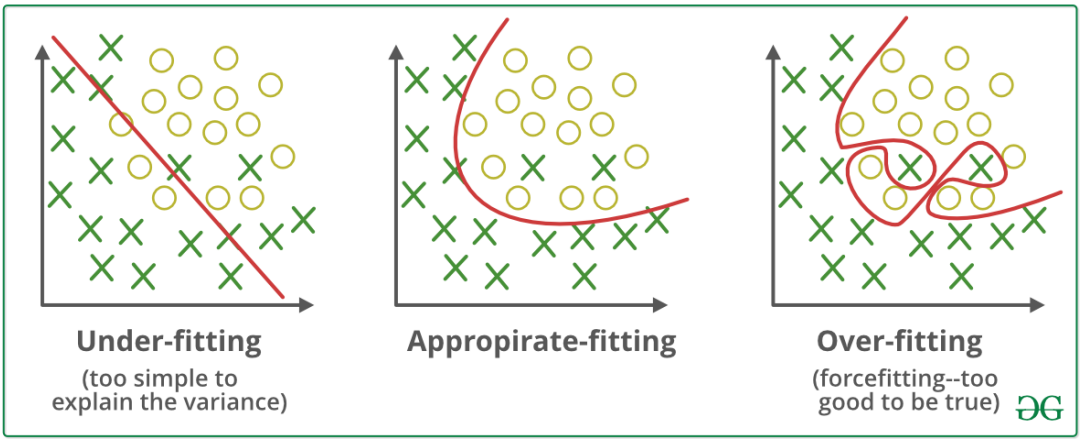
- Only by searching all possible category subsets can we do that. CHAID is also limited to categorical predictors, so it cannot be used for quantitative or mixed categorical quantitative models.

GINI Index Impurity Measure:

- GINI Index Used by the CART (classification and regression tree) algorithm, Gini impurity is a measure of how often a randomly chosen element from the set would be incorrectly labeled if it were randomly labeled according to the distribution of labels in the subset. Gini impurity can be computed by summing the probability fi of each item being chosen times the probability 1-fi of a mistake in categorizing that item.

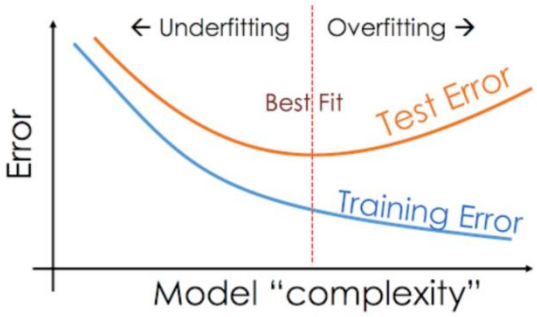
Overfitting and Underfitting





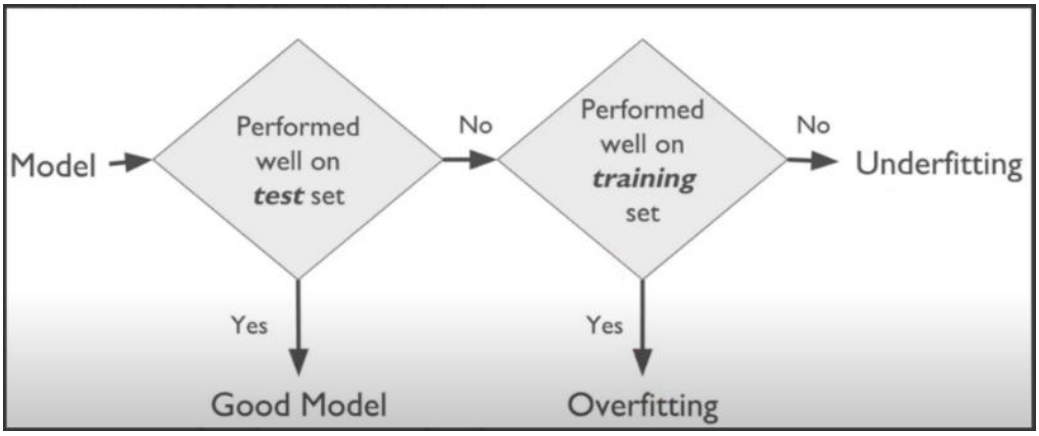
- Let's clearly understand overfitting, underfitting and perfectly fit models.

- From the three graphs shown above, one can clearly understand that the leftmost figure line does not cover all the data points, so we can say that the model is under-fitted. In this case, the model has failed to generalize the pattern to the new dataset, leading to poor performance on testing. The under-fitted model can be easily seen as it gives very high errors on both training and testing data. This is because the dataset is not clean and contains noise, the model has High Bias, and the size of the training data is not enough.
- When it comes to the overfitting, as shown in the rightmost graph, it shows the model is covering all the data points correctly, and you might think this is a perfect fit. But actually, no, it is not a good fit! Because the model learns too many details from the dataset, it also considers noise. Thus, it negatively affects the new data set; not every detail that the model has learned during training needs also apply to the new data points, which gives a poor performance on testing or validation



dataset. This is because the model has trained itself in a very complex manner and has high variance.

- The best fit model is shown by the middle graph, where both training and testing (validation) loss are minimum, or we can say training and testing accuracy should be near each other and high in value.



Time Series Methods:

- Time series forecasting focuses on analyzing data changes across equally spaced time intervals.
- Time series analysis is used in a wide variety of domains, ranging from econometrics to geology and earthquake prediction; it's also used in almost all applied sciences and engineering.
- Time-series databases are highly popular and provide a wide spectrum of numerous applications such as stock market analysis, economic and sales forecasting, budget analysis, to name a few.
- They are also useful for studying natural phenomena like atmospheric pressure, temperature, wind speeds, earthquakes, and medical prediction for treatment.
- Time series data is data that is observed at different points in time. Time Series Analysis finds hidden patterns and helps obtain useful insights from the time series data.
- Time Series Analysis is useful in predicting future values or detecting anomalies from the data. Such analysis typically requires many data points to be present in the dataset to ensure consistency and reliability.
- The different types of models and analyses that can be created through time series analysis are:
    - Classification: To Identify and assign categories to the data.
    - Curve fitting: Plot the data along a curve and study the relationships of variables present within the data.
    - Descriptive analysis: Help Identify certain patterns in time-series data such as trends, cycles, or seasonal variation.
    - Explanative analysis: To understand the data and its relationships, the dependent features, and cause and effect and its tradeoff.
    - Exploratory analysis: Describe and focus on the main characteristics of the time series data, usually in a visual format.
    - Forecasting: Predicting future data based on historical trends. Using the historical data as a model for future data and predicting scenarios that could happen along with the future plot points.
    - Intervention analysis: The Study of how an event can change the data.
    - Segmentation: Splitting the data into segments to discover the underlying properties from the source information.

Components of Time Series:

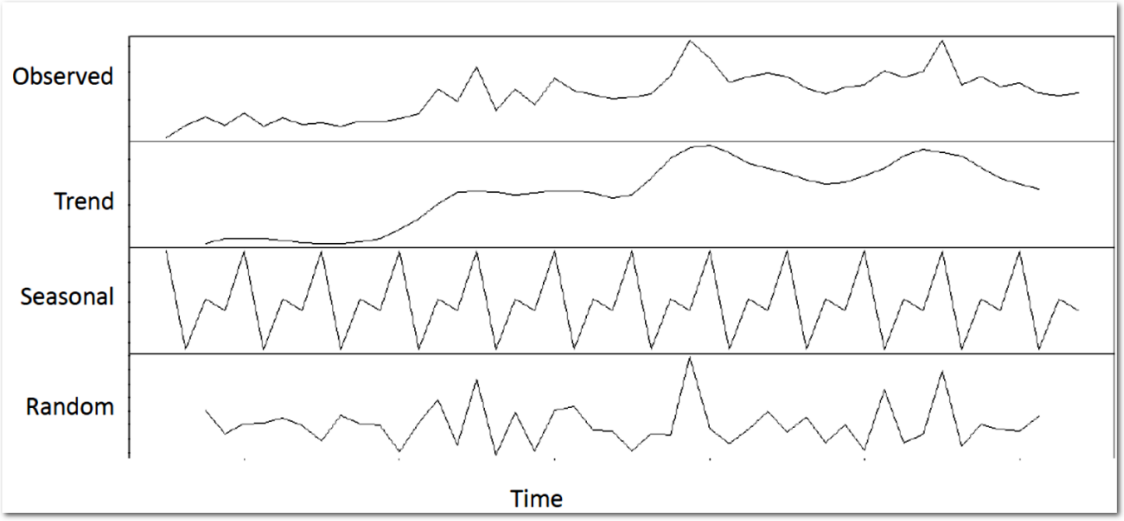
Long term trend – The smooth long term direction of time series where the data can increase or decrease in some pattern.

Seasonal variation – Patterns of change in a time series within a year which tends to repeat every year.

Cyclical variation – Its much alike seasonal variation but the rise and fall of time series over periods are longer than one year.

Irregular variation – Any variation that is not explainable by any of the three above mentioned components. They can be classified into – stationary and non – stationary variation.

Stationary Variation: When the data neither increases nor decreases, i.e. it's completely random it's called stationary variation. Or When the data has some explainable portion remaining and can be analyzed further then such case is called non – stationary variation.



ARIMA & ARMA:

What is ARIMA?

- In time series analysis, ARIMA is an acronym that stands for AutoRegressive Integrated Moving Average model is ageneralization of an autoregressive moving average (ARMA) model. These models are fitted to time series data either to better understand the data or to predict future points in the series (forecasting).
- They are applied in some cases where data show evidence of non-stationary,
- .A popular and very widely used statistical method for time series forecasting and analysis is the ARIMA model.

- It is a class of models that capture a spectrum of different standard temporal structures present in time series data. By implementing an ARIMA model, you can forecast and analyze a time series using past values, such as predicting future prices based on historical earnings.
- Univariate models such as these are used to understand better a single time-dependent variable present in the data, such as temperature over time. They predict future data points of and from the variables.
- wherean initial differencing step (corresponding to the "integrated" part of the model) can be applied to reduce the non-stationary.A standard notation used for describing ARIMA is by parameters p,d and q.
- Non-seasonal ARIMA models are generally denoted ARIMA(p, d, q) where parameters p, d, and q are non-negative integers, p is the order of the Autoregressive model, d is the degree of differencing, and q is the order of the Moving-average model.
- The parameters are substituted with an integer value to indicate the specific ARIMA model being used quickly. The parameters of the ARIMA model are further described as follows:

    - p: Stands for the number of lag observations included in the model, also known as the lag order.
    - d: The number of times the raw observations are differentiated, also called the degree of differencing.
    - q: Is the size of the moving average window and also called the order of moving average.

Univariate stationary processes (ARMA)

A covariance stationary process is an ARMA (p, q) process of autoregressive order p and moving

average order q if it can be written as

The acronym ARIMA stands for Auto-Regressive Integrated Moving Average. Lags of the stationarized series in the forecasting equation are called "autoregressive" terms, lags of the forecast errors are called "moving average" terms, and a time series which needs to be differenced to be made stationary is said to be an "integrated" version of a stationary series. Random-walk and random-trend models, autoregressive models, and exponential smoothing models are all special cases of ARIMA models.

A nonseasonal ARIMA model is classified as an "ARIMA(p,d,q)" model, where:

- p is the number of autoregressive terms,
- d is the number of nonseasonal differences needed for stationarity, and
- q is the number of lagged forecast errors in the prediction equation.

The forecasting equation is constructed as follows. First, let y denote the d^{th} difference of Y, which means:

If d=0: $y_t = Y_t$

If d=1: $y_t = Y_t - Y_{t-1}$

If d=2: $y_t = (Y_t - Y_{t-1}) - (Y_{t-1} - Y_{t-2}) = Y_t - 2Y_{t-1} + Y_{t-2}$

Note that the second difference of Y (the d=2 case) is not the difference from 2 periods ago. Rather, it is the *first-difference-of-the-first difference*, which is the discrete analog of a second derivative, i.e., the local acceleration of the series rather than its local trend.

In terms of y, the general forecasting equation is:

$\hat{y}_t = \mu + \varphi_1 y_{t-1} + \ldots + \varphi_p y_{t-p} - \theta_1 e_{t-1} - \ldots - \theta_q e_{t-q}$

Measure of Forecast Accuracy: Forecast Accuracy can be defined as the deviation of Forecast or Prediction from the actual results.

Error = Actual demand – Forecast OR $e_i = A_t - F_t$

We measure Forecast Accuracy by 2 methods : 1. Mean Forecast Error (MFE) For n time periods where we have actual demand and forecast values:

$$MFE = \frac{\sum_{i=1}^{n}(e_i)}{n}$$

Ideal value = 0; MFE > 0, model tends to under-forecast MFE < 0, model tends to over-forecast 2. Mean Absolute Deviation (MAD) For n time periods where we have actual demand and forecast values:

$$MAD = \frac{\sum_{i=1}^{n}(e_i)}{n}$$

While MFE is a measure of forecast model bias, MAD indicates the absolute size of the errors

Uses of Forecast error:

- Forecast model bias
- Absolute size of the forecast errors
- Compare alternative forecasting models

- Identify forecast models that need adjustment

ETL Approach:

Extract, Transform and Load (ETL) refers to a process in database usage and especially in data warehousing that:

- Extracts data from homogeneous or heterogeneous data sources
- Transforms the data for storing it in proper format or structure for querying and analysis purpose
- Loads it into the final target (database, more specifically, operational data store, data mart, or data warehouse)

Usually all the three phases execute in parallel since the data extraction takes time, so while the data is being pulled another transformation process executes, processing the already received data and prepares the data for loading and as soon as there is some data ready to be loaded into the target, the data loading kicks off without waiting for the completion of the previous phases.

ETL systems commonly integrate data from multiple applications (systems), typically developed and supported by different vendors or hosted on separate computer hardware. The disparate systems containing the original data are frequently managed and operated by different employees. For example, a cost accounting system may combine data from payroll, sales, and purchasing.

Commercially available ETL tools include:

- Anatella
- Alteryx
- CampaignRunner
- ESF Database Migration Toolkit
- InformaticaPowerCenter
- Talend
- IBM InfoSphereDataStage
- Ab Initio
- Oracle Data Integrator (ODI)
- Oracle Warehouse Builder (OWB)
- Microsoft SQL Server Integration Services (SSIS)
- Tomahawk Business Integrator by Novasoft Technologies.
- Pentaho Data Integration (or Kettle) opensource data integration framework
- Stambia

- Diyotta DI-SUITE for Modern Data Integration
- FlyData
- Rhino ETL
- SAP Business Objects Data Services
- SAS Data Integration Studio
- SnapLogic
- Clover ETL opensource engine supporting only basic partial functionality and not server
- SQ-ALL - ETL with SQL queries from internet sources such as APIs
- North Concepts Data Pipeline

There are various steps involved in ETL. They are as below in detail:

Extract:

The Extract step covers the data extraction from the source system and makes it accessible for further processing. The main objective of the extract step is to retrieve all the required data from the source system with as little resources as possible. The extract step should be designed in a way that it does not negatively affect the source system in terms or performance, response time or any kind of locking.

There are several ways to perform the extract:

- Update notification - if the source system is able to provide a notification that a record has been changed and describe the change, this is the easiest way to get the data.
- Incremental extract - some systems may not be able to provide notification that an update has occurred, but they are able to identify which records have been modified and provide an extract of such records. During further ETL steps, the system needs to identify changes and propagate it down. Note, that by using daily extract, we may not be able to handle deleted records properly.
- Full extract - some systems are not able to identify which data has been changed at all, so a full extract is the only way one can get the data out of the system. The full extract requires keeping a copy of the last extract in the same format in order to be able to identify changes. Full extract handles deletions as well.
- When using Incremental or Full extracts, the extract frequency is extremely important. Particularly for full extracts; the data volumes can be in tens of gigabytes.

- Clean: The cleaning step is one of the most important as it ensures the quality of the data in the data warehouse. Cleaning should perform basic data unification rules, such as:
- Making identifiers unique (sex categories Male/Female/Unknown, M/F/null, Man/Woman/Not Available are translated to standard Male/Female/Unknown)
- Convert null values into standardized Not Available/Not Provided valueConvert phone numbers, ZIP codes to a standardized form
- Validate address fields, convert them into proper naming, e.g. Street/St/St./Str./Str
- Validate address fields against each other (State/Country, City/State, City/ZIP code, City/Street).

Transform:

- The transform step applies a set of rules to transform the data from the source to the target.
- This includes converting any measured data to the same dimension (i.e. conformed dimension) using the same units so that they can later be joined.
- The transformation step also requires joining data from several sources, generating aggregates, generating surrogate keys, sorting, deriving new calculated values, and applying advanced validation rules.

Load:

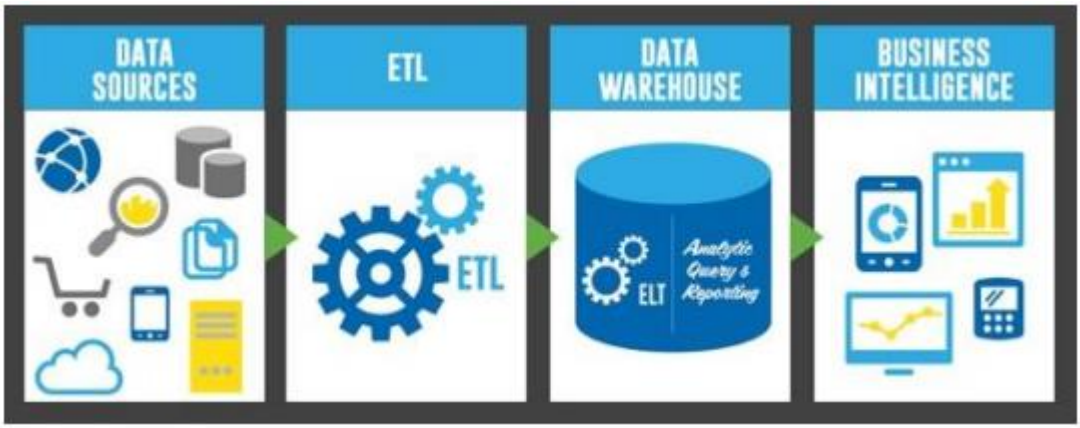
- During the load step, it is necessary to ensure that the load is performed correctly and with as little resources as possible. The target of the Load process is often a database.
- In order to make the load process efficient, it is helpful to disable any constraints and indexes before the load and enable them back only after the load completes. The referential integrity needs to be maintained by ETL tool to ensure consistency.

Managing ETL Process

The ETL process seems quite straight forward. As with every application, there is a possibility that the ETL process fails. This can be caused by missing extracts from one of the systems, missing values in one of the reference tables, or simply a connection or power outage. Therefore, it is necessary to design the ETL process keeping fail-recovery in mind.

Staging:

It should be possible to restart, at least, some of the phases independently from the others. For example, if the transformation step fails, it should not be necessary to restart the Extract step. We can ensure this by implementing proper staging. Staging means that the data is simply dumped to the location (called the Staging Area) so that it can then be read by the next processing phase. The staging area is also used during ETL process to store intermediate results of processing. This is ok for the ETL process which uses for this purpose. However, the staging area should be accessed by the load ETL process only. It should never be available to anyone else; particularly not to end users as it is not intended for data presentation to the end-user. May contain incomplete or in-the-middle-of-the-processing data.



*** End of Unit-4 ***

Decision Tree Example
for
DATA ANALYTICS
(UNIT 4)

Hypothesis space search in decision tree learning:

In order to derive the Hypothesis space, we compute the Entropy and Information Gain of Class and attributes. For them we use the following statistics formulae:

Entropy of Class is:

$$Entropy(Class) = -\frac{P}{P+N}\log_2\left(\frac{P}{P+N}\right) - \frac{N}{P+N}\log_2\left(\frac{N}{P+N}\right)$$

For any Attribute,

$$InformationGain(Attribute)$$

$$I(\mathrm{p}_i, n_i) = -\frac{\mathrm{p}_i}{\mathrm{p}_i+n_i}\log_2\left(\frac{\mathrm{p}_i}{\mathrm{p}_i+n_i}\right) - \frac{n_i}{\mathrm{p}_i+n_i}\log_2\left(\frac{n_i}{\mathrm{p}_i+n_i}\right)$$

Entropy of an Attribute is:

$$Entropy(Attribute) = \frac{\sum(\mathrm{p}_i+n_i)}{P+N} I\left(\mathrm{p}_i+n_i\right)$$

$$Gain = Entropy(Class) - Entropy(Attribute)$$

Illustrative Example:

Concept: "Play Tennis":

Data set:

Day	Outlook	Temperature	Humidity	Wind	Play Golf
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

By using the said formulae, the decision tree is derived as follows:

Pg1

Concept:- Play tennis

Day	outlook	temperature	Humidity	wind	Play tennis
D_1	Sunny	Hot	High	weak	NO
D_2	Sunny	Hot	High	strong	NO
• D_3	Overcast	Hot	High	weak	yes
• D_4	Rain	Mild	High	weak	yes
• D_5	Rain	Cool	Normal	weak	yes
D_6	Rain	Cool	Normal	strong	NO
• D_7	Overcast	Cool	Normal	strong	yes
D_8	Sunny	Mild	High	weak	NO
• D_9	Sunny	Cool	Normal	weak	yes
• D_{10}	Rain	Mild	Normal	weak	yes
• D_{11}	Sunny	Mild	Normal	strong	yes
• D_{12}	Overcast	Mild	High	strong	yes
• D_{13}	Overcast	Hot	Normal	weak	yes
D_{14}	Rain	Mild	High	strong	NO

Entrophy of the class:-

$$\text{Entrophy(class)} = \frac{-P}{P+N}\log_2\left(\frac{P}{P+N}\right) - \frac{N}{P+N}\log_2\left(\frac{N}{P+N}\right)$$

$P=9, N=5$

$$= \frac{-9}{9+5}\log_2\left(\frac{9}{9+5}\right) - \frac{5}{9+5}\log_2\left(\frac{5}{9+5}\right)$$

$$= 0.40977 + 0.53050$$

$$= 0.94027 \simeq 0.94$$

pg2

Entropy of each attribute:-

1) Outlook attribute :-

Outlook	P_i	N_i	$I(P_i, N_i)$
Sunny ($D_1, D_2, D_8, D_9, D_{11}$)	2	3	0.971
Overcast (D_3, D_7, D_{12}, D_{13})	4	0	0
Rain ($D_4, D_5, D_6, D_{10}, D_{14}$)	3	2	0.971

Information gain for Sunny:-

$$I(P_i, N_i) = \frac{-P_i}{P_i+N_i} \log_2\left(\frac{P_i}{P_i+N_i}\right) - \frac{N_i}{P_i+N_i} \log_2\left(\frac{N_i}{P_i+N_i}\right)$$

$$= \frac{-2}{2+3} \log_2\left(\frac{2}{2+3}\right) - \frac{3}{2+3} \log_2\left(\frac{3}{2+3}\right)$$

$$= 0.52877 + 0.44217$$

$$= 0.97094$$

$$I(2,3) \simeq 0.971$$

Information gain for overcast:-

$$I(P_i, N_i) = \frac{-P_i}{P_i+N_i} \log_2\left(\frac{P_i}{P_i+N_i}\right) - \frac{N_i}{P_i+N_i} \log_2\left(\frac{N_i}{P_i+N_i}\right)$$

$$= \frac{-4}{4+0} \log_2\left(\frac{4}{4+0}\right) - \frac{0}{4+0} \log_2\left(\frac{0}{4+0}\right)$$

$$I(4,0) = 0$$

Information gain for Rain:-

$$I(P_i,N_i) = \frac{-P_i}{P_i+N_i}\log_2\left(\frac{P_i}{P_i+N_i}\right) - \frac{N_i}{P_i+N_i}\log_2\left(\frac{N_i}{P_i+N_i}\right)$$

$$= \frac{-3}{3+2}\log_2\left(\frac{3}{3+2}\right) - \frac{2}{2+3}\log_2\left(\frac{2}{3+2}\right)$$

$$= 0.44217 + 0.52877$$

$$= 0.97094$$

$$I(3,2) \simeq 0.971$$

Entropy of outlook:-

$$\text{Entropy(outlook)} = \frac{\Sigma P_i+N_i}{P+N}\left(I(P_i,N_i)\right)$$

$$= \frac{2+3}{9+5}\times 0.971 + \frac{4+0}{9+5}\times 0 + \frac{3+2}{9+5}\times 0.971$$

$$= \frac{5}{14}\times 0.971 + \frac{5}{14}\times 0.971$$

$$\text{Entropy(outlook)} = 0.693571$$

Gain of outlook:-

$$\text{Gain(outlook)} = \text{Entropy(class)} - \text{Entropy(outlook)}$$

$$= 0.940 - 0.693571$$

$$\text{Gain(outlook)} = 0.246$$

2) Temperature attribute:-

Temperature	P_i	N_i	$I(P_i,N_i)$
Hot (D_1, D_2, D_3, D_{13})	2	2	1
Mild ($D_4, D_8, D_{10}, D_{11}, D_{12}, D_{14}$)	4	2	0.92
Cool (D_5, D_6, D_7, D_9)	3	1	0.81

Information gain for Hot:-

$$I(P_i,N_i) = \frac{-P_i}{P_i+N_i}\log_2\left(\frac{P_i}{P_i+N_i}\right) - \frac{N_i}{P_i+N_i}\log_2\left(\frac{N_i}{P_i+N_i}\right)$$

$$= \frac{-2}{2+2}\log_2\left(\frac{2}{2+2}\right) - \frac{2}{2+2}\log_2\left(\frac{2}{2+2}\right)$$

$$= 0.5+0.5$$

$$I(2,2) = 1$$

Information gain for Mild:-

$$I(P_i,N_i) = \frac{-P_i}{P_i+N_i}\log_2\left(\frac{P_i}{P_i+N_i}\right) - \frac{N_i}{P_i+N_i}\log_2\left(\frac{N_i}{P_i+N_i}\right)$$

$$= \frac{-4}{4+2}\log_2\left(\frac{4}{4+2}\right) - \frac{2}{4+2}\log_2\left(\frac{2}{4+2}\right)$$

$$I(4,2) = 0.91829$$

Information gain for cool:-

$$I(P_i,N_i) = \frac{-P_i}{P_i+N_i}\log_2\left(\frac{P_i}{P_i+N_i}\right) - \frac{N_i}{P_i+N_i}\log_2\left(\frac{N_i}{P_i+N_i}\right)$$

$$= \frac{-3}{3+1}\log_2\left(\frac{3}{3+1}\right) - \frac{1}{3+1}\log_2\left(\frac{1}{3+1}\right)$$

$$I(3,1) = 0.81127$$

Entropy of temperature:-

$$\text{Entropy(temperature)} = \sum \frac{P_i+N_i}{P+N}\left(I(P_i,N_i)\right)$$

$$= \frac{2+2}{9+5}\times 1 + \frac{4+2}{9+5}\times 0.92 + \frac{3+1}{9+5}\times 0.81$$

$$\text{Entropy(temperature)} = 0.9114$$

Gain of temperature:-

$$\text{Gain(temperature)} = \text{Entropy(class)} - \text{Entropy(temperature)}$$

(2)

pg5

$= 0.94 - 0.9114$

Gain (temperature) = 0.028

3) Humidity attribute:-

Humidity	P_i	N_i	$I(P_i, N_i)$
High ($D_1, D_2, D_3, D_4, D_8, D_{12}, D_{14}$)	3	4	0.985
Normal ($D_5, D_6, D_7, D_9, D_{10}, D_{11}, D_{13}$)	6	1	0.591

Information gain for High:-

$$I(P_i, N_i) = -\frac{P_i}{P_i+N_i}\log_2\left(\frac{P_i}{P_i+N_i}\right) - \frac{N_i}{P_i+N_i}\log_2\left(\frac{N_i}{P_i+N_i}\right)$$

$$= \frac{-3}{3+4}\log_2\left(\frac{3}{3+4}\right) - \frac{4}{4+3}\log_2\left(\frac{4}{3+4}\right)$$

I(3,4) = 0.985

Information gain for Normal:-

$$I(P_i, N_i) = -\frac{P_i}{P_i+N_i}\log_2\left(\frac{P_i}{P_i+N_i}\right) - \frac{N_i}{P_i+N_i}\log_2\left(\frac{N_i}{P_i+N_i}\right)$$

$$= -\frac{6}{6+1}\log_2\left(\frac{6}{6+1}\right) - \frac{1}{6+1}\log_2\left(\frac{1}{6+1}\right)$$

I(6,1) = 0.591

Entropy of Humidity:-

$$\text{Entropy(Humidity)} = \sum \frac{P_i+N_i}{P+N}\left(I(P_i, N_i)\right)$$

$$= \frac{3+4}{9+5} \times 0.95 + \frac{6+1}{9+5} \times 0.591$$

Entropy (Humidity) = 0.7884

10

Gain of Humidity:-

$$\text{Gain} = \text{Entropy(class)} - \text{Entropy(attribute)}$$

$$= 0.94 - 0.7884$$

$$\text{Gain} = 0.1515$$

4) Wind attribute:-

wind	P_i	N_i	$I(P_i, N_i)$
weak ($D_1, D_3, D_4, D_5, D_8, D_9, D_{10}, D_{13}$)	6	2	0.8113
Strong ($D_2, D_6, D_7, D_{11}, D_{12}, D_{14}$)	3	3	1

Information gain for weak:-

$$I(P_i, N_i) = -\frac{P_i}{P_i+N_i}\log_2\left(\frac{P_i}{P_i+N_i}\right) - \frac{N_i}{P_i+N_i}\log_2\left(\frac{N_i}{P_i+N_i}\right)$$

$$= \frac{-6}{6+2}\log_2\left(\frac{6}{6+2}\right) - \frac{2}{6+2}\log_2\left(\frac{2}{6+2}\right)$$

$$I(P_i, N_i) = 0.8113$$

Information gain for strong:-

$$I(P_i, N_i) = -\frac{P_i}{P_i+N_i}\log_2\left(\frac{P_i}{P_i+N_i}\right) - \frac{N_i}{P_i+N_i}\log_2\left(\frac{N_i}{P_i+N_i}\right)$$

$$= \frac{-3}{3+3}\log_2\left(\frac{3}{3+3}\right) - \frac{3}{3+3}\log_2\left(\frac{3}{3+3}\right)$$

$$I(3,3) = 1$$

Entropy of wind:-

$$\text{Entropy(wind)} = \sum \frac{P_i+N_i}{P+N}\left(I(P_i, N_i)\right)$$

$$= \frac{6+2}{9+5} \times 0.8113 + \frac{3+3}{9+5} \times 1$$

pg7 (3)

Entropy(wind) = 0.8922

Gain of wind:-

Gain = Entropy(class) - Entropy(wind)

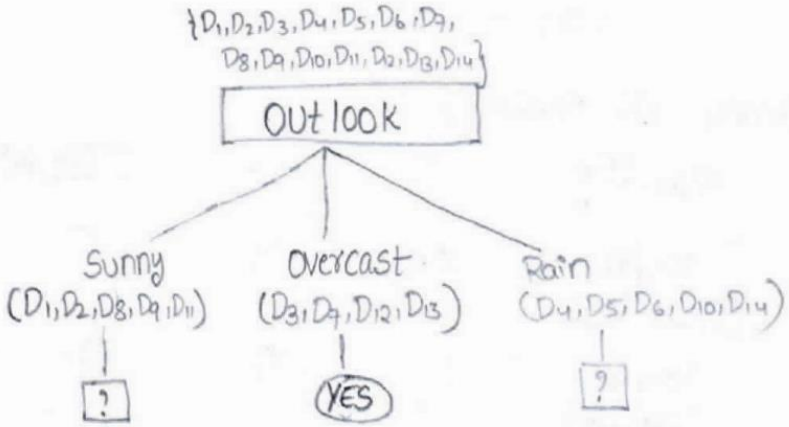
= 0.94 - 0.8922

Gain = 0.0478

Thus we have

Gain (S, outlook) = 0.246
Gain (S, temperature) = 0.029
Gain (S, Humidity) = 0.151
Gain(S, wind) = 0.048

∴ node is out look



Ssunny attribute:-

1) Sunny with temperature:-

Temperature	Pi	Ni	I(Pi, Ni)
Hot (D1, D2)	0	2	0
Mild (D8, D11)	1	1	1
Cool (D9)	1	0	0

Information gain of Mild :-

$$I(P_i,N_i) = -\frac{P_i}{P_i+N_i}\log_2\left(\frac{P_i}{P_i+N_i}\right) - \frac{N_i}{P_i+N_i}\log_2\left(\frac{N_i}{P_i+N_i}\right)$$

$$= -\frac{1}{1+1}\log_2\left(\frac{1}{1+1}\right) - \frac{1}{1+1}\log_2\left(\frac{1}{1+1}\right)$$

$$= 1$$

Entropy of temperature with sunny :-

$$\text{Entropy(temperature)} = \sum \frac{P_i+N_i}{P+N}\left(I(P_i,N_i)\right)$$

$$= \frac{0+2}{2+3}\times 0 + \frac{1+1}{2+3}\times 1 + \frac{1+0}{2+3}\times 0$$

$$\text{Entropy(temperature)} = 0.4$$

Gain of temperature with sunny :-

$$\text{Gain} = \text{Entropy(Sunny)} - \text{Entropy(temperature)}$$

$$= 0.971 - 0.4$$

$$\text{Gain} = 0.5$$

2) Ssunny with Humidity :-

Humidity	P_i	N_i	$I(P_i,N_i)$
High (D_1, D_2, D_8)	0	3	0
Normal (D_9, D_{11})	2	0	0

Entropy of Humidity with sunny :-

$$\text{Entropy(Humidity)} = \sum \frac{P_i+N_i}{P+N}\left(I(P_i,N_i)\right)$$

$$= \frac{0+3}{2+3}\times 0 + \frac{2+0}{2+3}\times 0$$

$$\text{Entropy(Humidity)} = 0$$

Gain of Humidity with Sunny:-

$$\text{Gain} = \text{Entropy}(\text{Sunny}) - \text{Entropy}(\text{Humidity})$$

$$= 0.971 - 0$$

$$\text{Gain} = 0.971$$

3) SSunny with Wind:-

Wind	Pi	Ni	I(Pi, Ni)
Weak (D_1, D_8, D_9)	1	2	0.9182
Strong (D_2, D_{11})	1	1	1

Information gain for weak:-

$$I(P_i, N_i) = \frac{-P_i}{P_i+N_i} \log_2\left(\frac{P_i}{P_i+N_i}\right) - \frac{N_i}{P_i+N_i} \log_2\left(\frac{P_i}{P_i+N_i}\right)$$

$$= \frac{-1}{1+2} \log_2\left(\frac{1}{1+2}\right) - \frac{2}{1+2} \log_2\left(\frac{2}{1+2}\right)$$

$$I(1,2) = 0.9182$$

Entropy of wind with Sunny:-

$$\text{Entropy}(\text{Wind}) = \sum \frac{P_i+N_i}{P+N} \left(I(P_i, N_i)\right)$$

$$= \frac{1+2}{2+3} \times 0.9182 + \frac{1+1}{2+3} \times 1$$

$$= 0.9509$$

Gain of wind with Sunny:-

$$\text{Gain} = \text{Entropy}(\text{Sunny}) - \text{Entropy}(\text{Wind})$$

$$= 0.971 - 0.9509$$

$$\text{Gain} = 0.02008$$

10

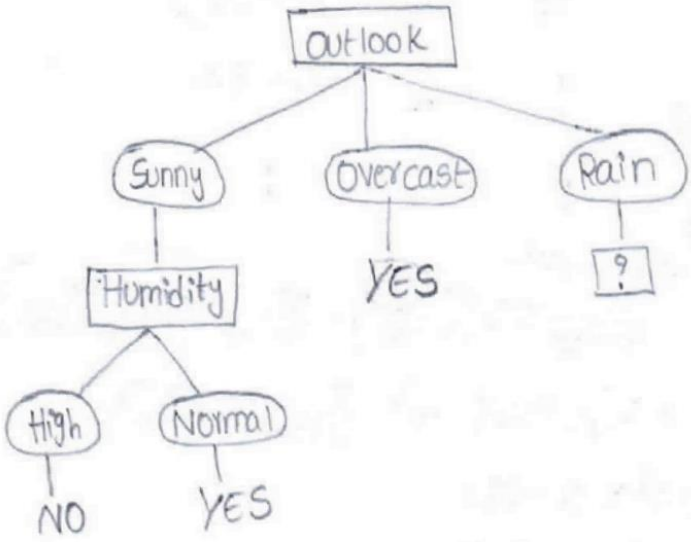
Thus we have

Gain (Sunny, temperature) = 0.5

Gain (Sunny, Humidity) = 0.971

Gain (Sunny, wind) = 0.02008

∴ node is Humidity



SRain:-

1) SRain with temperature:-

Temperature	Pi	Ni	I(Pi,Ni)
Hot	0	0	0
Mild (D4, D10, D14)	2	1	0.9182
Cool (D5, D6)	1	1	1

Entropy of temperature with SRain:-

$$\text{Entropy (temperature)} = \sum \frac{P_i+N_i}{P+N} \left(I(P_i,N_i)\right)$$

$$= \frac{2+1}{3+2} \times 0.9182 + \frac{1+1}{3+2}$$

Entropy(temperature) = 0.95092

Gain of temperature with SRain:-

Gain = Entropy(Rain) − Entropy(temperature)

= 0.971 − 0.95092

Gain = 0.02008

2) SRain with Humidity:-

Humidity	Pi	Ni	I(Pi,Ni)
High (D4, D14)	1	1	1
Normal (D5, D6, D10)	2	1	0.9182

Entropy of Humidity with SRain:-

$$\text{Entropy(Humidity)} = \sum \frac{Pi+Ni}{P+N} (I(Pi,Ni))$$

$$= \frac{1+1}{3+2} \times 1 + \frac{2+1}{3+2} \times 0.9182$$

Entropy(Humidity) = 0.95092

Gain of Humidity with SRain:-

Gain = Entropy(Rain) − Entropy(Humidity)

= 0.971 − 0.95092

Gain = 0.02008

3) SRain with Wind:-

Wind	Pi	Ni	I(Pi,Ni)
Weak (D4, D5, D10)	3	0	0
Strong (D6, D14)	0	2	0

Entropy of wind with SRain:-

$$\text{Entropy(wind)} = \sum \frac{P_i+N_i}{P+N} (I(P_i,N_i))$$

$$= \frac{3+0}{3+2} \times 0 + \frac{0+2}{3+2} \times 0$$

$$\text{Entropy(wind)} = 0$$

Gain of wind with SRain:-

$$\text{Gain} = \text{Entropy(Rain)} - \text{Entropy(wind)}$$

$$= 0.971 - 0$$

$$\text{Gain} = 0.971$$

Thus we have,

Gain (Rain, temperature) = 0.02008

Gain (Rain, Humidity) = 0.02008

Gain (Rain, wind) = 0.971

∴ node is wind

Final decision tree:-