Decision Trees and Surveys

Larry Fenn

Contents

1 Decision Trees

Decision trees are tools that showcase "decisions" and their possible outcomes. They have extensive utility in displaying how an algorithm works, or in determining the optimal strategy (which is nothing more than a sequence of decisions), or in classifying data. Abstractly, a decision tree is a directed tree graph. The root node is the source, and every node thereafter is either a *decision node* or an *end node*. Decision nodes have multiple outward-pointing edges, and there is a *splitting rule* at each node governing the significance of each edge leading away. It is this splitting rule that constitutes the "decisions" that a decision tree displays. The splitting rule itself can be any conditional statement:

- Deterministic: "If you have previously filed your taxes this year, go to node X; else, go to node Y."
- Stochastic: "With 50% probability go to node X or Y."
- Non-numeric: "If today is Monday, go to node X; else, go to node Y."

The following definitions from [10, p. 660] will prove useful:

- An object can be described entirely by a set of *attributes*, each of which can be ordered (such as numerical data) or unordered (such as boolean data).
- The *domain* is the set of all objects that are valid inputs for a decision tree.
- A *class* is a label assigned to objects in the domain. Each leaf node of a decision tree has a class identified with it.
- The *concept* is the "true" mapping (defined based on prior knowledge about the domain)from attributes to a class. A decision tree is a mapping from attributes to the class. Often the concept is not explicitly known, such as in natural language processing problems.
- A *goodness measure* is a function mapping all possible splitting rules to a numerical score for comparison.[7, p. 347]
- *Discrimination* is the process of deriving decision tree nodes from existing data sets.
- *Classification* is the process of applying an existing decision tree to an object to determine its class.
- An object in the domain is classified by starting at the root node, and following the splitting rules to subsequent nodes until a leaf node is reached. The class identified with the leaf node is thus assigned to the object.
- The proportion of objects for which the decision tree correctly assigns the class label (based on the concept) is the *accuracy* of the decision tree; conversely, the proportion of objects for which the decision tree assigns a class label that is not correct is the *error* of the decision tree.

The end nodes are the leaf nodes of the graph; they have only one directed edge leading into them. In some circumstances end nodes can be identified together, allowing some degree of violation to the tree structure. For example, a decision tree for the yes or no decision "should I buy this object" may have many decisions and consequently very many end nodes; but ultimately, the only outcomes can be "yes" or "no".

The design of a decision tree thus amounts to constructing splitting rules in the same way one would construct steps of an algorithm. In particular, it is possible to use decision trees to structure an algorithm for classifying data: if we start with a root node, each additional splitting rule we add will have the effect of partitioning the data set depending on the different outcomes of the splitting rule. For example, consider the following decision tree from cladistics in evolutionary biology. In this instance, the purpose of the tree is to provide a classification scheme for the domain of "small insects".

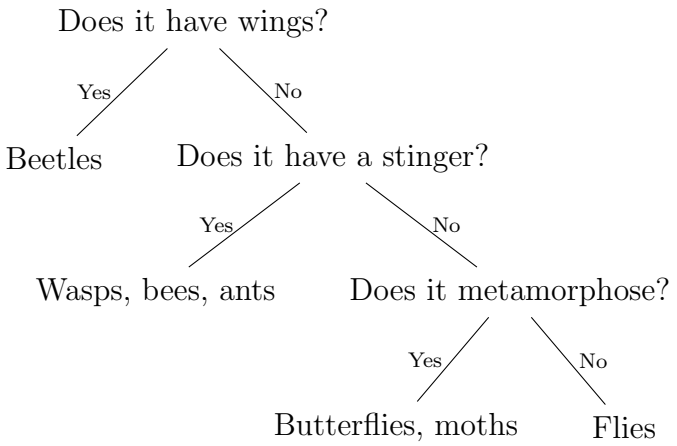

Figure 1: Taxonomy of certain insects

If we examine all the leaf nodes, we see that the only possible outputs of the algorithm are "beetles", "wasps, bees, ants", "butterflies, moths", and lastly "flies". Moreover, consider the parent node to "flies" and "butterflies, moths": any object that we are attempting to classify with a decision tree that reaches this node must *necessarily* have wings and no stinger; else we would not be at this node. In other words, an edge between two vertices directly corresponds with an outcome for the splitting rule; traversing that edge means that the object under scrutiny must satisfy that particular conditional in the splitting rule. This principle of decision trees representing partitioning of a set can be generalized to arbitrary paths on decision trees: [10, p. 663]

Given a set S, a *filtration* of S is a collection of subsets S_i such that if $i \leq j$, then $S_i \subseteq S_j$. In the context of decision trees, each path through the tree results in a filtration:

Proposition 1. *For any decision tree on domain D, all paths on the decision tree describe a filtration of D.*

Proof. By induction on length of paths:

Base case: for a path of length 1 in a decision tree, we represent it as the edge e_1. This edge e_1 is traversed only when the splitting rule at the start of the edge allows it. Thus if we evaluate this splitting rule over all elements of D, let S_1 be a subset of D consisting of those objects in D that obey the splitting rule. We can thus identify S_1 with the edge e_1. Thus we have that $S_1 \subseteq D$ is a filtration of D.

Inductive hypothesis: for all paths of length $n-1$ in a decision tree, every path describes a filtration $\mathcal{F}_{n-1}$ of D.

For a path of length n in the decision tree (denoted $e_1e_2...e_n$ by edges, or $v_0v_1...v_n$ by vertices), we can split it into a path of length $n-1$ leading to vertex v_{n-1}, and then a path of length 1 from vertex v_{n-1} to v_n. By the inductive hypothesis, the length $n-1$ path describes a filtration of D where $S_i \subseteq S_j$ for $i \leq j$: $\{S_1, S_2, ..., S_{n-1}\}$. The vertex v_{n-1} will correspond with the smallest set in the filtration S_1; now, the edge e_n between v_{n-1} and v_n is defined by the splitting rule at vertex v_{n-1}. So we can define a new set, $S_0 \subseteq S_1$, consisting of all objects in S_1 that follow that splitting rule. Thus the path of length n defines the filtration $\{S_0, S_1, ..., S_{n-1}\}$. □

Every path from the root node to a leaf node is therefore equivalent to a conditional statement of the form "if [1] and [2] and ... and [n], then [c]", where [1], [2], ... are the conditions given by the splitting rules guiding the path and [c] is the class label at the leaf node. It is this identification that allows for the ability to identify a node in a tree with a certain subset of a filtration, by way of the path from the root node to that node. Objects can be said to be "at" a node (equivalently, a step in the algorithm) if they are a member of that node's subset of the filtration. Moreover, if we assume that questions are nontrivial in the sense that any question has more than one response that is possibly valid, then this filtration definition implies that cycles cannot exist in the decision tree structure (since no set can be a proper subset of itself).

In the context of survey construction, we will refer to an *analytical outcome* as a class of objects that cannot be distinguished using the questions from a survey; the *analytical potential* is the set of all analytical outcomes.

2 Decision Trees as Surveys

The goal of a survey is to classify a population into sets depending on their responses. For example, a survey might be used to determine the demographics of voters for various candidates. In this context, we can both represent the survey as a decision tree as a descriptive tool, or use decision tree methodologies to create a survey with the target analytical outcome.

In general, decision trees can be thought of as a data exploration tool, one that can quickly organize and classify data by determination of (ideally) simple tests on the data set [7, p. 345].

2.1 Descriptive Decision Trees

Given a survey and a data set of its responses, we can construct the decision tree for the survey directly. We make the root node of the decision tree the very first question, and then we create edges to the next level of the tree based on the possible responses. The nodes at the other end of the edge will all be the second question (assuming the survey does not direct its participants) or otherwise, the question that responders are directed to answer next. The splitting rule will be a weight from 0 to 1 representing the proportion each response receives. Now, for the remainder of the tree proceed in a similar fashion: at every node, construct edges for each possible response to the question that node represents, and assign probabilities to each edge based on the proportion of responses given that they have responded accordingly to all of the previous questions represented by the path from the node to that vertex[7, p. 347]. In this way we see that decision trees adapt conditional probabilities; while a single question in the survey may be represented by many nodes, each node in particular represents the different possible prior responses for the survey leading up to that question. The decision tree as a whole will represent the data set of its responses in a way that makes not only the distribution of responses known but also how effective each question was at distinguishing between people surveyed. An ideal question should divide the population based on response as evenly as possible.

2.2 Decision Tree Survey Construction

Decision trees can also be used to design surveys that have an intended classification. The goal of using decision trees to construct surveys is to aid in finding the optimal decision tree for some criterion. For examples in similar contexts, in both evolutionary biology and linguistics tree-like structures are used to determine the most likely configuration of intermediate species given the current-day taxonomy; see [2], [11].

The problem is an optimization problem. Subject to some analytical potential that we wish to achieve, what is the proper determination of splitting rules in the decision tree that will describe this analytical outcome and optimizes some other quality?

For a trivial example: we can establish a decision tree consisting of a root and the splitting rule at the root simply has every analytical outcome in its conditional statement. Thus this tree will have as few nodes as possible, and be as short as possible, but this amounts to writing every question in the survey as one enormous compound question.

Even if we mandate that every splitting rule can only constitute one question, there are lots of possible determinations that depend on what is being optimized. For example, to construct a survey whose questions are as simple as possible, we would use only binary splitting rules- yes or no questions. However, if we were seeking to use a survey for non-categorical data, this approach is not ideal. In order to determine a numerical quantity, the all yes-or-no decision tree would have to represent inside it what amounts to binary search over the integers- when a multiple-response "binning" question may be more appropriate.

The following procedure is one way to construct a decision tree whose survey has some given analytical potential [7, p. 349]:

- If all possible objects at the current node occupy the same analytical outcome, make the node a leaf node identified with that outcome.
- If not, score all possible splits of the objects at the current node using a goodness measure and choose the best split.
- Create child nodes based on the best split, and partition all possible objects at the current node based on that splitting rule. Each child is now identified with a subset of all possible objects at the current node.
- Repeat on all non-leaf nodes.

The question of which heuristics to use to produce decision trees has been covered in the past in multiple papers[5]. The main heuristics considered here will be:

1. Minimize the expected number of questions (survey brevity).
2. Minimize the expected number of possible responses to a question (question simplicity).

There is a fundamental trade off between these two heuristics:

Proposition 2. *Given a number N of classes, let $T \in \mathcal{T}_N$ be an arbitrary decision tree that has at least N classes. If D_T represents the expected number of questions total in the decision tree T, then for any fixed value B of expected number of possible responses we have:*

$$\left\lceil \frac{\log N}{\log B} \right\rceil \leq \min_{T \in \mathcal{T}_N} D_T$$

Proof. As mentioned earlier, the optimal splitting rule at any node is one that equally partitions a set to each of the child nodes. Thus, if B is fixed, the optimal tree with this value of B is one where every node has precisely B many children; with N many distinct classes represented as N leaf nodes, we have that $\left\lceil \frac{\log N}{\log B} \right\rceil$ is a lower bound representing the most ideally-balanced tree where every node (except for perhaps those immediately preceding the leaf nodes) has B children. Any other decision tree must have at least this depth. □

This represents a rather intuitive fact about classifying objects: in order to achieve a richer analytical potential (i.e. discriminate between more classes), more questions or more detailed questions are required.

2.3 Case Study: EM Survey

The analytical potential used here will be a simplified version of the analytic potential for the EM survey. In particular, suppose that the only outcomes of the survey are as follows:

- Unemployed

- Employee at a family business.
- Employee in a non-family business with union status U1, union coverage U2, with job duration J.
- Self-employed.

The attributes of the problem domain are thus characterized, at most, by the following list:

- Union coverage given employed: true or false.
- Union status given union coverage: true or false.
- Employment status: four possibilities (unemployed, family business, self employed, employed).
- Job duration given employed: five possibilities (permanent, seasonal, temporary, term, casual).

There are 19 total possibilities for responses (4 employment statuses, 15 sub-possibilities given employment): thus our target decision tree must have at least 19 distinct leaf nodes in order to correctly classify via this analytic potential.

The first tree to be considered will be a binary tree structure. For 19 distinct leaf nodes, at least 5 questions are required. Let the balance of a node be defined as $\alpha = \dfrac{1+L}{2+L+R}$, where L and R are the number of nodes in the left and right sub-tree respectively [10]. The goodness measure will be the score $1 - 4\left(\alpha - \dfrac{1}{2}\right)^2$.

Among all possible splits of the 19 total possibilities, the best possible balance is for sub-trees of size 9 and 10. This is achieved with the question "Were you employed with union coverage?": there are ten possibilities given employment with union coverage (five job durations and two possible union membership statuses), and nine otherwise.

Given employment with union coverage, the best possible balance is for two sub-trees of size five. This is achieved with the question "Were you a union member?": a yes answer has five possibilities for job duration, and a no answer has five (those same five job durations).

From here, both sub-trees are virtually identical: they accomplish the task of determining job duration. An example series of questions is "Was this job permanent or seasonal?" followed by "Was this job temporary?" and lastly "Was this job casual?".

Returning to the other possibility, a similar sequence of questions suffices to split by the balance goodness measure.

The other half of the tree proceeds similarly.

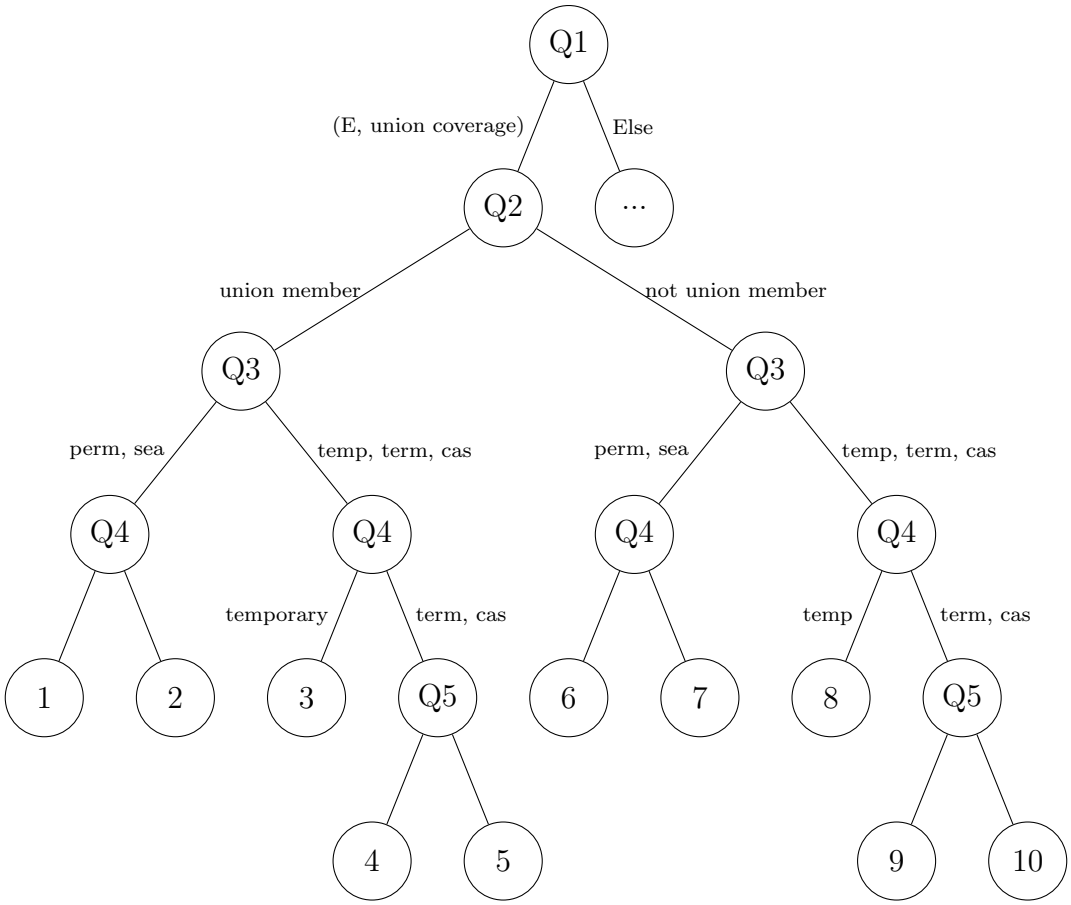

Figure 2: Balanced Tree Survey

The second goodness measure for tree construction will be expected number of questions asked. The approximation to expected number of questions asked will be implement a constraint on the maximum depth of the tree. This is primarily because this specific problem does not have any prior probability distribution on possible responses. The method to accomplish this will be to use a tree structure similar to a B-tree: instead of only permitting binary responses, suppose that up to five responses per question are allowed in the design; moreover, that the depth of the tree is now constrained at 2. The limitation is increasing the number of question responses (generally) increases the question complexity. However, the notion of question complexity is not directly linked with the number of possible responses; other considerations, such as brevity in statement and simplicity of statement, are significant [9].

With 19 outcomes, increasing the number of responses to five and constraining to two questions results in a decision tree such as the following:

- Question 1: "Were you (a) either unemployed, employed at a family business, or self-employed, (b) employed with union status and coverage, (c) with union coverage and without union status, or (d) without union status and without union coverage?"

- Question 2.1: "Were you (a) unemployed, (b) employed at a family business, or (c) self-employed?"
- Question 2.2: "Was the job (a) permanent, (b) seasonal, (c) term, (d) temporary, or (e) casual?"

The tree is as follows. Note the prevalence of very similar sub-trees, just as in the above case:

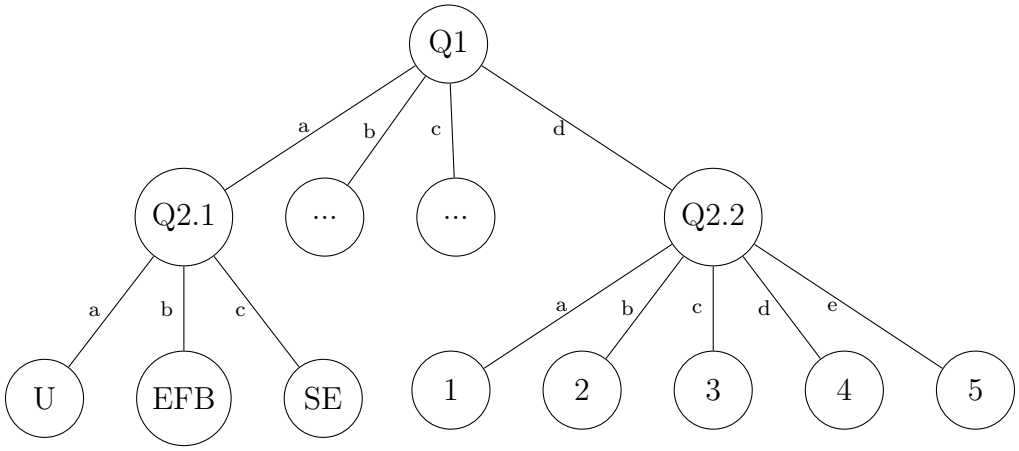

Figure 3: Shorter, Broader Tree Survey

In this case, the question of choosing the proper depth or breadth for a tree was informed primarily by the presence of a categorical variable with five categories. In general, discussions on optimal decision tree design can be found in [10]. Minimizing the size (by number of nodes) of a decision tree is known to be NP-hard[5].

One strength of decision trees is that they are very flexible in their design: the analytical potential can be made coarser or finer as necessary, and the goodness measure can be chosen in a manner that is highly tailored to application. This can be a double-edged sword, as the NP-hard nature of decision tree size optimization indicates.

One significant weakness in the two decision trees above is that they do not have any non-response options. This can be potentially rectified by adding to the space of each attribute a "NA" possibility, thus increasing the overall size of the tree significantly. Thus, instead of a true/false response for union membership there would be a true/false/NA response. This is subject to the survey designer's considerations for treatment of non-responses.

Lastly, the decision trees above also suffer from poor handling of erroneous responses by survey participants. Each splitting rule is predicated on completely accurate responses. One potential way to address this is to add redundant questions. Discussion of constructing surveys that perform well with erroneous responses is difficult because any survey would suffer from this vulnerability; surveys are not lie-detectors.

3 Decision Tree Transformations

Externally, modifying a survey to change its analytical potential amounts to changing the possible classes in a decision tree (i.e. its leaf nodes). However, at any given internal node the selection of an optimal splitting rule can result in different splitting rules being chosen when classes are modified. In general, splitting rules can be replaced by only very small changes in the survey; in [6] an example is given where only a 3% change is sufficient to replace a splitting rule.

Internally, it is very easy for a survey designer to manipulate the decision tree and get new surveys. The simplest operation is to make a question simpler: noting the prevalence of very similar sub-trees in the example decision trees, a question with multiple responses can be split by grouping together some of the responses and inserting internal nodes that lead to them. In a diagram:

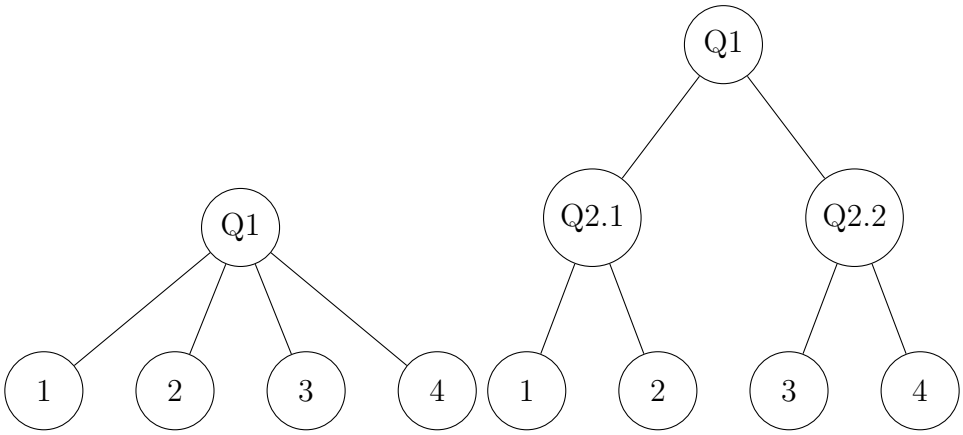

Figure 4: Splitting and Merging Questions

Similarly, questions can be removed by doing the reverse of this process. Given a node with children $\{C_1, C_2, ..., C_n\}$ with splitting rule given by "if P_i, go to C_i", it is possible to accumulate all of the splitting rules from the children. For each child C_i, let the splitting rule at C_i be "if Q_k^i, go to node N_k^i". The accumulated splitting rule at the end then becomes a combination of all possible child rules with the original splitting rule: "if P_i and Q_k^i, go to N_k^i".

The ability to merge questions indicates that it is possible to merge splitting trees as well. Thus, a globally complex problem can be approached by first creating decision trees for much simpler sub-problems, and then merging the trees together. For example, the lower sub-trees for the balanced sub-tree from earlier serves the purpose of locally identifying employment duration. The drawback of this approach is that it results in many more leaf nodes than is actually necessary [10].

Rearranging questions is far more problematic. The issue is that the splitting rule at

an internal node represents a conditional statement of the form "Given (all the statements represented by the path from the root node to here), go to nodes according to this rule". This indicates that the splitting rule at a given node is dependent on all of the nodes between the root node and the given node. Thus, all children of a node that is modified must be updated to account for the modified conditional statements. For example, in the employment survey above- asking either of question 2.1 or question 2.2 first would not make any logical sense, since the preconditions they depend on (employment, union membership, and union coverage) no longer exist and therefore must be added to the question. Similarly, the responses that formerly took for granted some given information must now be modified to reflect the lack of that given information: for example, if we were to ask question 2.1 first then the possible responses must now include "unemployment", a response that used to be in the parent node (question 1).

4 Survey Charts Comparison

Survey charts are another approach based on graphs that are used for survey construction and design. The survey chart itself is a graphical representation of a survey, where every question is a node and directed edges between nodes represent the order in which questions are asked. There are two transformations for survey charts, which serve as tools to derive other surveys without changing the analytical potential. The main comparisons between survey charts and decision trees will be how surveys are represented and how transformations of surveys are implemented.

Decision trees require much more space than survey charts. This is because there are at least as many leaf nodes as there are classes in the end. This indicates that the total number of nodes is at least as large as the number of possible analytical outcomes. This is in contrast with survey charts, which have as many nodes as questions and conditions in the survey. Survey charts do contain edges that are never traversed because the conditions for traversal would be contradictory; in contrast, for decision trees, every splitting rule (and subsequently, every edge) must represent a distinction between different, nonempty analytical outcomes.

There are two types of transformations for survey charts. One of the transformations is analogous to the merging transformation for decision trees: combining questions to simplify the overall structure of the survey. The other transformation on survey charts entails rearranging questions. For decision trees, the high dependence between a node and all of its children means rearranging questions may require significant changes to the topology of the tree. In contrast, the rearranging transformation in survey charts allows for survey designers to manipulate the order of the survey to arrive at progressively improved surveys with respect to expected number of questions. The transformation of splitting questions on decision trees does give designers the opportunity to simplify their questions. The consequence of splitting a question, however, is that it adds to the overall length of the survey.

Unlike survey charts, decision trees can be employed to create surveys.. Given an analytical potential and goodness measure, a decision tree can be made through inductive

construction as in [7], [10]. This suggests the following strategy for survey designers: if a target analytical potential is desired, first employ a decision tree discrimination process to build the corresponding decision tree, then use a survey chart approach for additional improvement.

References

[1] Zhang D., Zhou X., Leung S., and Zheng J. Vertical bagging decision trees model for credit scoring. *Exp Syst Appl*, 37:7838–7843, 2010.

[2] R. D. Gray and Q. D. Atkinson. Language-tree divergence times support the Anatolian theory of Indo-European origin. *Nature*, 426:435–439, 2003.

[3] L. Hyafil and R. L. Rivest. Constructing optimal binary decision trees is NP-complete. *Information Processing Letters*, 5(1):15–17, 1976.

[4] D. E. Knuth. Optimum binary search trees. *ACTA Informatica*, 1:14–25, 1971.

[5] S. B. Kotsiantis. Decision trees: a recent overview. *Artificial Intelligence Review*, 39:261–283, 2013.

[6] R. H. Li and G. G. Belford. Instability of decision tree classification algorithms. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 570–575, Edmonton, 2002.

[7] S. K. Murthy. Automatic construction of decision trees from data: A multi-disciplinary survey. *Data Mining and Knowledge Discovery*, 2:345–389, 1998.

[8] D. S. Nau. Decision quality as a function of search depth on game trees. *Journal of the Association of Computing Machinery*, 30(4):687–708, 1983.

[9] S. L. Payne. Case study in question complexity. *Public Opinion Quarterly*, 13:653–658, 1949.

[10] S. R. Safavian and D. Landgrebe. A survey of decision tree classifier methodology. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(3):660–674, 1991.

[11] M. Serva and F. Petroni. Indo-European languages tree by Levenshtein distance. *Europhysics Letters*, 81, 2008.