i) $\times$ (number of tuples in class j)/N, where N is the total number of data tuples. Low χ^2 values for an interval pair indicate that the intervals are independent of the class and can, therefore, be merged.

The stopping criterion is typically determined by three conditions. First, merging stops when χ^2 values of all pairs of adjacent intervals exceed some threshold, which is determined by a specified significance level. A too (or very) high value of significance level for the χ^2 test may cause overdiscretization, whereas a too (or very) low value may lead to underdiscretization. Typically, the significance level is set between 0.10 and 0.01. Second, the number of intervals cannot be over a prespecified *max-interval*, such as 10 to 15. Finally, recall that the premise behind ChiMerge is that the relative class frequencies should be fairly consistent within an interval. In practice, some inconsistency is allowed, although this should be no more than a prespecified threshold, such as 3%, which may be estimated from the training data. This last condition can be used to remove irrelevant attributes from the data set.

Cluster Analysis

Cluster analysis is a popular data discretization method. A clustering algorithm can be applied to discretize a numerical attribute, A, by partitioning the values of A into clusters or groups. Clustering takes the distribution of A into consideration, as well as the closeness of data points, and therefore is able to produce high-quality discretization results. Clustering can be used to generate a concept hierarchy for A by following either a top-down splitting strategy or a bottom-up merging strategy, where each cluster forms a node of the concept hierarchy. In the former, each initial cluster or partition may be further decomposed into several subclusters, forming a lower level of the hierarchy. In the latter, clusters are formed by repeatedly grouping neighboring clusters in order to form higher-level concepts. Clustering methods for data mining are studied in Chapter 7.

Discretization by Intuitive Partitioning

Although the above discretization methods are useful in the generation of numerical hierarchies, many users would like to see numerical ranges partitioned into relatively uniform, easy-to-read intervals that appear intuitive or "natural." For example, annual salaries broken into ranges like ($50,000, $60,000] are often more desirable than ranges like ($51,263.98, $60,872.34], obtained by, say, some sophisticated clustering analysis.

The 3-4-5 rule can be used to segment numerical data into relatively uniform, natural-seeming intervals. In general, the rule partitions a given range of data into 3, 4, or 5 relatively equal-width intervals, recursively and level by level, based on the value range at the most significant digit. We will illustrate the use of the rule with an example further below. The rule is as follows:

- If an interval covers 3, 6, 7, or 9 distinct values at the most significant digit, then partition the range into 3 intervals (3 equal-width intervals for 3, 6, and 9; and 3 intervals in the grouping of 2-3-2 for 7).

- If it covers 2, 4, or 8 distinct values at the most significant digit, then partition the range into 4 equal-width intervals.
- If it covers 1, 5, or 10 distinct values at the most significant digit, then partition the range into 5 equal-width intervals.

The rule can be recursively applied to each interval, creating a concept hierarchy for the given numerical attribute. Real-world data often contain extremely large positive and/or negative outlier values, which could distort any top-down discretization method based on minimum and maximum data values. For example, the assets of a few people could be several orders of magnitude higher than those of others in the same data set. Discretization based on the maximal asset values may lead to a highly biased hierarchy. Thus the top-level discretization can be performed based on the range of data values representing the majority (e.g., 5th percentile to 95th percentile) of the given data. The extremely high or low values beyond the top-level discretization will form distinct interval(s) that can be handled separately, but in a similar manner.

The following example illustrates the use of the 3-4-5 rule for the automatic construction of a numerical hierarchy.

Example 2.6 Numeric concept hierarchy generation by intuitive partitioning. Suppose that profits at different branches of *AllElectronics* for the year 2004 cover a wide range, from −$351,976.00 to $4,700,896.50. A user desires the automatic generation of a concept hierarchy for *profit*. For improved readability, we use the notation $(l...r]$ to represent the interval $(l, r]$. For example, (−$1,000,000...$0] denotes the range from −$1,000,000 (exclusive) to $0 (inclusive).

Suppose that the data within the 5th percentile and 95th percentile are between −$159,876 and $1,838,761. The results of applying the 3-4-5 rule are shown in Figure 2.23.

1. Based on the above information, the minimum and maximum values are $MIN =$ −$351,976.00, and $MAX =$ $4,700,896.50. The low (5th percentile) and high (95th percentile) values to be considered for the top or first level of discretization are $LOW =$ −$159,876, and $HIGH =$ $1,838,761.
2. Given LOW and HIGH, the most significant digit (*msd*) is at the million dollar digit position (i.e., $msd = 1{,}000{,}000$). Rounding LOW down to the million dollar digit, we get $LOW' =$ −$1,000,000; rounding $HIGH$ up to the million dollar digit, we get $HIGH' =$ +$2,000,000.
3. Since this interval ranges over three distinct values at the most significant digit, that is, $(2{,}000{,}000 - (-1{,}000{,}000))/1{,}000{,}000 = 3$, the segment is partitioned into three equal-width subsegments according to the 3-4-5 rule: (−$1,000,000...$0], ($0...$1,000,000], and ($1,000,000...$2,000,000]. This represents the top tier of the hierarchy.

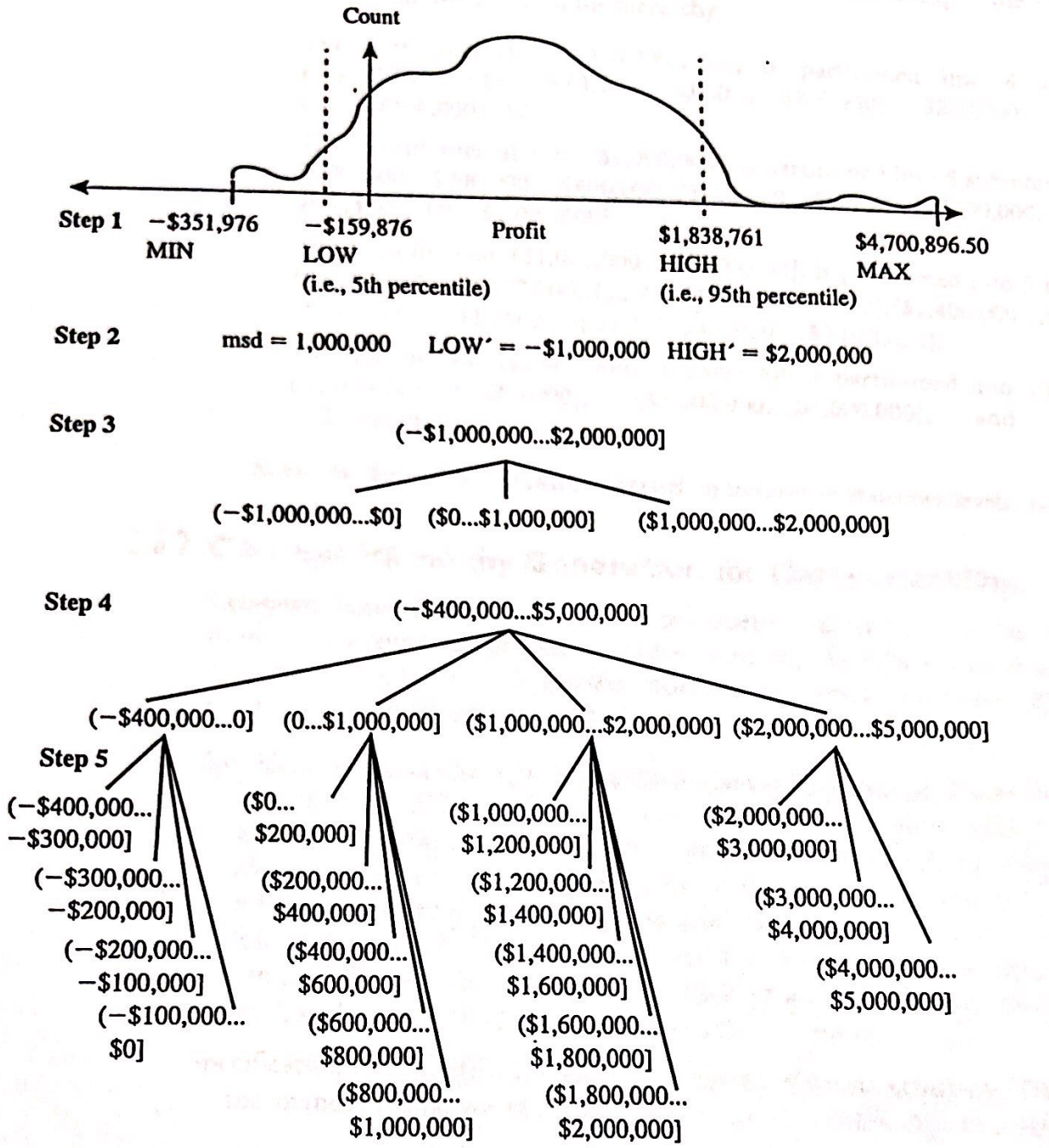

Figure 2.23 Automatic generation of a concept hierarchy for *profit* based on the 3-4-5 rule.

4. We now examine the MIN and MAX values to see how they "fit" into the first-level partitions. Since the first interval (−$1,000,000...$0] covers the *MIN* value, that is, $LOW' < MIN$, we can adjust the left boundary of this interval to make the interval smaller. The most significant digit of *MIN* is the hundred thousand digit position.

Rounding MIN down to this position, we get $MIN' = -\$400{,}000$. Therefore, the first interval is redefined as $(-\$400{,}000 \ldots 0]$.
Since the last interval, $(\$1{,}000{,}000 \ldots \$2{,}000{,}000]$, does not cover the MAX value, that is, $MAX > HIGH'$, we need to create a new interval to cover it. Rounding up MAX at its most significant digit position, the new interval is $(\$2{,}000{,}000 \ldots \$5{,}000{,}000]$. Hence, the topmost level of the hierarchy contains four partitions, $(-\$400{,}000 \ldots \$0]$, $(\$0 \ldots \$1{,}000{,}000]$, $(\$1{,}000{,}000 \ldots \$2{,}000{,}000]$, and $(\$2{,}000{,}000 \ldots \$5{,}000{,}000]$.

5. Recursively, each interval can be further partitioned according to the 3-4-5 rule to form the next lower level of the hierarchy:

   - The first interval, $(-\$400{,}000 \ldots \$0]$, is partitioned into 4 subintervals: $(-\$400{,}000 \ldots -\$300{,}000]$, $(-\$300{,}000 \ldots -\$200{,}000]$, $(-\$200{,}000 \ldots -\$100{,}000]$, and $(-\$100{,}000 \ldots \$0]$.
   - The second interval, $(\$0 \ldots \$1{,}000{,}000]$, is partitioned into 5 subintervals: $(\$0 \ldots \$200{,}000]$, $(\$200{,}000 \ldots \$400{,}000]$, $(\$400{,}000 \ldots \$600{,}000]$, $(\$600{,}000 \ldots \$800{,}000]$, and $(\$800{,}000 \ldots \$1{,}000{,}000]$.
   - The third interval, $(\$1{,}000{,}000 \ldots \$2{,}000{,}000]$, is partitioned into 5 subintervals: $(\$1{,}000{,}000 \ldots \$1{,}200{,}000]$, $(\$1{,}200{,}000 \ldots \$1{,}400{,}000]$, $(\$1{,}400{,}000 \ldots \$1{,}600{,}000]$, $(\$1{,}600{,}000 \ldots \$1{,}800{,}000]$, and $(\$1{,}800{,}000 \ldots \$2{,}000{,}000]$.
   - The last interval, $(\$2{,}000{,}000 \ldots \$5{,}000{,}000]$, is partitioned into 3 subintervals: $(\$2{,}000{,}000 \ldots \$3{,}000{,}000]$, $(\$3{,}000{,}000 \ldots \$4{,}000{,}000]$, and $(\$4{,}000{,}000 \ldots \$5{,}000{,}000]$.

Similarly, the 3-4-5 rule can be carried on iteratively at deeper levels, as necessary. ■

2.6.2 Concept Hierarchy Generation for Categorical Data

Categorical data are discrete data. Categorical attributes have a finite (but possibly large) number of distinct values, with no ordering among the values. Examples include *geographic location*, *job category*, and *item type*. There are several methods for the generation of concept hierarchies for categorical data.

Specification of a partial ordering of attributes explicitly at the schema level by users or experts: Concept hierarchies for categorical attributes or dimensions typically involve a group of attributes. A user or expert can easily define a concept hierarchy by specifying a partial or total ordering of the attributes at the schema level. For example, a relational database or a dimension *location* of a data warehouse may contain the following group of attributes: *street*, *city*, *province_or_state*, and *country*. A hierarchy can be defined by specifying the total ordering among these attributes at the schema level, such as $street < city < province\_or\_state < country$.

Specification of a portion of a hierarchy by explicit data grouping: This is essentially the manual definition of a portion of a concept hierarchy. In a large database, it