MODULE 04

Decision Trees

Decision trees are a simple way to guide one's path to a decision. The decision may be a simple binary one, whether to approve a loan or not. Or it may be a complex multi-valued decision, as to what may be the diagnosis for a particular sickness. Decision trees are hierarchically branched structures that help one come to a decision based on asking certain questions in a particular sequence.

Decision trees are one of the most widely used techniques for classification. A good decision tree should be short and ask only a few meaningful questions. They are very efficient to use, easy to explain, and their classification accuracy is competitive with other methods. Decision trees can generate knowledge from a few test instances that can then be applied to a broad population. Decision trees are used mostly to answer relatively simple binary decisions.

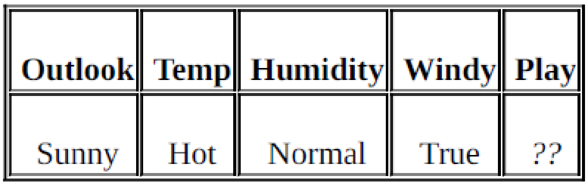
Decision Tree problem

Imagine a conversation between a doctor and a patient. The doctor asks questions to determine the cause of the ailment. The doctor would continue to ask questions, till she is able to arrive at a reasonable decision. If nothing seems plausible, she might recommend some tests to generate more data and options.

This is how experts in any field solve problems. They use decision trees or decision rules. For every question they ask, the potential answers create separate branches for further questioning. For each branch, the expert would know how to proceed ahead. The process continues until the end of the tree is reached, which means a leaf node is reached. Human experts learn from past experiences or data points. Similarly, a machine can be trained to learn from the past data points and extract some knowledge or rules from it. Decision trees use machine learning algorithms to abstract knowledge from data. A decision tree would have a predictive accuracy based on how often it makes correct decisions.

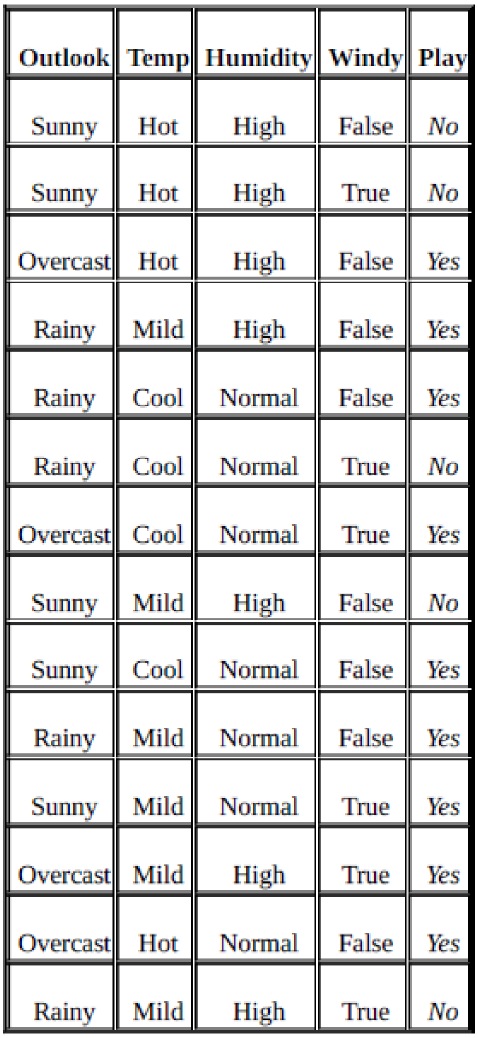
1. The more training data is provided, the more accurate its knowledge extraction will be, and thus, it will make more accurate decisions.
2. The more variables the tree can choose from, the greater is the likely of the accuracy of the decision tree.
3. In addition, a good decision tree should also be frugal so that it takes the least number of questions, and thus, the least amount of effort, to get to the right decision.

Here is an exercise to create a decision tree that helps make decisions about approving the play of an outdoor game. The objective is to predict the play decision given the atmospheric conditions out there. The decision is: Should the game be allowed or not? Here is the decision problem.



Outlook	Temp	Humidity	Windy	Play
Sunny	Hot	Normal	True	??

To answer that question, one should look at past experience, and see what decision was made in a similar instance, if such an instance exists. One could look up the database of past decisions to find the answer and try to come to an answer. Here is a list of the decisions taken in 14 instances of past soccer game situations. (Dataset courtesy: Witten, Frank, and Hall, 2010).



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

If there were a row for Sunny/Hot/Normal/Windy condition in the data table, it would match the current problem; and the decision from that row could be used to answer the current problem. However, there is no such past instance in this case. There are three disadvantages of looking up the data table:

1. As mentioned earlier, how to decide if there isn't a row that corresponds to the exact situation today? If there is no exact matching instance available in the database, the past experience cannot guide the decision.
2. Searching through the entire past database may be time consuming, depending on the number of variables and the organization of the database.
3. What if the data values are not available for all the variables? In this instance, if the data for humidity variable was not available, looking up the past data would not help.

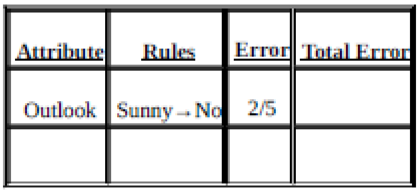
A better way of solving the problem may be to abstract the knowledge from the past data into decision tree or rules. These rules can be represented in a decision tree, and then that tree can be used make the decisions. The decision tree may not need values for all the variables.

Decision Tree Construction

A decision tree is a hierarchically branched structure. What should be the first question asked in creating the tree? One should ask the more important question first, and the less important questions later. What is the most important question that should be asked to solve the problem? How is the importance of the questions determined? Thus, how should the root node of the tree be determined?

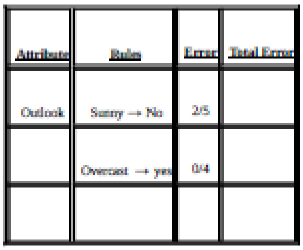
Determining root node of the tree: In this example, there are four choices based on the four variables. One could begin by asking one of the following questions: what is the outlook, what is the temperature, what is the humidity, and what is the wind speed? A criterion should be used to evaluate these choices. The key criterion would be that: which one of these questions gives the most insight about the situation? Another way to look at it would be the criterion of frugality. That is, which question will provide us the shortest ultimate decision tree? Another way to look at this is that if one is allowed to ask one and only one question, which one would one ask? In this case, the most important question should be the one that, by itself, helps make the most correct decisions with the fewest errors. The four questions can now be systematically compared, to see which variable by itself will help make the most correct decisions. One should systematically calculate the correctness of decisions based on each question. Then one can select the question with the most correct predictions, or the fewest errors.

Start with the first variable, in this case outlook. It can take three values, sunny, overcast, and rainy. Start with the sunny value of outlook. There are five instances where the outlook is sunny. In 2 of the 5 instances the play decision was yes, and in the other three, the decision was No. Thus, if the decision rule was that Outlook:sunny → No, then 3 out of 5 decisions would be correct, while 2 out of 5 such decisions would be incorrect. There are 2 errors out of 5. This can be recorded in Row 1.



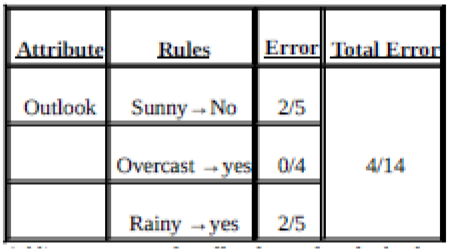
Attribute	Rules	Error	Total Error
Outlook	Sunny → No	2/5	

Similar analysis would be done for other values of the outlook variable. There are four instances where the outlook is overcast. In all 4 out 4 instances the Play decision was yes. Thus, if the decision rule was that Outlook:overcast →Yes, then 4 out of 4 decisions would be correct, while none of decisions would be incorrect. There are 0 errors out of 4. This can be recorded in the next row.



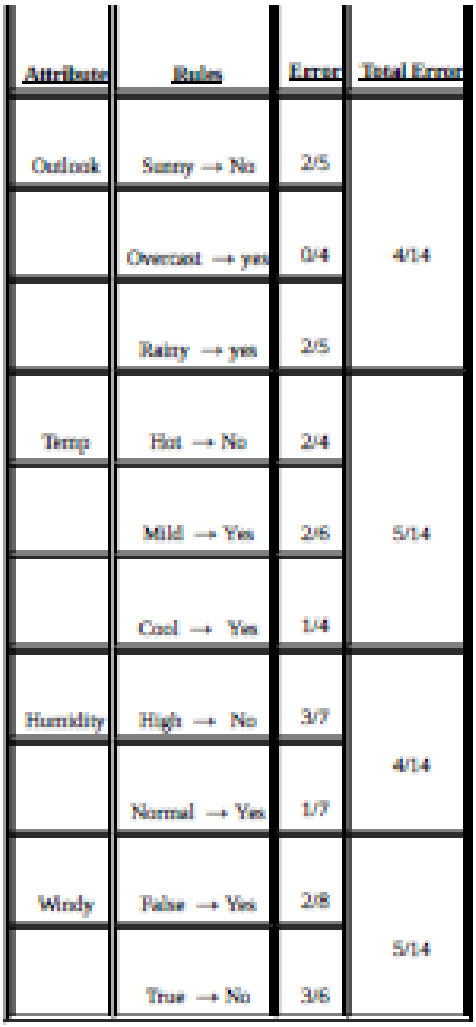
Attribute	Rules	Error	Total Error
Outlook	Sunny → No	2/5	
	Overcast → yes	0/4	

There are five instances where the outlook is rainy. In 3 of the 5 instances the play decision was yes, and in the other three, the decision was no. Thus, if the decision rule was that Outlook:rainy→ Yes, then 3 out of 5 decisions would be correct, while 2 out of 5 decisions would be incorrect. There will be 2/5 errors. This can be recorded in next row.



Attribute	Rules	Error	Total Error
Outlook	Sunny → No	2/5	4/14
	Overcast → yes	0/4	
	Rainy → yes	2/5	

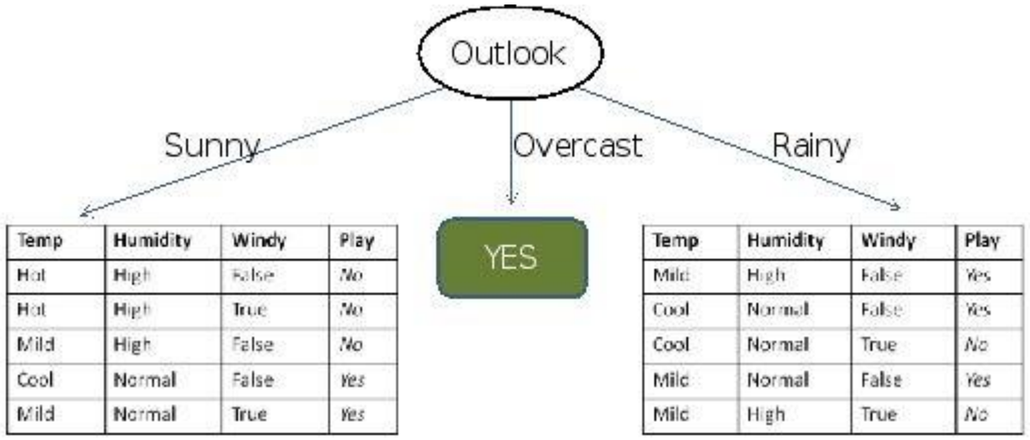
Adding up errors for all values of outlook, there are 4 errors out of 14. In other words, Outlook gives 10 correct decisions out of 14, and 4 incorrect ones.A similar analysis can be done for the other three variables. At the end of that analytical exercise, the following Error table will be constructed.



Attribute	Rules	Error	Total Error
Outlook	Sunny → No	2/5	4/14
	Overcast → yes	0/4	
	Rainy → yes	2/5	
Temp	Hot → No	2/4	5/14
	Mild → Yes	2/6	
	Cool → Yes	1/4	
Humidity	High → No	3/7	4/14
	Normal → Yes	1/7	
Windy	False → Yes	2/8	5/14
	True → No	3/6	

The variable that leads to the least number of errors (and thus the most number of correct decisions) should be chosen as the first node. In this case, two variables have the least number of errors. There is a tie between outlook and humidity, as both have 4 errors out of 14 instances. The tie can be broken using another criterion, the purity of resulting sub-trees. If all the errors were concentrated in a few of the subtrees, and some of the branches were completely free of error, that is preferred from a usability perspective. Outlook has one error-free branch, for the overcast value, while there is no such pure sub-class for humidity variable. Thus the tie is broken in favor of outlook. The decision tree will use outlook as the first node, or the first splitting variable. The first question that should be asked to solve the Play problem, is 'What is the value of outlook'?

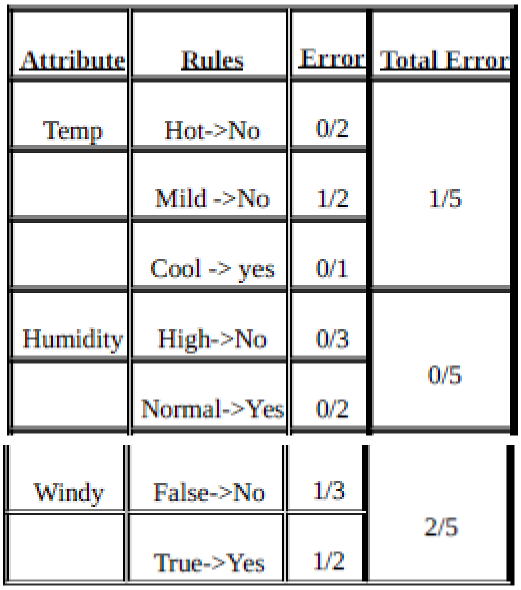
Splitting the Tree: From the root node, the decision tree will be split into three branches or sub-rees, one for each of the three values of outlook. Data for the root node (the entire data) will be divided into the three segments, one for each of the value of outlook. The sunny branch will inherit the data for the instances that had sunny as the value of outlook. These will be used for further building of that sub-tree. Similarly, the rainy branch will inherit data for the instances that had rainy as the value of outlook. These will be used for further building of that sub-tree. The overcast branch will inherit the data for the instances that had overcast as the outlook. However, there will be no need to build further on that branch. There is a clear decision, yes, for all instances when outlook value is overcast. The decision tree will look like this after the first level of splitting.



Temp	Humidity	Windy	Play
Hot	High	False	No
Hot	High	True	No
Mild	High	False	No
Cool	Normal	False	Yes
Mild	Normal	True	Yes

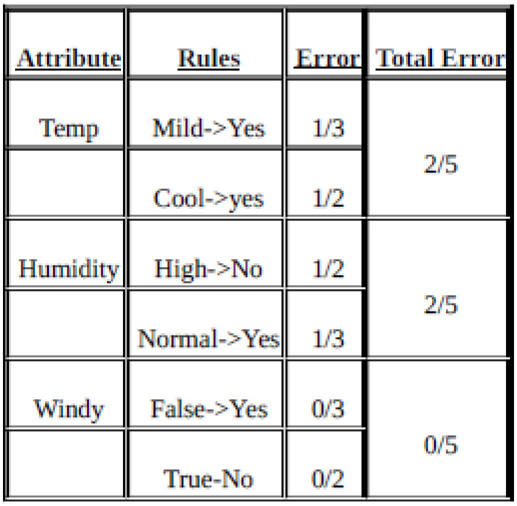
Temp	Humidity	Windy	Play
Mild	High	False	Yes
Cool	Normal	False	Yes
Cool	Normal	True	No
Mild	Normal	False	Yes
Mild	High	True	No

Determining the next nodes of the tree: A similar recursive logic of tree building should be applied to each branch. For the sunny branch on the left, error values will be calculated for the three other variables – temp, humidity and windy. Final comparison looks like this:



Attribute	Rules	Error	Total Error
Temp	Hot->No	0/2	1/5
	Mild ->No	1/2	
	Cool -> yes	0/1	
Humidity	High->No	0/3	0/5
	Normal->Yes	0/2	
Windy	False->No	1/3	2/5
	True->Yes	1/2	

The variable of humidity shows the least amount of error, i.e. zero error. The other two variables have non-zero errors. Thus the Outlook:sunny branch on the left will use humidity as the next splitting variable. Similar analysis should be done for the 'rainy' value of the tree. The analysis would look like this.



Attribute	Rules	Error	Total Error
Temp	Mild->Yes	1/3	2/5
	Cool->yes	1/2	
Humidity	High->No	1/2	2/5
	Normal->Yes	1/3	
Windy	False->Yes	0/3	0/5
	True-No	0/2	

For the Rainy branch, it can similarly be seen that the variable Windy gives all the correct answers, while none of the other two variables makes all the correct decisions. This is how the final decision tree looks like. Here it is produced using Weka open-source data mining platform (Figure 6.1). This is the model that abstracts the knowledge of the past data of decision.

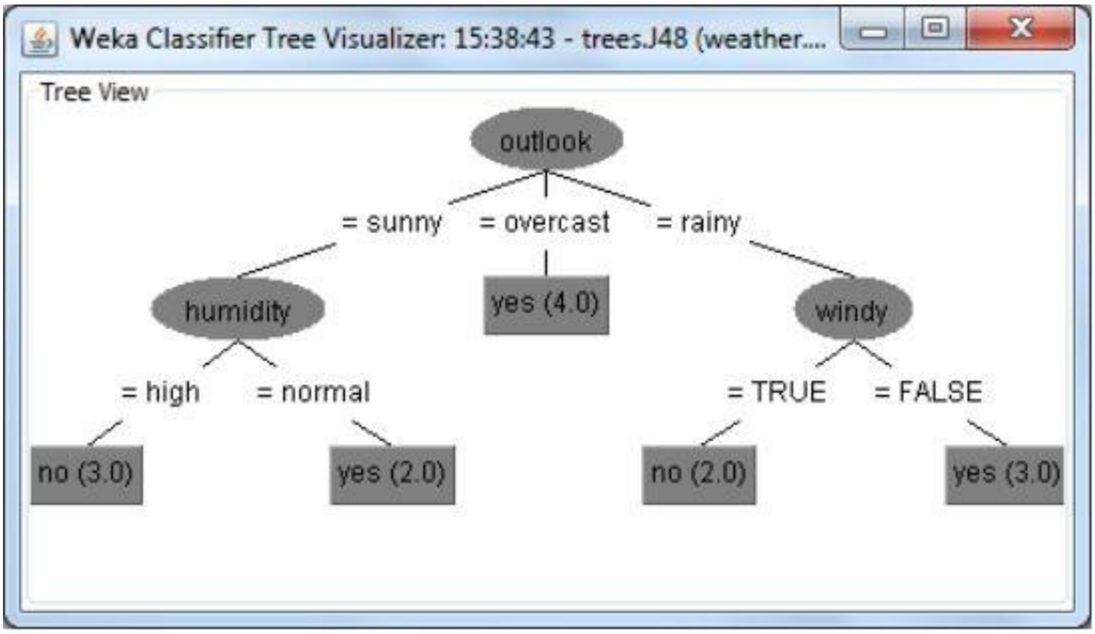
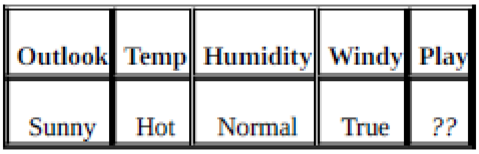


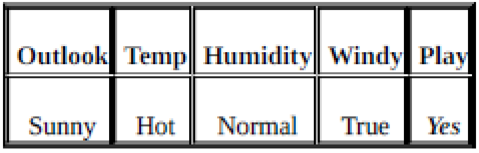
Figure 6.1: Decision Tree for the weather problem

This decision tree can be used to solve the current problem. Here is the problem again.



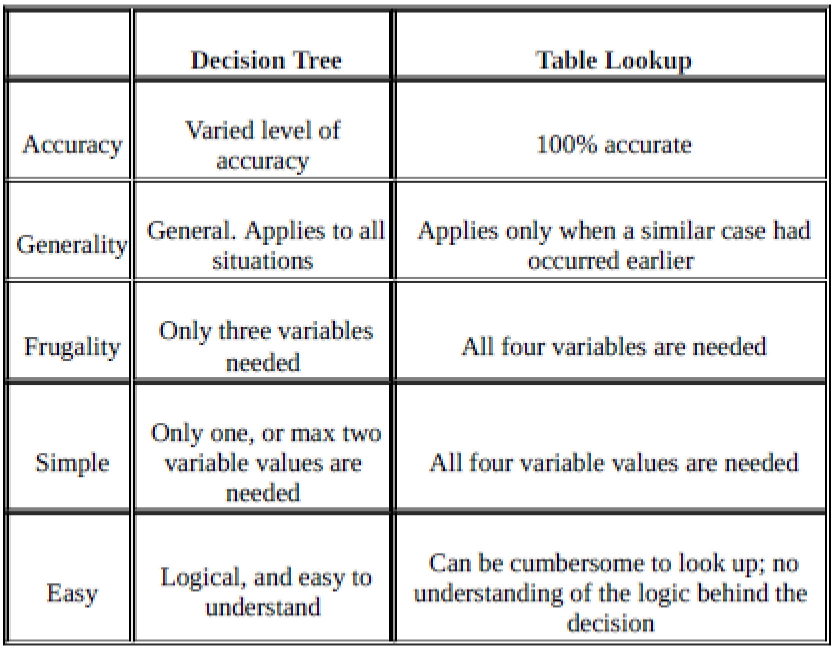
Outlook	Temp	Humidity	Windy	Play
Sunny	Hot	Normal	True	??

According to the tree, the first question to ask is about outlook. In this problem the outlook is sunny. So, the decision problem moves to the Sunny branch of the tree. The node in that sub-tree is humidity. In the problem, Humidity is Normal. That branch leads to an answer Yes. Thus, the answer to the play problem is Yes.



Outlook	Temp	Humidity	Windy	Play
Sunny	Hot	Normal	True	*Yes*

Lessons from constructing trees Here are some benefits of using this decision tree compared with looking up the answers from the data table (Figure 6.1)



	Decision Tree	Table Lookup
Accuracy	Varied level of accuracy	100% accurate
Generality	General. Applies to all situations	Applies only when a similar case had occurred earlier
Frugality	Only three variables needed	All four variables are needed
Simple	Only one, or max two variable values are needed	All four variable values are needed
Easy	Logical, and easy to understand	Can be cumbersome to look up; no understanding of the logic behind the decision

Figure 6.1: Comparing Decision Tree with Table Look-up

Here are a few observations about how the tree was constructed:

1. The final decision tree has zero errors in mapping to the prior data. In other words, the tree has a predictive accuracy of 100%. The tree completely fits the data. In real life situations, such perfect predictive accuracy is not possible when making decision trees. When there are larger, complicated data sets, with many more variables, a perfect fit is unachievable. This is especially true in business and social contexts, where things are not always fully clear and consistent.

2. The decision tree algorithm selected the minimum number of variables that are needed to solve the problem. Thus, one can start with all available data variables, and let the decision-tree algorithm select the ones that are useful, and discard the rest.
3. This tree is almost symmetric with all branches being of almost similar lengths. However, in real life situations, some of the branches may be much longer than the others, and the tree may need to be pruned to make it more balanced and usable.
4. It may be possible to increase predictive accuracy by making more sub-trees and making the tree longer. However, the marginal accuracy gained from each subsequent level in the tree will be less, and may not be worth the loss in ease and interpretability of the tree. If the branches are long and complicated, it will be difficult to understand and use. The longer branches may need to be trimmed to keep the tree easy to use.
5. A perfectly fitting tree has the danger of over-fitting the data, thus capturing all the random variations in the data. It may fit the training data well, but may not do well in predicting the future real instances.
6. There was a single best tree for this data. There could however be two or more equally efficient decision trees of similar length with similar predictive accuracy for the same data set. Decision trees are based strictly on patterns within the data, and do not rely on any underlying theory of the problem domain. When multiple candidate trees are available, one could choose whichever is easier to understand, communicate or implement.

Decision Tree Algorithms

As we saw, decision trees employ the divide and conquer method. The data is branched at each node according to certain criteria until all the data is assigned to leaf nodes. It recursively divides a training set until each division consists of examples from one class.
The following is a pseudo code for making decision trees:

1. Create a root node and assign all of the training data to it.
2. Select the best splitting attribute according to certain criteria.
3. Add a branch to the root node for each value of the split.
4. Split the data into mutually exclusive subsets along the lines of the specific split.

5. Repeat steps 2 and 3 for each and every leaf node until a stopping
criteria is reached.

There are many algorithms for making decision trees. Decision tree algorithms differ on three key elements:

Splitting criteria

1. Which variable to use for the first split? How should one determine the most important variable for the first branch, and subsequently, for each sub-tree? There are many measures like least errors, information gain, gini's coefficient, etc.
2. What values to use for the split? If the variables have continuous values such as for age or blood pressure, what value-ranges should be used to make bins?
3. How many branches should be allowed for each node? There could be binary trees, with just two branches at each node. Or there could be more branches allowed.

Stopping criteria:

1. When to stop building the tree? There are two major ways to make that determination.
2. The tree building could be stopped when a certain depth of the branches has been reached and the tree becomes unreadable after that. The tree could also be stopped when the error level at any node is within predefined tolerable levels.

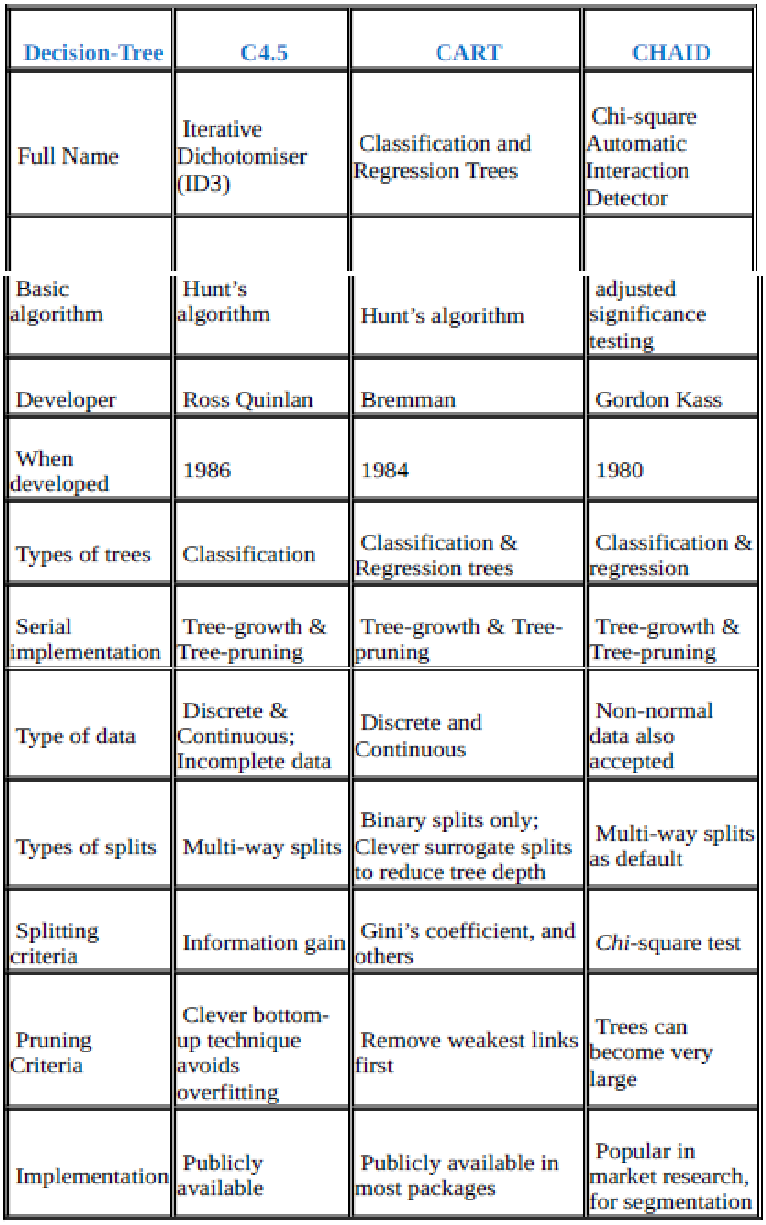
3. Pruning : The tree could be trimmed to make it more balanced and more easily usable. The pruning is often done after the tree is constructed, to balance out the tree and improve usability. The symptoms of an over- fitted tree are a tree too deep, with too many branches, some of which
may reflect anomalies due to noise or outliers. Thus, the tree should be pruned. There are two approaches to avoid over-fitting.

- Pre-pruning means to halt the tree construction early, when certain criteria are met. The downside is that it is difficult to decide what criteria to use for halting the construction, because we do not know what may happen subsequently, if we keep growing the tree.

-
Post-pruning: Remove branches or sub-trees from a "fully grown" tree. This method is commonly used. C4.5 algorithm uses a statistical method to estimate the errors at each node for pruning. A validation set may be used for pruning as well.

The most popular decision tree algorithms are C5, CART and CHAID (Table 6.2)

Figure 6.2: Comparing popular Decision Tree algorithms



Decision-Tree	C4.5	CART	CHAID
Full Name	Iterative Dichotomiser (ID3)	Classification and Regression Trees	Chi-square Automatic Interaction Detector
Basic algorithm	Hunt's algorithm	Hunt's algorithm	adjusted significance testing
Developer	Ross Quinlan	Bremman	Gordon Kass
When developed	1986	1984	1980
Types of trees	Classification	Classification & Regression trees	Classification & regression
Serial implementation	Tree-growth & Tree-pruning	Tree-growth & Tree-pruning	Tree-growth & Tree-pruning
Type of data	Discrete & Continuous; Incomplete data	Discrete and Continuous	Non-normal data also accepted
Types of splits	Multi-way splits	Binary splits only; Clever surrogate splits to reduce tree depth	Multi-way splits as default
Splitting criteria	Information gain	Gini's coefficient, and others	*Chi*-square test
Pruning Criteria	Clever bottom-up technique avoids overfitting	Remove weakest links first	Trees can become very large
Implementation	Publicly available	Publicly available in most packages	Popular in market research, for segmentation

Regression

Regression is a well-known statistical technique to model the predictive relationship between several independent variables (DVs) and one dependent variable. The objective is to find the best-fitting curve for a dependent variable in a multidimensional space, with each independent variable being a dimension. The curve could be a straight line, or it could be a nonlinear curve. The quality of fit of the curve to the data can be measured by a coefficient of correlation (r), which is the square root of the amount of variance explained by the curve.

The key steps for regression are simple:

List all the variables available for making the model. Establish a Dependent Variable (DV) of interest. Examine visual (if possible) relationships between variables of interest. Find a way to predict DV using the other variables.

Correlations and Relationships

Statistical relationships are about which elements of data hang together, and which ones hang separately. It is about categorizing variables that have a relationship with one another, and categorizing variables that are distinct and unrelated to other variables. It is about describing significant positive relationships and significant negative differences. The first and foremost measure of the strength of a relationship is co-relation (or correlation). The strength of a correlation is a quantitative measure that is measured in a normalized range between 0 (zero) and 1. A correlation of 1 indicates a perfect relationship, where the two variables are in perfect sync. A correlation of 0 indicates that there is no relationship between the variables. The relationship can be positive, or it can be an inverse relationship, that is, the variables may move together in the same direction or in the opposite direction. Therefore, a good measure of correlation is the correlation coefficient, which is the square root of correlation. This coefficient, called r, can thus range from −1 to +1. An r value of 0 signifies no relationship. An r value of 1 shows perfect relationship in the same direction, and an r value of −1 shows a perfect relationship but moving in opposite directions. Given two numeric variables x and y, the coefficient of correlation r is mathematically computed by the following equation.¯x (called x-bar) is the mean of x, and ȳ (y-bar) is the mean of y.

$$r = \frac{[(x-\bar{x})][y-\bar{y}]}{\sqrt{[(x-\bar{x})^2][(y-\bar{y})^2]}}$$

Visual look at relationships

A scatter plot (or scatter diagram) is a simple exercise for plotting all data points between two variables on a two-dimensional graph. It provides a visual layout of where all the data points are placed in that two-dimensional space. The scatter plot can be useful for graphically intuiting the relationship between two variables. Here is a picture (Figure 7.1) that shows many possible patterns in scatter diagrams.

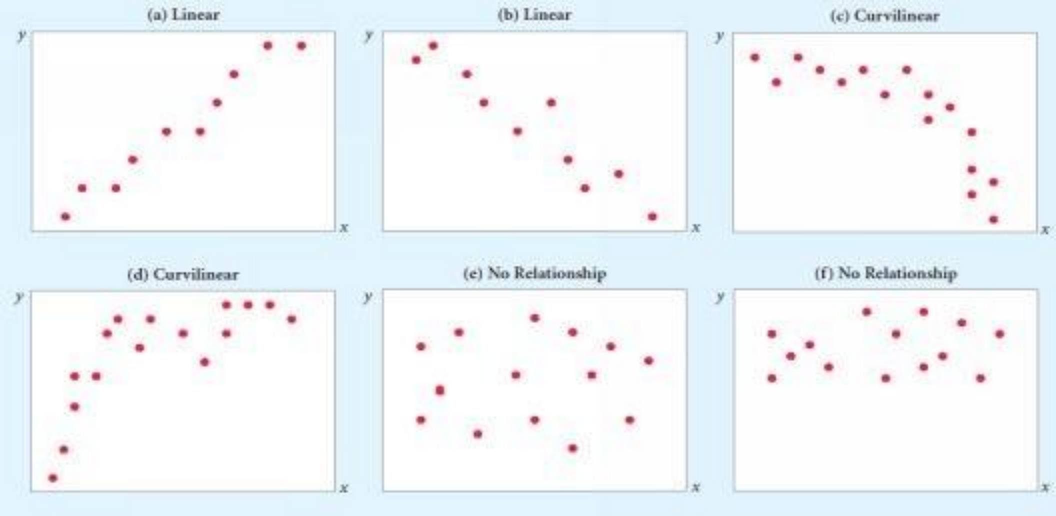


Figure 7.1: Scatter plots showing types of relationships among two variables(Source: Groebner et al. 2013)
(Source: Groebner et al. 2013)

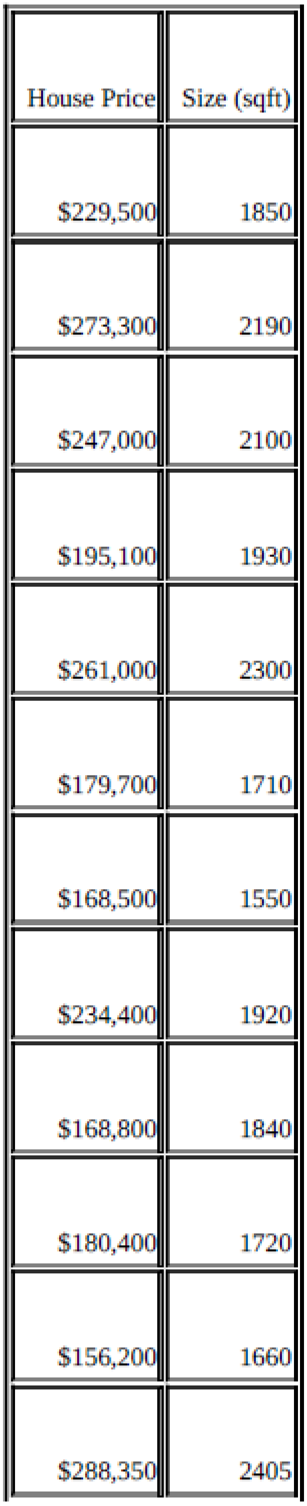
Chart (a) shows a very strong linear relationship between the variables x and y. That means the value of y increases proportionally with x. Chart (b) also shows a strong linear relationship between the variables x and y. Here it is an inverse relationship. That means the value of y decreases proportionally with x.

Chart (c) shows a curvilinear relationship. It is an inverse relationship, which means that the value of y decreases proportionally with x. However, it seems a relatively well-defined relationship, like an arc of a circle, which can be represented by a simple quadratic equation (quadratic means the power of two, that is, using terms like x and y). Chart (d) shows a positive curvilinear relationship. However, it does not seem to resemble a regular shape, and thus would not be a strong relationship. Charts (e) and (f) show no relationship. That means variables x and y are independent of each other.

Charts (a) and (b) are good candidates that model a simple linear regression model (the terms regression model and regression equation can be used interchangeably). Chart (c) too could be modeled with a little more complex, quadratic regression equation. Chart (d) might require an even higher order polynomial regression equation to represent the data. Charts (e) and (f) have no relationship, thus, they cannot be modeled together, by regression or using any other modeling tool.

Regression Exercise

The regression model is described as a linear equation that follows. y is the dependent variable, that is, the variable being predicted. x is the independent variable, or the predictor variable. There could be many predictor variables (such as x , x , . . .) in a regression equation. However, there can be only one dependent variable (y) in the regression equation.
1
2
y = β + β x + ε
0
1
A simple example of a regression equation would be to predict a house price from the size of the house. Here is a sample house prices data:



House Price	Size (sqft)
$229,500	1850
$273,300	2190
$247,000	2100
$195,100	1930
$261,000	2300
$179,700	1710
$168,500	1550
$234,400	1920
$168,800	1840
$180,400	1720
$156,200	1660
$288,350	2405

$186,750	1525
$202,100	2030
$256,800	2240

The two dimensions of (one predictor, one outcome variable) data can be plotted on a scatter diagram. A scatter plot with a best-fitting line looks like the graph that follows (Figure 7.2).

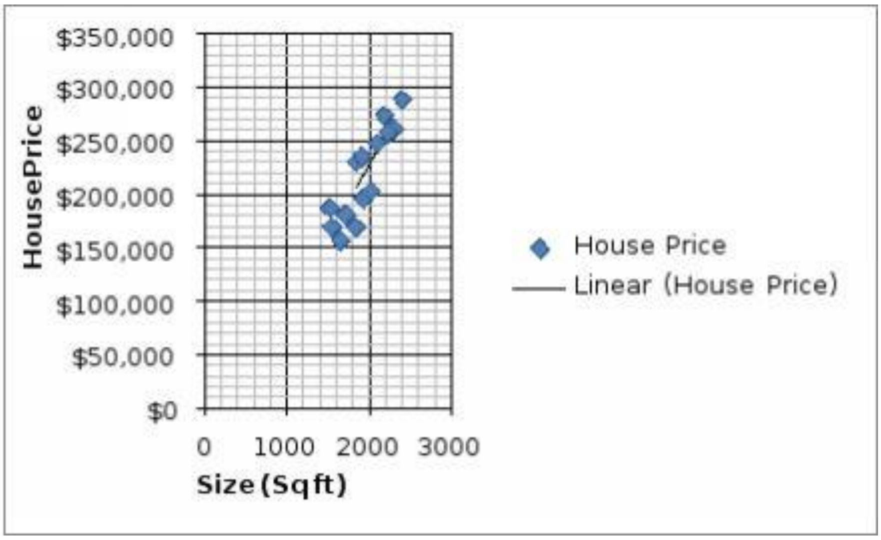
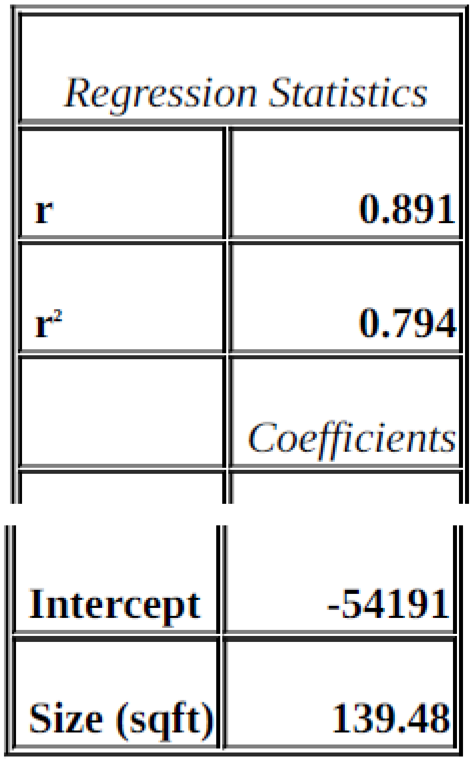


Figure 7.2: Scatter plot and regression equation between House price and house size.

Visually, one can see a positive correlation between House Price and Size (sqft). However, the relationship is not perfect. Running a regression model between the two variables produces the following output (truncated).

Visually, one can see a positive correlation between House Price and Size (sqft). However, the relationship is not perfect. Running a regression model between the two variables produces the following output (truncated).



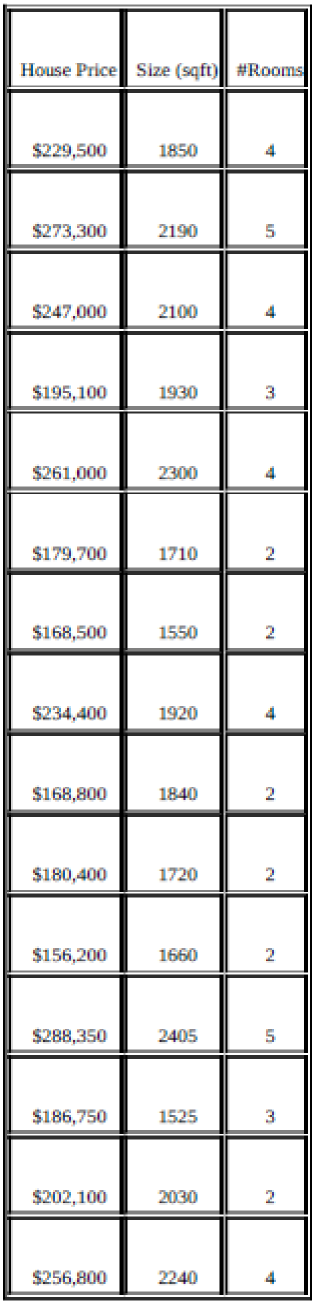
Regression Statistics	
r	**0.891**
r^2	**0.794**
	Coefficients
Intercept	**-54191**
Size (sqft)	**139.48**

It shows the coefficient of correlation is 0.891. r , the measure of total variance explained by the equation, is 0.794, or 79%. That means the two variables are moderately and positively correlated. Regression coefficients help create the following equation for predicting house prices.

House Price ($) = 139.48 * Size(sqft) – 54191

This equation explains only 79% of the variance in house prices. Suppose other predictor variables are made available, such as the number of rooms in the house. It might help improve the regression model.

The house data now looks like this:



House Price	Size (sqft)	#Rooms
$229,500	1850	4
$273,300	2190	5
$247,000	2100	4
$195,100	1930	3
$261,000	2300	4
$179,700	1710	2
$168,500	1550	2
$234,400	1920	4
$168,800	1840	2
$180,400	1720	2
$156,200	1660	2
$288,350	2405	5
$186,750	1525	3
$202,100	2030	2
$256,800	2240	4

Logistic Regression

Regression models traditionally work with continuous numeric value data for dependent and independent variables. Logistic regression models can, however, work with dependent variables with binary values, such as whether a loan is approved (yes or no). Logistic regression measures the relationship between a categorical dependent variable and one or more independent variables. For example, Logistic regression might be used to predict whether a patient has a given disease (e.g. diabetes), based on observed characteristics of the patient (age, gender, body mass index, results of blood tests, etc.). Logistical regression models use probability scores as the predicted values of the dependent variable. Logistic regression takes the natural logarithm of the odds of the dependent variable being a case (referred to as the logit) to create a continuous criterion as a transformed version of the dependent variable. Thus the logit transformation is used in logistic regression as the dependent variable. The net effect is that although the dependent variable in logistic regression is binomial (or categorical, i.e. has only two possible values), the logit is the continuous function upon

which linear regression is conducted. Here is the general logistic function, with independent variable on the horizontal axis and the logit dependent variable on the vertical axis (Figure
7.3).

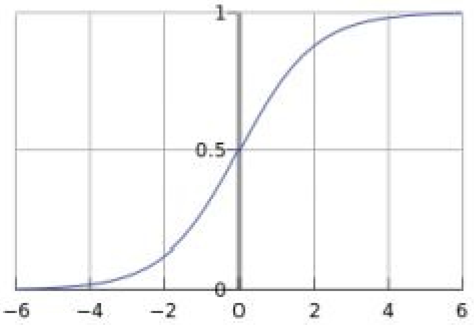


Figure 7.3: General Logit function

All popular data mining platforms provide support for regular multiple regression models, as well as options for Logistic Regression.

Advantages and Disadvantages of Regression Models

Regression Models are very popular because they offer many advantages.

1. Regression models are easy to understand as they are built upon basic statistical principles such as correlation and least square error.
2. Regression models provide simple algebraic equations that are easy to understand and use.
3. The strength (or the goodness of fit) of the regression model is measured in terms of the correlation coefficients, and other related statistical parameters that are well understood.
4. Regression models can match and beat the predictive power of other modeling techniques.
5. Regression models can include all the variables that one wants to include in the model.
6. Regression modeling tools are pervasive. They are found in statistical packages as well as data mining packages. MS Excel spreadsheets can also provide simple regression modeling capabilities.

Regression models can however prove inadequate under many circumstances.

1. Regression models can not cover for poor data quality issues. If the data is not prepared well to remove missing values, or is not well-behaved in terms of a normal distribution, the validity of the model suffers.
2. Regression models suffer from collinearity problems (meaning strong linear correlations among some independent variables). If the independent variables have strong correlations among themselves, then they will eat into each other's predictive power and the regression coefficients will lose their ruggedness. Regression models will not automatically choose between highly collinear variables, although some packages attempt to do that.

3. Regression models can be unwieldy and unreliable if a large number of variables are included in the model. All variables entered into the model will be reflected in the regression equation, irrespective of their contribution to the predictive power of the model. There is no concept of
automatic pruning of the regression model.

4. Regression models do not automatically take care of non-linearity. The user needs to imagine the kind of additional terms that might be needed to be added to the regression model to improve its fit.
5. Regression models work only with numeric data and not with categorical variables. There are ways to deal with categorical variables though by creating multiple new variables with a yes/no value.

Artificial Neural Networks

Artificial Neural Networks (ANN) are inspired by the information processing model of the mind/brain. The human brain consists of billions of neurons that link with one another in an intricate pattern. Every neuron receives information from many other neurons, processes it, gets excited or not, and passes its state information to other neurons. Just like the brain is a multipurpose system, so also the ANNs are very versatile systems. They can be used for many kinds of pattern recognition and prediction. They are also used for classification, regression, clustering, association, and optimization activities. They are used in finance, marketing, manufacturing, operations, information systems applications, and so on. ANNs are composed of a large number of highly interconnected processing elements (neurons) working in a multi-layered structures that receive inputs, process the inputs, and produce an output. An ANN is designed for a specific application, such as pattern recognition or data classification, and trained through a learning process. Just like in biological systems, ANNs make adjustments to the synaptic connections with each learning instance. ANNs are like a black box trained into solving a particular type of problem, and they can develop high predictive powers. Their intermediate synaptic parameter values evolve as the system obtains feedback on its predictions, and thus an ANN learns from more training data (Figure 8.1).

Inputs
ArtificialNeuralNetwork
Outputs

Figure 8.1: General ANN model

Business Applications of ANN

Neural networks are used most often when the objective function is complex, and where there exists plenty of data, and the model is expected to improve over a period of time. A few sample applications:
1. They are used in stock price prediction where the rules of the game are extremely complicated, and a lot of data needs to be processed very quickly.
2. They are used for character recognition, as in recognizing hand-written text, or damaged or mangled text. They are used in recognizing finger prints. These are complicated patterns and are unique for each person. Layers of neurons can progressively clarify the pattern leading to a remarkably accurate result.
3. They are also used in traditional classification problems, like approving a financial loan application.

Design Principles of an Artificial Neural Network

1.A neuron is the basic processing unit of the network. The neuron (or processing element) receives inputs from its preceding neurons (or PEs), does some nonlinear weighted computation on the basis of those inputs, transforms the result into its output value, and then passes on the output to the next neuron in the network (Figure 8.2). X's are the inputs, w's are the weights for each input, and y is the output.

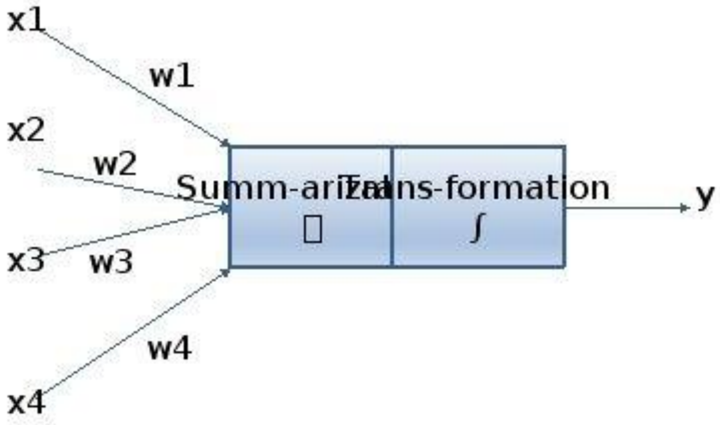


Figure 8.2: Model for a single artificial neuron

2. A Neural network is a multi-layered model. There is at least one input neuron, one output neuron, and at least one processing neuron. An ANN with just this basic structure would be a simple, single-stage computational unit. A simple task may be processed by just that one neuron and the result may be communicated soon. ANNs however, may have multiple layers of processing elements in sequence. There could be many neurons involved in a sequence depending upon the complexity of the predictive action. The layers of PEs could work in sequence, or they could work in parallel (Figure 8.3).

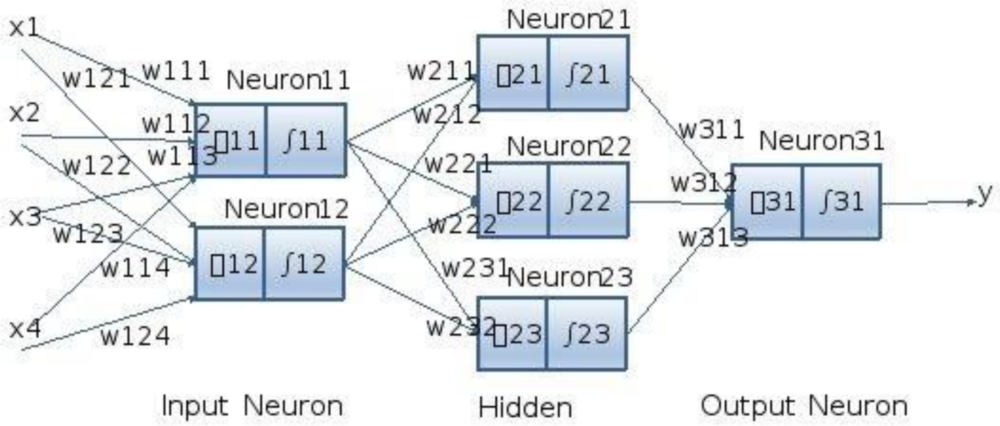


Figure 8.3: Model for a multi-layer ANN

3. The processing logic of each neuron may assign different weights to the various incoming input streams. The processing logic may also use non- linear transformation, such as a sigmoid function, from the processed values to the output value. This processing logic and the intermediate weight and processing functions are just what works for the system as a whole, in its objective of solving a problem collectively. Thus, neural networks are considered to be an opaque and a black-box system.

4. The neural network can be trained by making similar decisions over and over again with many training cases. It will continue to learn by adjusting its internal computation and communication based on feedback about its previous decisions. Thus, the neural networks become better at making a decision as they handle more and more decisions.

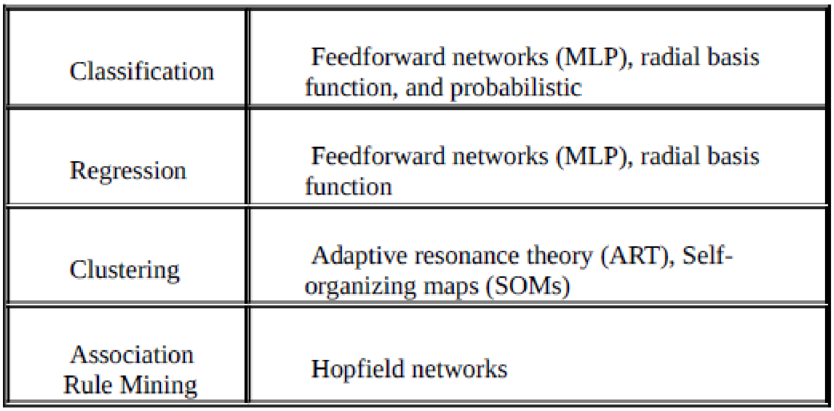
Representation of a Neural Network

A neural network is a series of neurons that receive inputs from other neurons. They do a weighted summation function of all the inputs, using different weights (or importance) for each input. The weighted sum is then transformed into an output value using a transfer function. Learning in ANN occurs when the various processing elements in the neural network adjust the underlying relationship (weights, transfer function, etc) between input and outputs, in response to the feedback on their predictions. If the prediction made was correct, then the weights would remain the same, but if the prediction was incorrect, then the parameter values would change. The Transformation (Transfer) Function is any function suitable for the task at hand. The transfer function for ANNs is usually a non-linear sigmoid function. Thus, if the normalized computed value is less than some value (say 0.5) then the output value will be zero. If the computed value is at the cut-off threshold, then the output value will be a 1. It could be a nonlinear hyperbolic function in which the output is either a -1 or a 1. Many other functions could be designed for any or all of the processing elements.
Thus, in a neural network, every processing element can potentially have a different number of input values, a different set of weights for those inputs, and a different transformation function. Those values support and compensate for one another until the neural network as a whole learns to provide the correct output, as desired by the user.

Architecting a Neural Network

There are many ways to architect the functioning of an ANN using fairly simple and open rules with a tremendous amount of flexibility at each stage. The most popular architecture is a Feed-forward, multi-layered perceptron with back-propagation learning algorithm. That means there are multiple layers of PEs in the system and the output of neurons are fed forward to the PEs in the next layers; and the feedback on the prediction is fed back into the neural network for learning to occur. This is essentially what was described in the earlier paragraphs. ANN architectures for different applications are shown in Table 8.1.

Table 8.1: ANN architectures for different applications

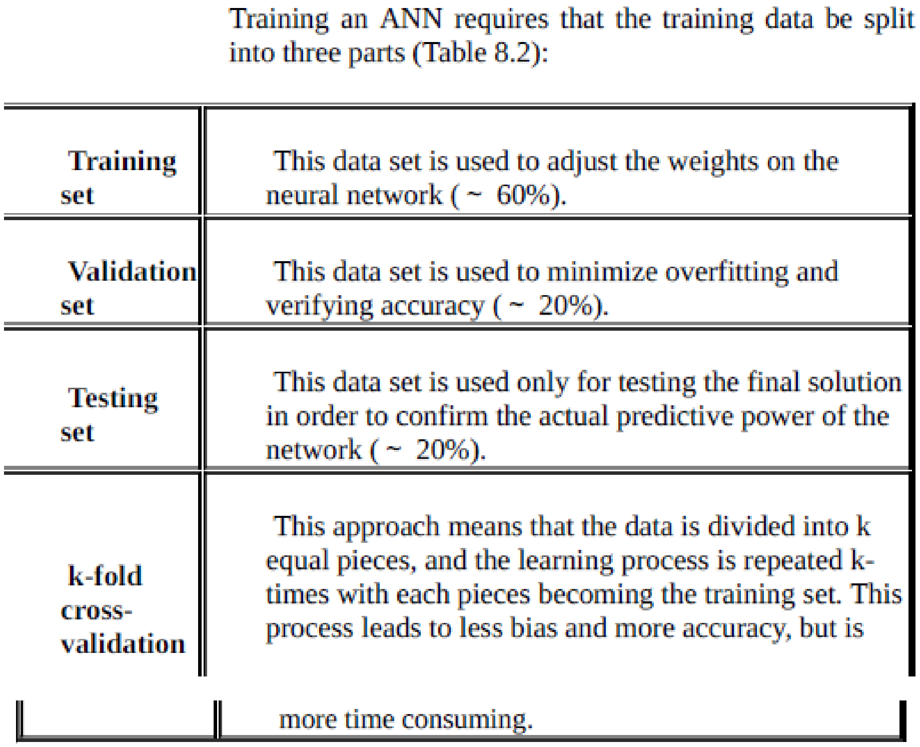


Classification	Feedforward networks (MLP), radial basis function, and probabilistic
Regression	Feedforward networks (MLP), radial basis function
Clustering	Adaptive resonance theory (ART), Self-organizing maps (SOMs)
Association Rule Mining	Hopfield networks

Developing an ANN

It takes resources, training data, skill and time to develop a neural network. Most data mining platforms offer at least the Multi-Layer-Perceptron (MLP) algorithm to implement a neural network. Other neural network architectures include Probabilistic networks and Self-organizing feature maps.

The steps required to build an ANN are as follows:
1. Gather data. Divide into training data and test data. The training data needs to be further divided into training data and validation data.
2. Select the network architecture, such as Feedforward network.
3. Select the algorithm, such as Multi-layer Perception.
4. Set network parameters.
5. Train the ANN with training data.
6. Validate the model with validation data.
7. Freeze the weights and other parameters.
8. Test the trained network with test data.
9. Deploy the ANN when it achieves good predictive accuracy.



Training an ANN requires that the training data be split into three parts (Table 8.2):

Training set	This data set is used to adjust the weights on the neural network (~ 60%).
Validation set	This data set is used to minimize overfitting and verifying accuracy (~ 20%).
Testing set	This data set is used only for testing the final solution in order to confirm the actual predictive power of the network (~ 20%).
k-fold cross-validation	This approach means that the data is divided into k equal pieces, and the learning process is repeated k-times with each pieces becoming the training set. This process leads to less bias and more accuracy, but is more time consuming.

Table 8.2: ANN Training datasets

Advantages and Disadvantages of using ANNs

There are many benefits of using ANN.

1. ANNs impose very little restrictions on their use. ANN can deal with (identify/model) highly nonlinear relationships on their own, without much work from the user or analyst. They help find practical data-driven solutions where algorithmic solutions are non-existent or too complicated.
2. There is no need to program neural networks, as they learn from examples. They get better with use, without much programming effort.
3. They can handle a variety of problem types, including classification, clustering, associations, etc.
4. ANN are tolerant of data quality issues and they do not restrict the data to follow strict normality and/or independence assumptions.
5. They can handle both numerical and categorical variables.
6. ANNs can be much faster than other techniques.

7. Most importantly, they usually provide better results (prediction and/or clustering) compared to statistical counterparts, once they have been trained enough.

The key disadvantages arise from the fact that they are not easy to interpret or explain or compute.
1. They are deemed to be black-box solutions, lacking explainability. Thus they are difficult to communicate about, except through the strength of their results.
2. Optimal design of ANN is still an art: it requires expertise and extensive experimentation.
3. It can be difficult to handle a large number of variables (especially the rich nominal attributes).
4. It takes large data sets to train an ANN.

Cluster Analysis

Applications of Cluster Analysis

Cluster analysis is used in almost every field where there is a large variety of transactions. It helps provide characterization, definition, and labels for populations. It can help identify natural groupings of customers, products, patients, and so on. It can also help identify outliers in a specific domain and thus decrease the size and complexity of problems. A prominent business application of cluster analysis is in market research. Customers are segmented into clusters based on their characteristics—wants and needs, geography, price sensitivity, and so on. Here are some examples of clustering:

1. Market Segmentation: Categorizing customers according to their similarities, for instance by their common wants and needs, and propensity to pay, can help with targeted marketing.
2. Product portfolio: People of similar sizes can be grouped together to make small, medium and large sizes for clothing items.
3. Text Mining: Clustering can help organize a given collection of text documents according to their content similarities into clusters of related topics.

Definition of a Cluster

An operational definition of a cluster is that, given a representation of n objects, find K groups based on a measure of similarity, such that objects within the same group are alike but the objects in different groups are not alike.
However, the notion of similarity can be interpreted in many ways. Clusters can differ in terms of their shape, size, and density. Clusters are patterns, and there can be many kinds of patterns. Some clusters are the traditional types, such as data points hanging together. However, there are other clusters, such as all points representing the circumference of a circle. There may be concentric circles with points of different circles representing different clusters. The presence of noise in the data makes the detection of the clusters even more difficult.

An ideal cluster can be defined as a set of points that is compact and isolated. In reality, a cluster is a subjective entity whose significance and interpretation requires domain knowledge. In the sample data below (Figure 9.1), how many clusters can one visualize?

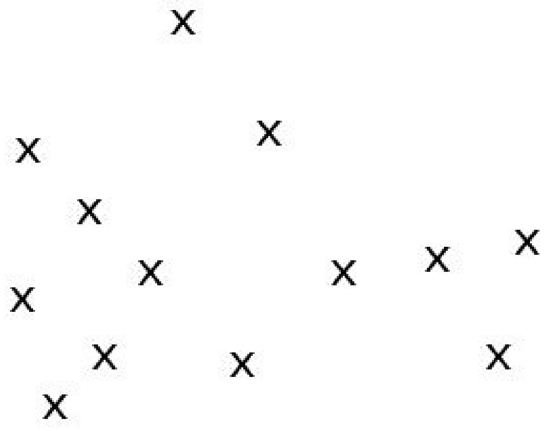


Figure 9.1: Visual cluster example

It seems like there are two clusters of approximately equal sizes. However, they can be seen as three clusters, depending on how we draw the dividing lines. There is not a truly optimal way to calculate it. Heuristics are often used to define the number of clusters.

Representing clusters

The clusters can be represented by a central or modal value. A cluster can be defined as the centroid of the collection of points belonging to it. A centroid is a measure of central tendency. It is the point from where the sum total of squared distance from all the points is the minimum. A real-life equivalent would be the city center as the point that is considered the most easy to use by all constituents of the city. Thus all cities are defined by their centers or downtown areas. A cluster can also be represented by the most frequently occurring value in the cluster, i.e. the cluster can defined by its modal value. Thus, a particular cluster representing a social point of view could be called the 'soccer moms', even though not all members of that cluster need currently be a mom with soccer-playing children.

Clustering techniques

Cluster analysis is a machine-learning technique. The quality of a clustering result depends on the algorithm, the distance function, and the application. First, consider the distance function. Most cluster analysis methods use a distance measure to calculate the closeness between pairs of items. There are two major measures of distances: Euclidian distance ("as the crow flies" or straight line) is the most intuitive measure. The other popular measure is the Manhattan (rectilinear) distance, where one can go only in orthogonal directions. The Euclidian distance is the hypotenuse of a right triangle, while the Manhattan distance is the sum of the two legs of the right triangle. In either case, the key objective of the clustering algorithm is the same:

-
Inter-clusters distanceÞ
maximized; and

-
Intra-clusters distanceÞ
minimized

There are many algorithms to produce clusters. There are top-down, hierarchical methods that start with creating a given number of best-fitting clusters. There are also bottom-up methods that begin with identifying naturally occurring clusters.

The most popular clustering algorithm is the K-means algorithm. It is a top-down, statistical technique, based on the method of minimizing the least squared distance from the center points of the clusters. Other techniques, such as neural networks, are also used for clustering. Comparing cluster algorithms is a difficult task as there is no single right number of clusters. However, the speed of the algorithm and its versatility in terms of different dataset are important criteria.

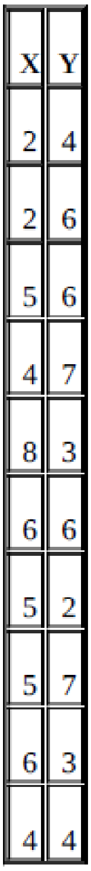
Here is the generic pseudocode for clustering

```
1. Pick an arbitrary number of groups/segments to be created
2. Start with some initial randomly-chosen center values for groups
3. Classify instances to closest groups
4. Compute new values for the group centers
5. Repeat step 3 & 4 till groups converge
6. If clusters are not satisfactory, go to step 1 and pick a different number of groups/segments
```

The clustering exercise can be continued with a different number of clusters and different location of those points. Clusters are considered good if the cluster definitions stabilize, and the stabilized definitions prove useful for the purpose at hand. Else, repeat the clustering exercise with a different number of clusters, and different starting points for group means.

Clustering Exercise

Here is a simple exercise to visually and intuitive identify clusters from data. X and Y are two dimensions of interest. The objective is to determine the number of clusters, and the center points of those clusters.



X	Y
2	4
2	6
5	6
4	7
8	3
6	6
5	2
5	7
6	3
4	4

A scatter plot of 10 items in 2 dimensions shows them distributed fairly randomly. As a bottom-up technique, the number of clusters and their centroids can be intuited (Figure 9.2).

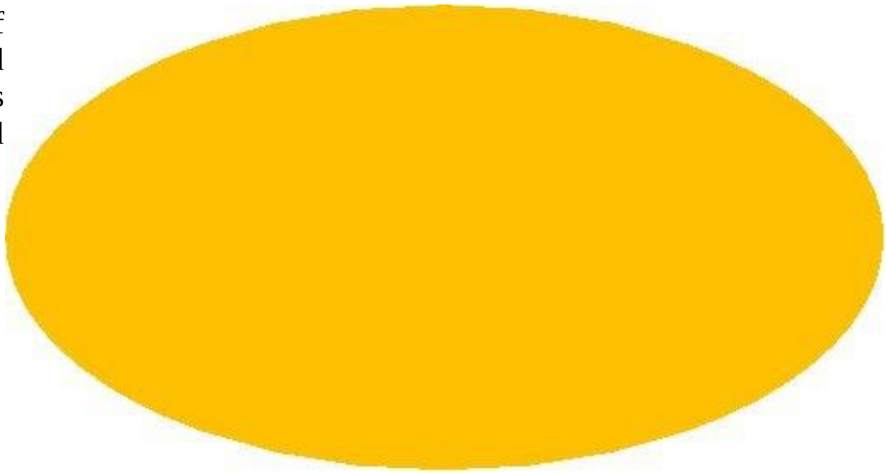


Figure 9.2: Initial data points and the centroid (shown as thick dot)

The points are distributed randomly enough that it could be considered one cluster. The solid circle would represent the central point (centroid) of these points.

However, there is a big distance between the points (2,6) and (8,3). So, this data could be broken into 2 clusters. The three points at the bottom right could form one cluster and the other seven could form the other cluster. The two clusters would look like this (Figure 9.3). The two circles will be the new centroids.

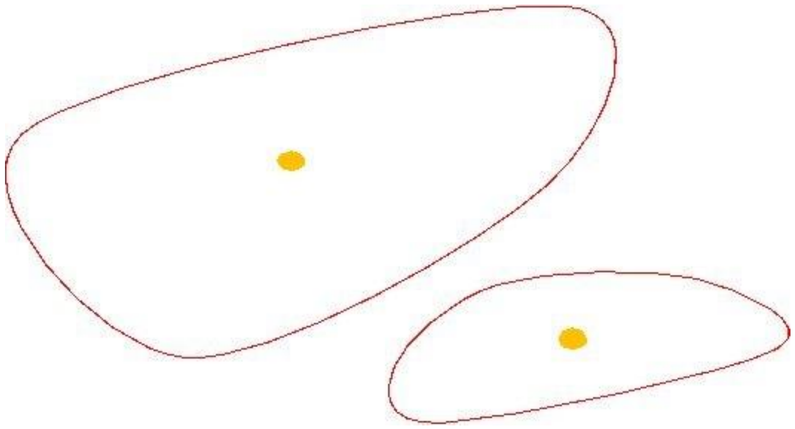


Figure 9.3: Dividing into two clusters (centroids shown as thick dots)

The bigger cluster seems too far apart. So, it seems like the 4 points on the top will form a separate cluster. The three clusters could look like this (Figure 9.4).

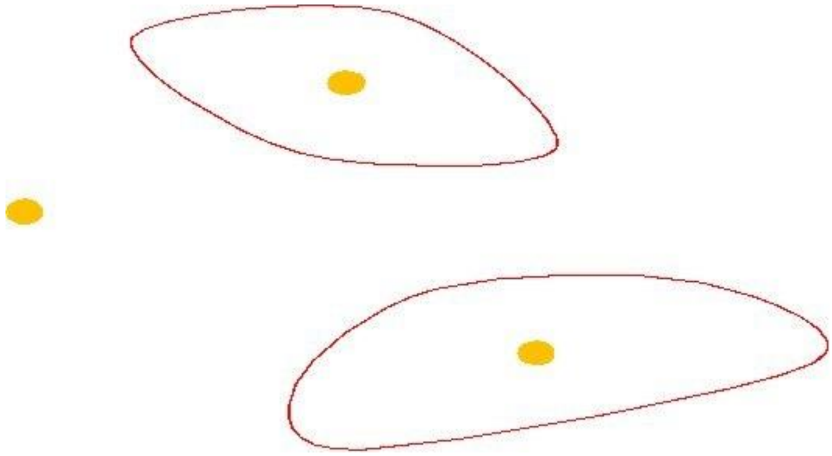


Figure 9.4: Dividing into three clusters (centroids shown as thick dots)

This solution has three clusters. The cluster on the right is far from the other two clusters. However, its centroid is not too close to all the data points. The cluster at the top looks very tight-fitting, with a nice centroid. The third cluster, at the left, is spread out and may not be of much usefulness. This was a bottom-up exercise in visually producing three best-fitting cluster definitions from the given data. The right number of clusters will depend on the data and the application for which the data would be used.

K-Means Algorithm for clustering

K-means is the most popular clustering algorithm. It iteratively computes the clusters and their centroids. It is a top down approach to clustering. Starting with a given number of K clusters, say 3 clusters. Thus three random centroids will be created as starting points of the centers of three clusters. The circles are initial cluster centroids (Figure 9.5).

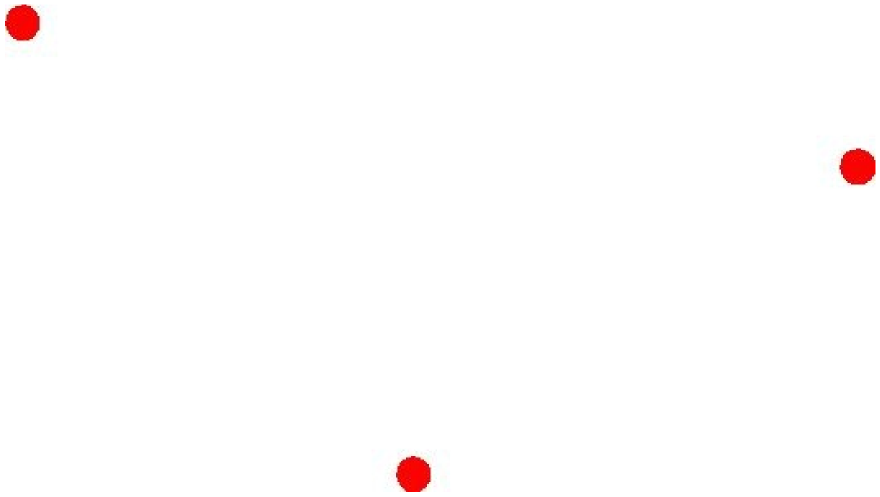


Figure 9.5: Randomly assigning three centroids for three data clusters

Step 1: For a data point, distance values will be from each of the three centroids. The data point will be assigned to the cluster with the shortest distance to the centroid. All data points will thus, be assigned to one data point or the other (Figure 9.6). The arrows from each data element shows the centroid that the point is assigned to.

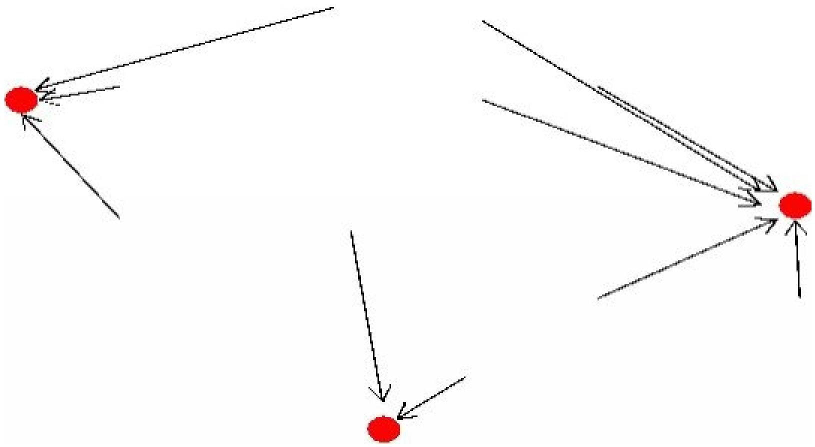


Figure 9.6: Assigning data points to closest centroid

Step 2: The centroid for each cluster will now be recalculated such that it is closest to all the data points allocated to that cluster. The dashed arrows show the centroids being moved from their old (shaded) values to the revised new values (Figure 9.7).

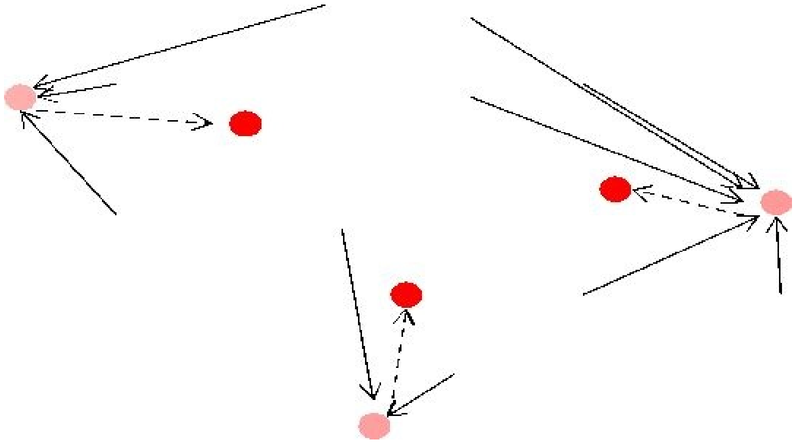


Figure 9.7: Recomputing centroids for each cluster

Step 3: Once again, data points are assigned to the three centroids closest to it (Figure 9.8).

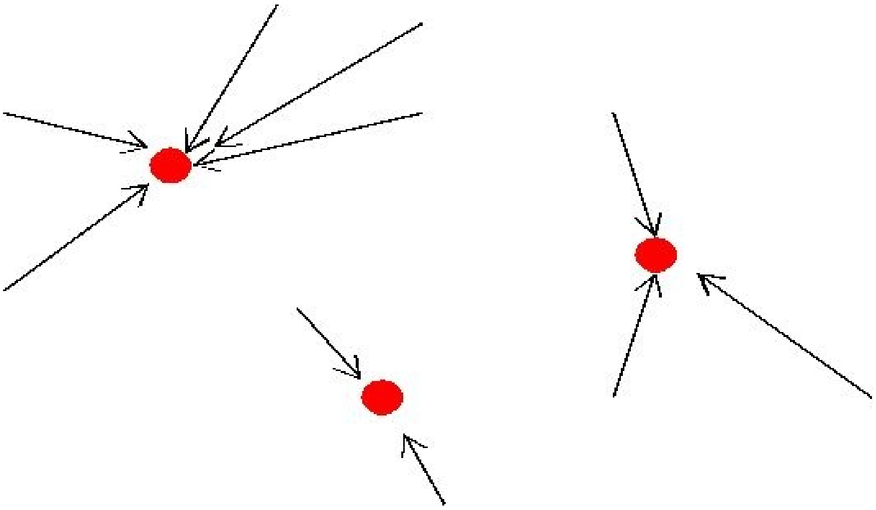


Figure 9.8: Assigning data points to recomputed centroids

The new centroids will be computed from the data points in the cluster until finally, the centroids stabilize in their locations. These are the three clusters computed by this algorithm.

Selecting the number of clusters

The correct choice of the value of k is often ambiguous. It depends on the shapeand scale of the distribution points in a data set and the desired clustering resolution of the user. Heuristics are needed to pick the right number. One can graph the percentage of variance explained by the clusters against the number of clusters (Fig 9.10). The first clusters will add more information (explain a lot of variance), but at some point the marginal gain in variance will fall, giving a sharp angle to the graph, looking like an elbow. Beyond that elbow point, adding more clusters will not add much incremental value. That would be the desired K.

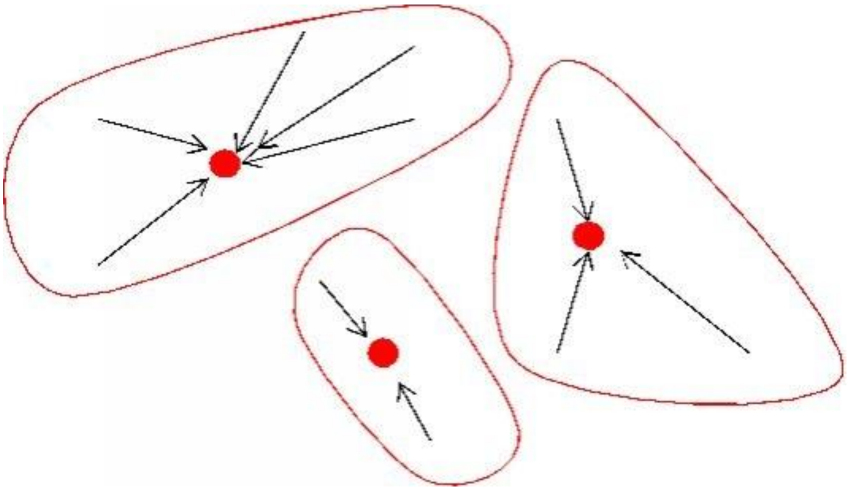


Figure 9.9: Recomputing centroids for each cluster till clusters stabilize

The three clusters shown are: a 3-datapoints cluster with centroid (6.5,4.5), a 2- datapoint cluster with centroid (4.5,3) and a 5-datapoint cluster with centroid (3.5,3) (Figure 9.9). These cluster definitions are different from the ones derived visually. This is a function of the random starting centroid values. The centroid points used earlier in the visual exercise were different from that chosen with the K- means clustering algorithm. The K-means clustering exercise should therefore, be run again with this data, but with new random centroid starting values. With many runs, the cluster definitions are likely to stabilize. If the cluster definitions do not stabilize, that may be a sign that the number of clusters chosen is too high or too low. The algorithm should also be run with different values of K.

Selecting the number of clusters

The correct choice of the value of k is often ambiguous. It depends on the shapeand scale of the distribution points in a data set and the desired clustering resolution of the user. Heuristics are needed to pick the right number. One can graph the percentage of variance explained by the clusters against the number of clusters (Fig 9.10). The first clusters will add more information (explain a lot of variance), but at some point the marginal gain in variance will fall, giving a sharp angle to the graph, looking like an elbow. Beyond that elbow point, adding more clusters will not add much incremental value. That would be the desired K.

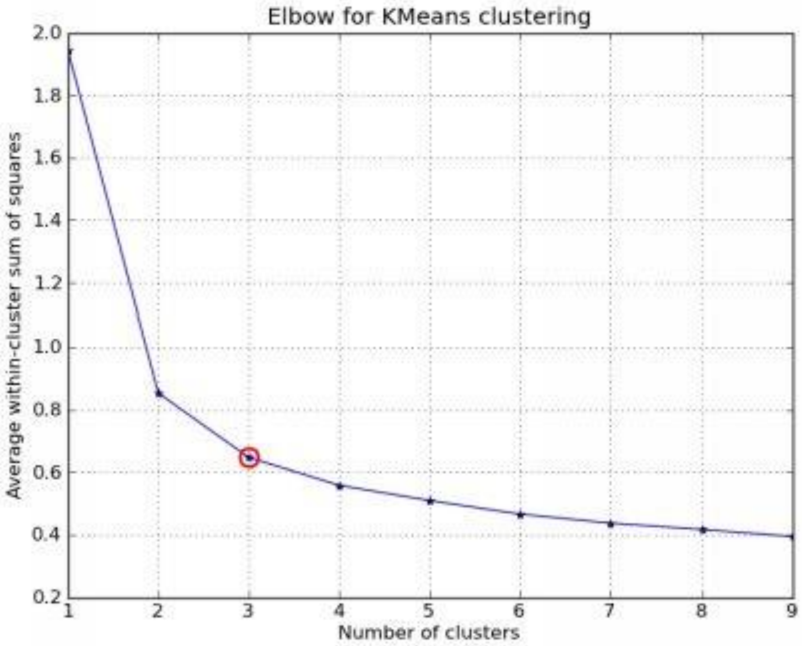


Figure 9.10: Elbow method for determining number of clusters in a data set

To engage with the data and to understand the clusters better, it is often better to start with a small number of clusters such as 2 or 3, depending upon the data set and the application domain. The number can be increased subsequently, as needed from an application point of view. This helps understand the data and the clusters progressively better.

Advantages and Disadvantages of K-Means algorithm

There are many advantages of K-Means Algorithm
1. K-Means algorithm is simple, easy to understand and easy to implement.
2. It is also efficient, in that the time taken to cluster k-means, rises linearly with the number of data points.
3. No other clustering algorithm performs better than K-Means, in general.
There are a few disadvantages too:
1. The user needs to specify an initial value of K.
2. The process of finding the clusters may not converge.
3. It is not suitable for discovering clusters shapes that are not hyper-ellipsoids (or hyper-spheres).
Neural networks can also be deployed for clustering, using the appropriate objective function. The neural network will produce the appropriate cluster centroids and cluster population for each cluster.

Association Rule Mining

Associate rule mining is a popular, unsupervised learning technique, used in business to help identify shopping patterns. It is also known as market basket analysis. It helps find interesting relationships (affinities) between variables (items or events). Thus, it can help cross-sell related items and increase the size of a sale.

All data used in this technique is categorical . There is no dependent variable. It uses machine learning algorithms. The fascinating "relationship between sales of diapers and beers' is how it is often explained in popular literature. This technique accepts as input the raw point-of-sale transaction data. The output produced is the description of the most frequent affinities among items. An example of an association rule would be, "A Customer who bought a flight tickets and a hotel reservation also bought a rental car plan 60 percent of the time."

Business Applications of Association Rules

In business environments a pattern or knowledge can be used for many purposes. In sales and marketing, it is used for cross-marketing and cross- selling, catalog design, e-commerce site design, online advertising optimization, product pricing, and sales/promotion configurations. This analysis can suggest not to put one item on sale at a time, and instead to create a bundle of products promoted as a package to sell other non-selling items. In retail environments, it can be used for store design. Strongly associated items can be kept close tougher for customer convenience. Or they could be placed far from each other so that the customer has to walk the aisles and by doing so is potentially exposed to other items. In medicine, this technique can be used for relationships between symptoms and illnesses; diagnosis and patient characteristics/treatments; genes and their functions; etc.

Representing Association Rules

A generic Association Rule is represented between a set X and Y: XÞ Y [S%,C%]

X, Y: products and/or services
X: Left-hand-side (LHS)
Y: Right-hand-side (RHS)
S: Support: how often X and Y go together in the dataset – i.e. P (X U Y)
C: Confidence: how often Y is found, given X – i.e. P (Y | X)
Example: {Hotel booking, Flight booking}Þ {Rental Car} [30%, 60%]
[Note: P (X) is the mathematical representation of the the probability or chance of X occurring in the data set.}
Computation example:
Suppose there are 1000 transactions in a data set. There are 300 occurrences of X, and 150 occurrences of (X,Y) in the data set.
Support S for XÞ Y will be P(X U Y) = 150/1000 = 15%.
Confidence for XÞ Y will be P (Y | X); or P (X U Y) / P (X) = 150/300 = 50%

Algorithms for Association Rule

Not all association rules are interesting and useful, only those that are strong rules and also those that occur frequently. In association rule mining, the goal is to find all rules that satisfy the user-specified minimum support and minimum confidence. The resulting sets of rules are all the same irrespective of the algorithm used, that is, given a transaction data set T, a minimum support and a minimum confidence, the set of association rules existing in T is uniquely determined.
Fortunately, there is a large number of algorithms that are available for generating association rules. The most popular algorithms are Apriori, Eclat, FP-Growth, along with various derivatives and hybrids of the three. All the algorithms help identify the frequent item sets, which are then converted to association rules.

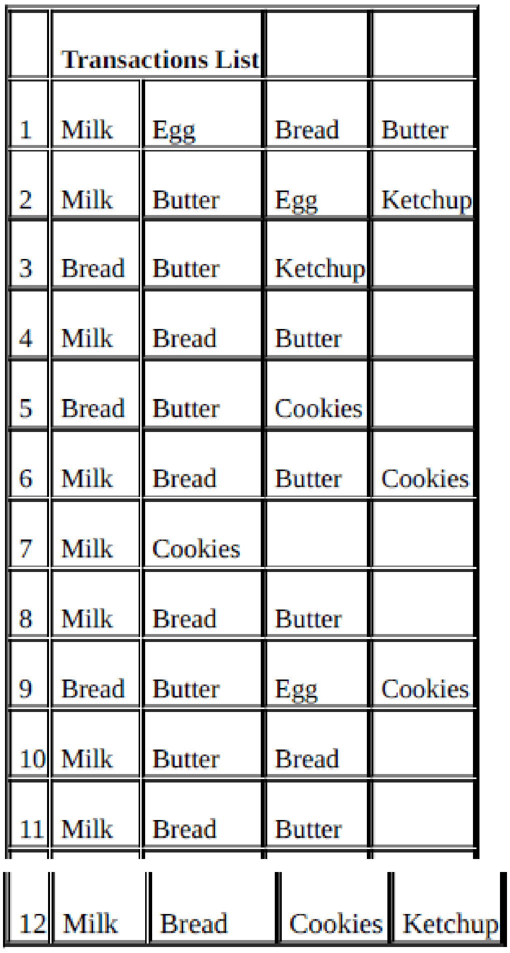
Apriori Algorithm

This is the most popular algorithm used for association rule mining. The objective is to find subsets that are common to at least a minimum number of the itemsets. A frequent itemset is an itemset whose support is greater than or equal to minimum support threshold. The Apriori property is a downward closure property, which means that any subsets of a frequent itemset are also frequent itemsets. Thus, if (A,B,C,D) is a frequent itemset, then any subset such as (A,B,C) or (B,D) are also frequent itemsets. It uses a bottom-up approach; and the size of frequent subsets is gradually increased, from one-item subsets to two-item subsets, then three-item subsets, and so on. Groups of candidates at each level are tested against the data for minimum support.

Association rules exercise

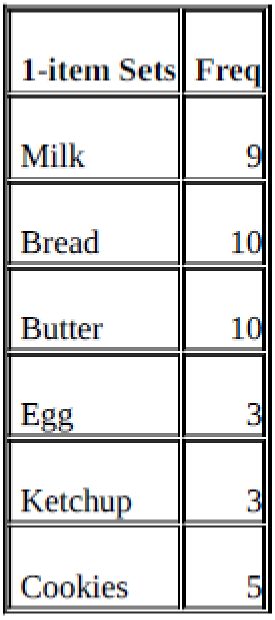
Here are a dozen sales transactions. There are six products being sold: Milk, Bread, Butter, Eggs, Cookies, and Ketchup. Transaction#1 sold Milk, Eggs, Bread and Butter. Transaction#2 sold Milk, Butter, Egg & Ketchup. And so on. The objective is to use this transaction data to find affinities between products, i.e. which products sell together often.

The support level will be set at 33 percent; the confidence level will be set at 50 percent. That means that we have decided to consider rules from only those itemsets that occur at least 33 percent of the time in the total set of transactions. Confidence level means that within those itemsets, the rules of the form X → Y should be such that there is at least 50 percent chance of Y occurring based on X occurring.



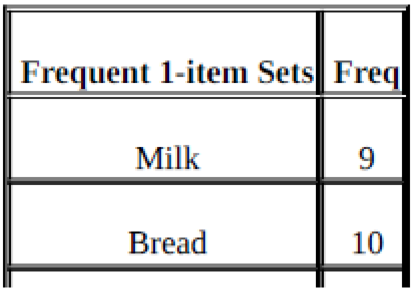
	Transactions List			
1	Milk	Egg	Bread	Butter
2	Milk	Butter	Egg	Ketchup
3	Bread	Butter	Ketchup	
4	Milk	Bread	Butter	
5	Bread	Butter	Cookies	
6	Milk	Bread	Butter	Cookies
7	Milk	Cookies		
8	Milk	Bread	Butter	
9	Bread	Butter	Egg	Cookies
10	Milk	Butter	Bread	
11	Milk	Bread	Butter	
12	Milk	Bread	Cookies	Ketchup

First step is to compute 1-item Itemsets. i.e. How often does any product individually sell.

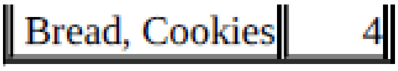


1-item Sets	Freq
Milk	9
Bread	10
Butter	10
Egg	3
Ketchup	3
Cookies	5

Thus, Milk sells in 9 out of 12 transactions. Bread sells in 10 out of 12 transactions. And so on.
At every point, there is an opportunity to select itemsets of interest, and thus further analysis. Other itemsets that occur very infrequently may be removed. If itemsets that occur 4 or more times out of 12 are selected, that corresponds to meeting a minimum support level of 33 percent (4 out of 12). Only 4 items make the cut. The frequent items that meet the support level of 33 percent are:

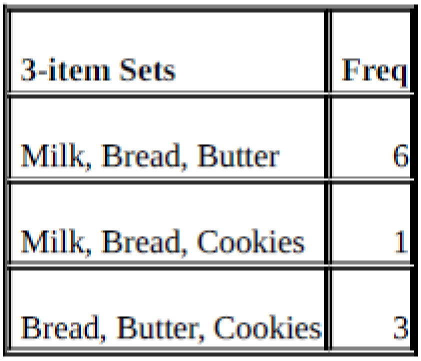


Frequent 1-item Sets	Freq
Milk	9
Bread	10



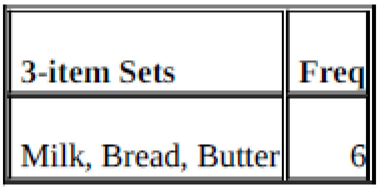
Bread, Cookies	4

The next step is to list the next higher level of itemsets: 3-item itemsets**.**



3-item Sets	Freq
Milk, Bread, Butter	6
Milk, Bread, Cookies	1
Bread, Butter, Cookies	3

Thus (Milk, Bread, Butter) sell 6 times out of 12. (Bread, Butter, Cookies) sell 3 times out of 12. One one 3-item itemset meets the minimum support requirements.

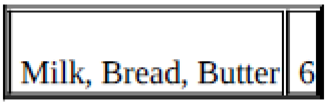


3-item Sets	Freq
Milk, Bread, Butter	6

There is no room to create a 4-item itemset for this support level.

Creating Association Rules

The most interesting and complex rules at higher size itemsets start top-down with the most frequent itemsets of higher size-numbers. Association rules are created that meet the support level (>33%) and confidence levels (> 50%). The highest level itemset that meets the support requirements is the three-item itemset. The following itemset has a support level of 50% (6 out of 12).



Milk, Bread, Butter	6

This itemset could lead to multiple candidate Association rules.

Start with the following rule: (Bread, Butter) Milk.

There are a total of total 12 transactions.
X (in this case Bread, Butter) occurs 9 times;
X,Y (in this case Bread, Butter, Milk) occurs 6 times.
The support level for this rule is 6/12 = 50%. The confidence level for this rule is 6/9 = 67%. This rule meets our thresholds for support (>33%) and confidence (>50%).

Thus, the first valid Association rule from this data is: (Bread, Butter)Milk {S=50%, C=67%}.

In exactly the same way, other rules can be considered for their validity.
Consider the rule: (Milk, Bread) Butter. Out of total 12 transactions, (Milk,

Bread) occur 7 times; and (Milk, Bread, Butter) occurs 6 times.

The support level for this rule is 6/12 = 50%. The confidence level for this rule is 6/7 = 84%. This rule meets our thresholds for support (>33%) and confidence (>50%).

Thus, the second valid Association rule from this data is (Milk, Bread)Butter {S=50%, C=67%}.

Consider the rule (Milk, Butter)Bread. Out of total 12 transactions (Milk, 201Butter) occurs 7 times while (Milk, Butter, Bread) occur 6 times.

The support level for this rule is 6/12 = 50%. The confidence level for this rule is 6/7 = 84%. This rule meets our thresholds for support (>33%) and confidence (>50%).

Thus, the next valid Association rule is: Milk,Butter Bread {S=50%, C=84%}.
Thus, there were only three possible rules at the 3-item itemset level, and all were found to be valid. One can get to the next lower level and generate association rules at the 2-item itemset level.

Consider the rule Milk Bread. Out of total 12 transactions Milk occurs 9 times while (Milk, Bread) occur 7 times.

The support level for this rule is 7/12 = 58%. The confidence level for this rule is 7/9 = 78%. This rule meets our thresholds for support (>33%) and confidence (>50%).

Thus, the next valid Association rule is:

Milk -> Bread {58%, 77%}.
Many such rules could be derived if needed.

Not all such association rules are interesting. The client may be interested in only the top few rules that they want to implement. The number of association rules depends upon business need. Implementing every rule in business will require some cost and effort, with some potential of gains. The strongest of rules, with the higher support and confidence rates, should be used first, and the others should be progressively implemented later.