# 6 Decision Trees

Learning Objectives

* Understand the concept of Decision Trees
* Know the importance of decision trees in data mining
* Learn how to construct a decision tree from a given dataset
* Know about algorithms for making decision trees, and their comparative features

### INTRODUCTION

Decision Trees are a simple way to guide one’s path to a decision. The decision may be a simple binary one, for example, 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.

#### Caselet: Predicting Heart Attacks using Decision Trees

*A study was done at UC San Diego concerning heart disease patient data. The patients were diagnosed with a heart attack from chest pain, diagnosed by EKG, high enzyme levels in their heart muscles, etc. The objective was to predict which of these patients was at risk of dying from a second heart attack within the next 30 days. The prediction would determine the treatment plan, such as whether to keep the patient in intensive care or not. For each patient more than 100 variables were collected, including demographics, medical history and lab data. Using that data, and the CART algorithm, a decision tree was constructed.*

*The decision tree showed that if blood pressure was low (<= 90), the chance of an- other heart attack was very high (70%). If the patient’s BP was ok, the next question to ask was the patient’s age. If the age was low (<= 62), then the patient’s survival was almost guaranteed (98%). If the age was higher, then the next question to ask*

*was about sinus problems. If their sinus was fine, the chances of survival were 89%. Otherwise, the chances of survival dropped to 50%. This decision tree predicts 86.5% of the cases correctly. (Source: Salford Systems)*

1. *Is a decision tree good enough in terms of accuracy, design, and readability, for this data?*
2. *Identify the benefits from creating such a decision tree. Can these be quantified?*

### 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.

* The more data available for training the decision tree, the more accurate its knowledge extraction will be, and thus, it will make more accurate decisions.
* The more variables the tree can choose from, the greater is the accuracy of the decision tree.
* 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 ap- proving 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 this question, one should look at the past experiences, 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. Dataset 6.1 shows a list of the decisions taken in 14 instances of past soccer game situations. (Dataset courtesy: Witten, Frank, and Hall, 2010)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Dataset 6.1 |  |  |  |  |
| 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 was a row for Sunny/Hot/Normal/Windy condition in the data table, it would match the current problem; 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 in- stance, if the data for humidity variable was not available, looking up the past data would not help.

A better way of solving the problem is 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 to 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 questions 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 the Root Node of the Tree

In this example, there are four choices based on the four variables. One can 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 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 can be done for other values of the outlook variable. There are four instances where the outlook is overcast. In all the 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 out of 5 errors. This can be recorded in next row.

Attribute Rules Error Total Error

Outlook Sunny -> No 2/5 Overcast -> Yes 0/4

Rainy -> Yes 2/5

4/14

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 the analytical exercise, the following error table (Dataset 6.2) can be constructed.

Dataset 6.2

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

The variable that leads to the least number of errors (and thus the greatest 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 subtrees.

If all the errors were concentrated in few of the subtrees and some of the branches were completely free of error, then that is preferred from a usability perspective. Outlook has one error-free branch, for the overcast value, while there is no such pure subclass 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 subtrees, one for each of the three values of outlook. Data for the root node (the entire data) will be divided into 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 subtree. 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 subtree. 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 as follows (Figure 6.1) after the first level of splitting.

Outlook

Sunny

Rainy

Overcast

Yes

|  |  |  |  |
| --- | --- | --- | --- |
| Temp | Humidity | Windy | Play |
| Hot | High | False | *No* |
| Hot | High | True | *No* |
| Mild | High | False | *No* |
| Cool | Normal | False | *No* |
| 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* |
|  |  |  |  |

FIGURE 6.1

*Determining the Next Nodes of the Tree*

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 – temperature, humidity and windy. Final comparison will look like as shown in Dataset 6.3 given below.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Dataset 6.3 |  | | | |
| Attribute | | Rules | Error | Total Error |

|  |  |  |  |
| --- | --- | --- | --- |
| Temperature | Hot -> No | 0/2 |  |
|  | Mild -> No | 1/2 | 1/5 |
|  | Cool -> Yes | 0/1 |  |
| Humidity | High -> No | 0/3 |  |
|  | Normal -> Yes | 0/2 | 0/5 |
| Windy | False -> No | 1/3 |  |
|  | True -> Yes | 1/2 | 2/5 |

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 following Dataset 6.4 depicts such analysis.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Dataset 6.4 |  | | | |
| Attribute | | Rules | Error | Total Error |

|  |  |  |  |
| --- | --- | --- | --- |
| Temperature | Mild -> Yes | 1/3 |  |
|  | Cool -> Yes | 1/2 | 2/5 |
| Humidity | High -> No | 1/2 |  |
|  | Normal -> Yes | 1/3 | 2/5 |
| Windy | False -> Yes | 0/3 |  |
|  | True -> No | 0/2 | 0/5 |

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 will look like. Here it is produced using Weka open-source data mining platform (Figure 6.2). This is the model that abstracts the knowledge of the past data of decision.

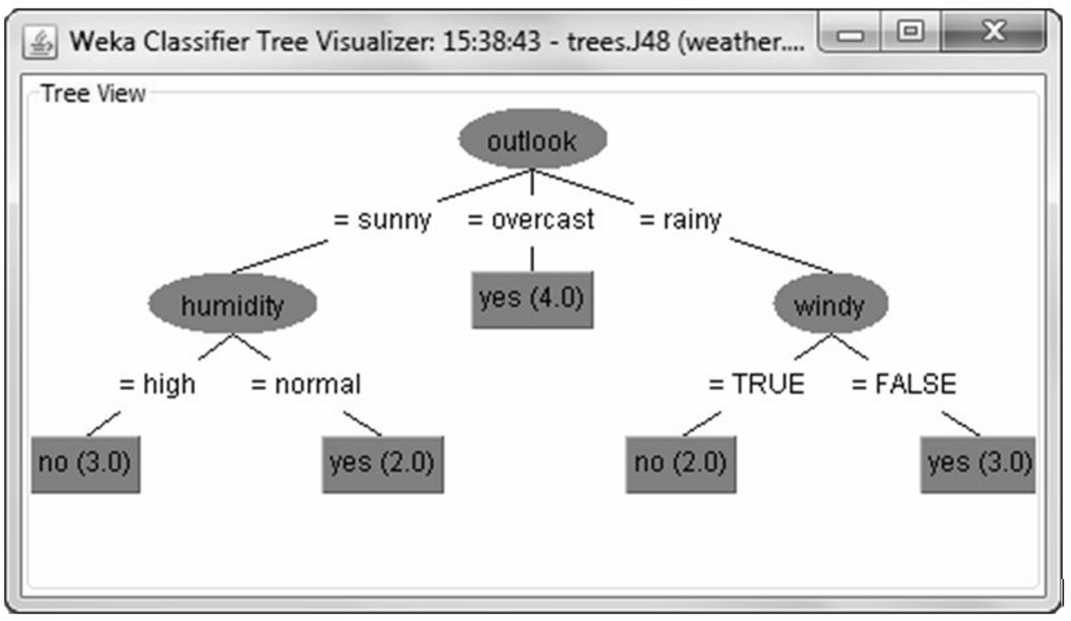


FIGURE 6.2 Decision Tree for the Weather Problem

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Outlook | Temperature | 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 subtree is humidity. In the problem, humidity is normal. That branch leads to an answer – Yes. Thus, the answer to the play problem is a yes.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Outlook | Temperature | Humidity | Windy | Play |
| Sunny | Hot | Normal | True | Yes |

### LESSONS FROM CONSTRUCTING TREES

Here are some benefits of using the decision tree compared with looking up the answers from the data table (Table 6.1)

Table 6.1 Comparing Decision Tree with Table Lookup

|  |  |  |
| --- | --- | --- |
|  | 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 |
| Simplicity | Only one, or maximum two variable values are needed | All four variable values are needed |
| Ease | Logical and easy to understand | Can be cumbersome to look up; no understanding of the logic behind the decision |

Here are a few observations about how the trees was constructed

* 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 datasets, 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.
* 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.
* 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.
* 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.
* 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.
* 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 dataset. 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 subtree?

Algorithms use different measures like least errors, information gain, Gini’s coefficient etc., to compute the splitting variable that provides the most benefit. Information gain is a mathematical construct to compute the reduction in information entropy from a prior state to the next state that takes some information as given. The greater the reduction in entropy, the better it is. The Gini coefficient is a statistical concept that measures the inequality among values of a frequency distribution. The lower the Gini’s coefficient, the better it is.

1. 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?
2. 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*

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

*Pruning*

It is the act of reducing the size of decision trees by removing sections of the tree that provide little value. The decision tree could be trimmed to make it more balanced, more general and more easily usable. The symptoms of an over-fitted tree are that it is too deep with too many branches which may reflect anomalies due to random noise or outliers instead of the underlying relationship. Pruning is often done after the tree is constructed. There are two approaches to avoid over-fitting.

* Prepruning 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.
* Postpruning means removing branches or subtrees 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)

Table 6.2 Comparing Popular Decision Tree Algorithms

|  |  |  |  |
| --- | --- | --- | --- |
| Decision Tree | C4.5 | CART | CHAID |
| Full name | Iterative Dichotomizer (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 |

(*contd.*)

|  |  |  |  |
| --- | --- | --- | --- |
| When developed | 1986 | 1984 | 1980 |
| Type of trees | Classification | Classification and Regression trees | Classification and Regression |
| Serial implementation | Tree growth and Tree pruning | Tree growth and Tree pruning | Tree growth and Tree pruning |
| Type of data | Discrete and Continuous; Incomplete data | Discrete and Continuous | Non-normal data also accepted |
| Type of splits | Multi-way splits | Binary splits only; clever surrogate splits to reduce tree depth | Multiway splits as de- fault |
| Splitting criteria | Information gain | Gini’s coefficient, and others | *Chi*-square test |
| Pruning criteria | Clever bottom-up technique avoids over-fit- ting | Remove weakest links first | Trees can become very large |
| Implementation | Publicly available | Publicly available in most packages | Popular in market re- search for segmentation |

## Conclusion

Decision trees are the most popular, versatile, and easy to use data mining technique with high predictive accuracy. They are also very useful as communication tools with executives. There are many successful decision tree algorithms. All publicly available data mining software platforms offer multiple decision tree implementations.

## Questions

1. What is a decision tree? Why are decision trees the most popular classification technique?
2. What is a splitting variable? Describe three criteria for choosing a splitting variable.
3. What is pruning? What are prepruning and postpruning techniques? Why choose one over the other?
4. What are Gini’s coefficient and information gain?

*Hands-on Exercise*

Create a decision tree for the for the data given in Dataset 6.5. The objective is to predict the class category (Loan approved or not).

Dataset 6.5

Age

Job

House

Credit

Loan Ap-

proved

Young False No Fair No

Young False No Good No

Young True No Good Yes

Young True Yes Fair Yes

Young False No Fair No

Middle False No Fair No

Middle False No Good No

Middle True Yes Good Yes   
Middle False Yes Excellent Yes

Middle False Yes Excellent Yes   
Old False Yes Excellent Yes

Old False Yes Good Yes

Old True No Good Yes

Old True No Excellent Yes

Old False No Fair No

Then solve the following problem using the model.

Age

Job

House

Credit

Loan Ap-

proved

Young False No Good ??

## True/False

1. Decision trees are essentially a hierarchy of if-then statements.
2. Decision trees apply only to strategic decisions made by executives.
3. A good decision tree should be short and ask only a few meaningful ques- tions.
4. A decision tree should be balanced so it can make accurate predictions.
5. Decision trees use statistical techniques to abstract knowledge from data.
6. A good decision tree does not need to be colorful.
7. A decision tree can have any number of branches at any of the nodes.
8. It is desirable to have a 100% predictive accuracy of the tree, even if the tree becomes very long.
9. Making a decision tree is a recursive process.
10. The way of selecting the most important variable in constructing a decision tree is called the splitting criteria.