

attributes, like *street*, can be generalized to higher-level concepts, like *city* or *country*. Similarly, values for numerical attributes, like *age*, may be mapped to higher-level concepts, like *youth*, *middle-aged*, and *senior*.

- **Normalization**, where the attribute data are scaled so as to fall within a small specified range, such as -1.0 to 1.0 , or 0.0 to 1.0 .
- **Attribute construction** (or *feature construction*), where new attributes are constructed and added from the given set of attributes to help the mining process.

Smoothing is a form of data cleaning and was addressed in Section 2.3.2. Section 2.3.3 on the data cleaning process also discussed ETL tools, where users specify transformations to correct data inconsistencies. Aggregation and generalization serve as forms of data reduction and are discussed in Sections 2.5 and 2.6, respectively. In this section, we therefore discuss normalization and attribute construction.

An attribute is normalized by scaling its values so that they fall within a small specified range, such as 0.0 to 1.0 . Normalization is particularly useful for classification algorithms involving neural networks, or distance measurements such as nearest-neighbor classification and clustering. If using the neural network backpropagation algorithm for classification mining (Chapter 6), normalizing the input values for each attribute measured in the training tuples will help speed up the learning phase. For distance-based methods, normalization helps prevent attributes with initially large ranges (e.g., *income*) from outweighing attributes with initially smaller ranges (e.g., binary attributes). There are many methods for data normalization. We study three: *min-max normalization*, *z-score normalization*, and *normalization by decimal scaling*.

Min-max normalization performs a linear transformation on the original data. Suppose that \min_A and \max_A are the minimum and maximum values of an attribute, A . Min-max normalization maps a value, v , of A to v' in the range $[\text{new_min}_A, \text{new_max}_A]$ by computing

$$v' = \frac{v - \min_A}{\max_A - \min_A} (\text{new_max}_A - \text{new_min}_A) + \text{new_min}_A. \quad (2.11)$$

Min-max normalization preserves the relationships among the original data values. It will encounter an "out-of-bounds" error if a future input case for normalization falls outside of the original data range for A .

Example 2.2 Min-max normalization. Suppose that the minimum and maximum values for the attribute *income* are \$12,000 and \$98,000, respectively. We would like to map *income* to the range $[0.0, 1.0]$. By min-max normalization, a value of \$73,600 for *income* is transformed to $\frac{73,600 - 12,000}{98,000 - 12,000} (1.0 - 0) + 0 = 0.716$. ■

In **z-score normalization** (or *zero-mean normalization*), the values for an attribute, A , are normalized based on the mean and standard deviation of A . A value, v , of A is normalized to v' by computing

$$v' = \frac{v - \bar{A}}{\sigma_A}, \quad (2.12)$$

where \bar{A} and σ_A are the mean and standard deviation, respectively, of attribute A . This method of normalization is useful when the actual minimum and maximum of attribute A are unknown, or when there are outliers that dominate the min-max normalization.

Example 2.3 *z-score normalization* Suppose that the mean and standard deviation of the values for the attribute *income* are \$54,000 and \$16,000, respectively. With z-score normalization, a value of \$73,600 for *income* is transformed to $\frac{73,600 - 54,000}{16,000} = 1.225$. ■

Normalization by decimal scaling normalizes by moving the decimal point of values of attribute A . The number of decimal points moved depends on the maximum absolute value of A . A value, v , of A is normalized to v' by computing

$$v' = \frac{v}{10^j}, \quad (2.13)$$

where j is the smallest integer such that $\text{Max}(|v|) < 1$.

Example 2.4 *Decimal scaling*. Suppose that the recorded values of A range from -986 to 917. The maximum absolute value of A is 986. To normalize by decimal scaling, we therefore divide each value by 1,000 (i.e., $j = 3$) so that -986 normalizes to -0.986 and 917 normalizes to 0.917. ■

Note that normalization can change the original data quite a bit, especially the latter two methods shown above. It is also necessary to save the normalization parameters (such as the mean and standard deviation if using z-score normalization) so that future data can be normalized in a uniform manner.

In attribute construction,⁵ new attributes are constructed from the given attributes and added in order to help improve the accuracy and understanding of structure in high-dimensional data. For example, we may wish to add the attribute *area* based on the attributes *height* and *width*. By combining attributes, attribute construction can discover missing information about the relationships between data attributes that can be useful for knowledge discovery.

2.5 Data Reduction

Imagine that you have selected data from the *AllElectronics* data warehouse for analysis. The data set will likely be huge! Complex data analysis and mining on huge amounts of data can take a long time, making such analysis impractical or infeasible.

⁵In the machine learning literature, attribute construction is known as *feature construction*.