# Chapter 8

# Outlier Analysis

"*You are unique, and if that is not fulfilled, then something has been lost.*"—Martha Graham

## 8.1 Introduction

An outlier is a data point that is very different from most of the remaining data. Hawkins formally defined the notion of an outlier as follows:

"*An outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism.*"

Outliers can be viewed as a complementary concept to that of clusters. While clustering attempts to determine groups of data points that are similar, outliers are *individual* data points that are different from the remaining data. Outliers are also referred to as *abnormalities*, *discordants*, *deviants*, or *anomalies* in the data mining and statistics literature. Outliers have numerous applications in many data mining scenarios:

1. *Data cleaning:* Outliers often represent noise in the data. This noise may arise as a result of errors in the data collection process. Outlier detection methods are, therefore, useful for removing such noise.

2. *Credit card fraud:* Unusual patterns of credit card activity may often be a result of fraud. Because such patterns are much rarer than the normal patterns, they can be detected as outliers.

3. *Network intrusion detection:* The traffic on many networks can be considered as a stream of multidimensional records. Outliers are often defined as unusual records in this stream or unusual changes in the underlying trends.

Most outlier detection methods create a model of normal patterns. Examples of such models include clustering, distance-based quantification, or dimensionality reduction. Outliers are defined as data points that do not naturally fit within this normal model. The "outlierness" of a data point is quantified by a numeric value, known as the outlier score. Consequently, most outlier detection algorithms produce an output that can be one of two types:

1. *Real-valued outlier score:* Such a score quantifies the tendency for a data point to be considered an outlier. Higher values of the score make it more (or, in some cases, less) likely that a given data point is an outlier. Some algorithms may even output a probability value quantifying the likelihood that a given data point is an outlier.

2. *Binary label:* A binary value is output, indicating whether or not a data point is an outlier. This type of output contains less information than the first one because a threshold can be imposed on the outlier scores to convert them into binary labels. However, the reverse is not possible. Therefore, outlier scores are more general than binary labels. Nevertheless, a binary score is required as the end result in most applications because it provides a crisp decision.

The generation of an outlier score requires the construction of a model of the normal patterns. In some cases, a model may be designed to produce *specialized* types of outliers based on a very restrictive model of normal patterns. Examples of such outliers are extreme values, and they are useful only for certain specific types of applications. In the following, some of the key models for outlier analysis are summarized. These will be discussed in more detail in later sections.

1. *Extreme values:* A data point is an extreme value, if it lies at one of the two ends of a probability distribution. Extreme values can also be defined equivalently for multidimensional data by using a multivariate probability distribution, instead of a univariate one. These are very *specialized* types of outliers but are useful in general outlier analysis because of their utility in converting scores to labels.

2. *Clustering models:* Clustering is considered a complementary problem to outlier analysis. The former problem looks for data points that occur together in a group, whereas the latter problem looks for data points that are isolated from groups. In fact, many clustering models determine outliers as a side-product of the algorithm. It is also possible to optimize clustering models to specifically detect outliers.

3. *Distance-based models:* In these cases, the $k$-nearest neighbor distribution of a data point is analyzed to determine whether it is an outlier. Intuitively, a data point is an outlier, if its $k$-nearest neighbor distance is much larger than that of other data points. Distance-based models can be considered a more fine-grained and instance-centered version of clustering models.

4. *Density-based models:* In these models, the local density of a data point is used to define its outlier score. Density-based models are intimately connected to distance-based models because the local density at a given data point is low only when its distance to its nearest neighbors is large.

5. *Probabilistic models:* Probabilistic algorithms for clustering are discussed in Chap. 6. Because outlier analysis can be considered a complementary problem to clustering, it is natural to use probabilistic models for outlier analysis as well. The steps are almost analogous to those of clustering algorithms, except that the EM algorithm is used for

clustering, and the probabilistic fit values are used to quantify the outlier scores of data points (instead of distance values).

6. *Information-theoretic models:* These models share an interesting relationship with other models. Most of the other methods fix the model of normal patterns and then quantify outliers in terms of deviations from the model. On the other hand, information-theoretic methods constrain the maximum deviation allowed from the normal model and then examine the difference in space requirements for constructing a model with or without a specific data point. If the difference is large, then this point is reported as an outlier.

In the following sections, these different types of models will be discussed in detail. Representative algorithms from each of these classes of algorithms will also be introduced.

It should be pointed out that this chapter defines outlier analysis as an *unsupervised problem* in which previous examples of anomalies and normal data points are not available. The supervised scenario, in which examples of previous anomalies are available, is a special case of the classification problem. That case will be discussed in detail in Chap. 11.

This chapter is organized as follows. Section 8.2 discusses methods for extreme value analysis. Probabilistic methods are introduced in Sect. 8.3. These can be viewed as modifications of EM-clustering methods that leverage the connections between the clustering and outlier analysis problem for detecting outliers. This issue is discussed more formally in Sect. 8.4. Distance-based models for outlier detection are discussed in Sect. 8.5. Density-based models are discussed in Sect. 8.6. Information-theoretic models are addressed in Sect. 8.7. The problem of cluster validity is discussed in Sect. 8.8. A summary is given in Sect. 8.9.

## 8.2 Extreme Value Analysis

Extreme value analysis is a very specific kind of outlier analysis where the data points at the outskirts of the data are reported as outliers. Such outliers correspond to the *statistical tails* of probability distributions. Statistical tails are more naturally defined for 1-dimensional distributions, although an analogous concept can be defined for the multidimensional case.

It is important to understand that extreme values are very specialized types of outliers; in other words, all extreme values are outliers, but the reverse may not be true. The traditional definition of outliers is based on Hawkins's definition of generative probabilities. For example, consider the 1-dimensional data set corresponding to $\{1, 3, 3, 3, 50, 97, 97, 97, 100\}$. Here, the values 1 and 100 may be considered extreme values. The value 50 is the mean of the data set and is therefore not an extreme value. However, it is the most *isolated* point in the data set and should, therefore, be considered an outlier from a generative perspective.

A similar argument applies to the case of multivariate data where the extreme values lie in the *multivariate tail area* of the distribution. It is more challenging to formally define the concept of multivariate tails, although the basic concept is analogous to that of univariate tails. Consider the example illustrated in Fig. 8.1. Here, data point A may be considered an extreme value, and also an outlier. However, data point B is also isolated, and should, therefore, be considered an outlier. However, it cannot be considered a multivariate extreme value.

Extreme value analysis has important applications in its own right, and, therefore, plays an integral role in outlier analysis. An example of an important application of extreme value analysis is that of converting outlier scores to binary labels by identifying those outlier scores that are extreme values. Multivariate extreme value analysis is often useful in multicriteria

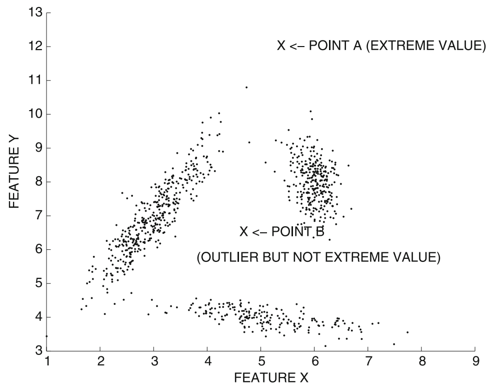


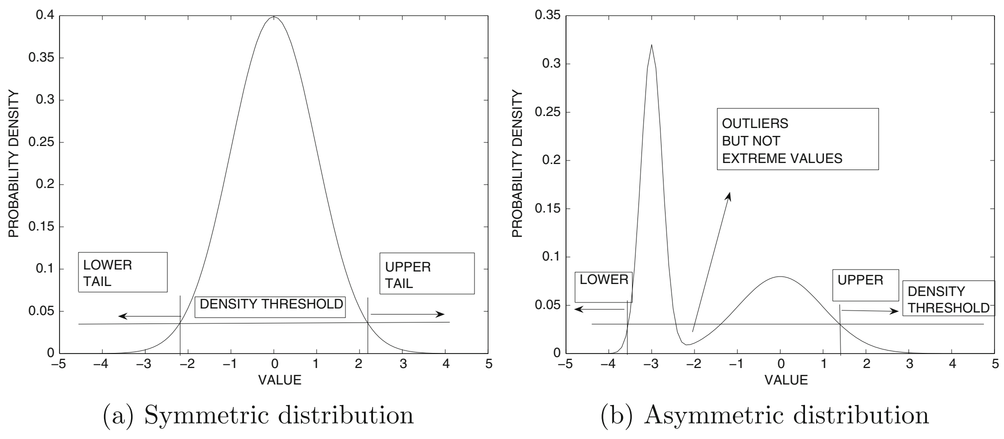
Figure 8.1: Multivariate extreme values

outlier-detection algorithms where it can be utilized to unify multiple outlier scores into a single value, and also generate a binary label as the output. For example, consider a meteorological application where outlier scores of spatial regions have been generated on the basis of analyzing their temperature and pressure variables independently. These evidences need to be unified into a single outlier score for the spatial region, or a binary label. Multivariate extreme value analysis is very useful in these scenarios. In the following discussion, methods for univariate and multivariate extreme value analysis will be discussed.

## 8.2.1 Univariate Extreme Value Analysis

Univariate extreme value analysis is intimately related to the notion of *statistical tail confidence tests*. Typically, statistical tail confidence tests assume that the 1-dimensional data are described by a specific distribution. These methods attempt to determine the fraction of the objects expected to be more extreme than the data point, based on these distribution assumptions. This quantification provides a level of confidence about whether or not a specific data point is an extreme value.

How is the "tail" of a distribution defined? For distributions that are not symmetric, it is often meaningful to talk about an upper tail and a lower tail, which may not have the same probability. The upper tail is defined as all extreme values larger than a particular threshold, and the lower tail is defined as all extreme values lower than a particular threshold. Consider the density distribution $f_X(x)$. In general, the tail may be defined as the two extreme regions of the distribution for which $f_X(x) \leq \theta$, for some user defined threshold $\theta$. Examples of the lower tail and the upper tail for symmetric and asymmetric distributions are illustrated in Fig. 8.2a and b, respectively. As evident from Fig. 8.2b, the area in the upper tail and the lower tail of an asymmetric distribution may not be the same. Furthermore, some regions in the interior of the distribution of Fig. 8.2b have density below the density threshold $\theta$, but are not extreme values because they do not lie in the tail of the distribution. The data points in this region may be considered outliers, but not extreme values. The areas inside the upper tail or lower tail in Fig. 8.2a and b represent the cumulative probability of these extreme regions. In symmetric probability distributions, the tail is defined in terms of this area, rather than a density threshold. However, the concept of density threshold is the defining characteristic of the tail, especially in the case of asymmetric univariate or multivariate



(a) Symmetric distribution (b) Asymmetric distribution

Figure 8.2: Tails of a symmetric and asymmetric distribution

distributions. Some asymmetric distributions, such as an exponential distribution, may not even have a tail at one end of the distribution.

A *model* distribution is selected for quantifying the tail probability. The most commonly used model is the normal distribution. The density function $f_X(x)$ of the normal distribution with mean $\mu$ and standard deviation $\sigma$ is defined as follows:

$$f_X(x) = \frac{1}{\sigma \cdot \sqrt{2 \cdot \pi}} \cdot e^{\frac{-(x-\mu)^2}{2 \cdot \sigma^2}}. \tag{8.1}$$

A *standard* normal distribution is one in which the mean is 0, and the standard deviation $\sigma$ is 1. In some application scenarios, the mean $\mu$ and standard deviation $\sigma$ of the distribution may be known through prior domain knowledge. Alternatively, when a *large* number of data samples is available, the mean and standard deviation may be estimated very accurately. These can be used to compute the $Z$-value for a random variable. The $Z$-number $z_i$ of an observed value $x_i$ can be computed as follows:

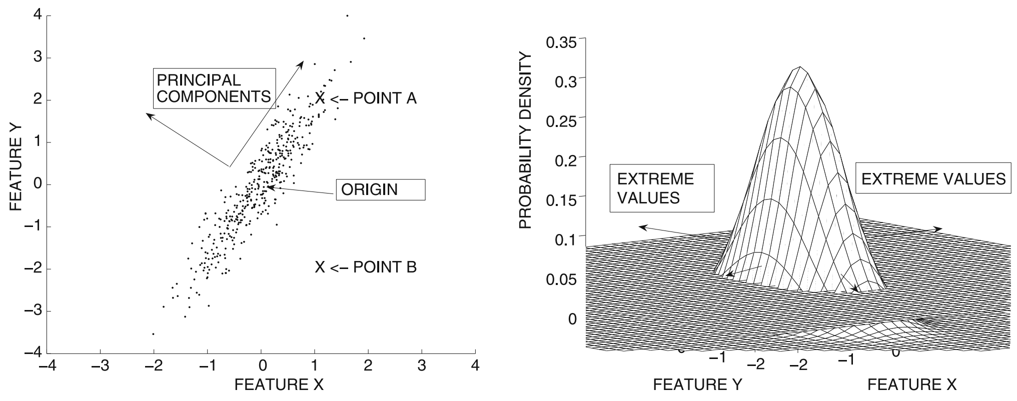
$$z_i = (x_i - \mu)/\sigma. \tag{8.2}$$

Large positive values of $z_i$ correspond to the upper tail, whereas large negative values correspond to the lower tail. The normal distribution can be expressed directly in terms of the $Z$-number because it corresponds to a scaled and translated random variable with a mean 0 and standard deviation of 1. The normal distribution of Eq. 8.3 can be written directly in terms of the $Z$-number, with the use of a *standard* normal distribution as follows:

$$f_X(z_i) = \frac{1}{\sigma \cdot \sqrt{2 \cdot \pi}} \cdot e^{\frac{-z_i^2}{2}}. \tag{8.3}$$

This implies that the cumulative normal distribution may be used to determine the area of the tail that is larger than $z_i$. As a rule of thumb, if the absolute values of the $Z$-number are greater than 3, the corresponding data points are considered extreme values. At this threshold, the cumulative area inside the tail can be shown to be less than 0.01 % for the normal distribution.

When a smaller number $n$ of data samples is available for estimating the mean $\mu$ and standard deviations $\sigma$, the aforementioned methodology can be used with a minor modification. The value of $z_i$ is computed as before, and the student $t$-distribution with $n$ degrees



(a) Multivariate extreme values

(b) Multivariate extreme values (probabilistic interpretation)

Figure 8.3: Multivariate extreme values

of freedom is used to quantify the cumulative distribution in the tail instead of the normal distribution. Note that, when $n$ is large, the $t$-distribution converges to the normal distribution.

## 8.2.2 Multivariate Extreme Values

Strictly speaking, tails are defined for univariate distributions. However, just as the univariate tails are defined as extreme regions with probability density less than a particular threshold, an analogous concept can also be defined for multivariate distributions. The concept is more complex than the univariate case and is defined for unimodal probability distributions with a single peak. As in the previous case, a multivariate Gaussian model is used, and the corresponding parameters are estimated in a data-driven manner. The implicit modeling assumption of multivariate extreme value analysis is that all data points are located in a probability distribution with a single peak (i.e., single Gaussian cluster), and data points in all directions that are as far away as possible from the center of the cluster should be considered extreme values.

Let $\overline{\mu}$ be the $d$-dimensional mean vector of a $d$-dimensional data set, and $\Sigma$ be its $d \times d$ covariance matrix. Thus, the $(i, j)$th entry of the covariance matrix is equal to the covariance between the dimensions $i$ and $j$. These represent the estimated parameters of the multivariate Gaussian distribution. Then, the probability distribution $f(\overline{X})$ for a $d$-dimensional data point $\overline{X}$ can be defined as follows:

$$f(\overline{X}) = \frac{1}{\sqrt{|\Sigma|} \cdot (2 \cdot \pi)^{(d/2)}} \cdot e^{-\frac{1}{2} \cdot (\overline{X}-\overline{\mu})\Sigma^{-1}(\overline{X}-\overline{\mu})^T}. \tag{8.4}$$

The value of $|\Sigma|$ denotes the determinant of the covariance matrix. The term in the exponent is half the square of the *Mahalanobis distance* between data point $\overline{X}$ and the mean $\overline{\mu}$ of the data. In other words, if $Maha(\overline{X}, \overline{\mu}, \Sigma)$ represents the Mahalanobis distance between $\overline{X}$ and $\overline{\mu}$, with respect to the covariance matrix $\Sigma$, then the probability density function of the normal distribution is as follows:

$$f(\overline{X}) = \frac{1}{\sqrt{|\Sigma|} \cdot (2 \cdot \pi)^{(d/2)}} \cdot e^{-\frac{1}{2} \cdot Maha(\overline{X}, \overline{\mu}, \Sigma)^2}. \tag{8.5}$$

For the probability density to fall below a particular threshold, the Mahalanobis distance needs to be *larger* than a particular threshold. Thus, the Mahalanobis distance to the mean of the data can be used as an extreme-value score. The relevant extreme values are defined by the multidimensional region of the data for which the Mahalanobis distance to the mean is larger than a particular threshold. This region is illustrated in Fig. 8.3b. Therefore, the extreme value score for a data point can be reported as the Mahalanobis distance between that data point and the mean. Larger values imply more extreme behavior. In some cases, one might want a more intuitive probability measure. Correspondingly, the extreme value *probability* of a data point $\overline{X}$ is defined by the cumulative probability of the multidimensional region for which the Mahalanobis distance to the mean $\overline{\mu}$ of the data is greater than that between $\overline{X}$ and $\overline{\mu}$. How can one estimate this cumulative probability?

As discussed in Chap. 3, the Mahalanobis distance is similar to the Euclidean distance except that it standardizes the data along uncorrelated directions. For example, if the axis system of the data were to be rotated to the principal directions (shown in Fig. 8.3), then the transformed coordinates in this new axis system would have no interattribute correlations (i.e., a diagonal covariance matrix). The Mahalanobis distance is simply equal to the Euclidean distance in such a transformed (axes-rotated) data set *after* dividing each of the transformed coordinate values by the standard deviation along its direction. This approach provides a neat way to model the probability distribution of the Mahalanobis distance, and it also provides a concrete estimate of the cumulative probability in the multivariate tail.

Because of the scaling by the standard deviation, each of the independent components of the Mahalanobis distances along the principal correlation directions can be modeled as a 1-dimensional *standard* normal distribution with mean 0 and variance 1. The sum of the squares of $d$ variables, drawn independently from standard normal distributions, will result in a variable drawn from an $\chi^2$ distribution with $d$ degrees of freedom. Therefore, the cumulative probability of the region of the $\chi^2$ distribution with $d$ degrees of freedom, for which the value is greater than $Maha(\overline{X}, \overline{\mu}, \Sigma)$, can be reported as the extreme value probability of $\overline{X}$. Smaller values of the probability imply greater likelihood of being an extreme value.

Intuitively, this approach models the data distribution along the various uncorrelated directions as statistically independent normal distributions and standardizes them so as to provide each such direction equal importance in the outlier score. In Fig. 8.3a, data point B can be more reasonably considered a multivariate extreme value than data point A, on the basis of the natural correlations in the data. On the other hand, the data point B is closer to the centroid of the data (than data point A) on the basis of Euclidean distance but not on the basis of the Mahalanobis distance. This shows the utility of the Mahalanobis distance in using the underlying statistical distribution of the data to infer the outlier behavior of the data points more effectively.

### 8.2.3 Depth-Based Methods

Depth-based methods are based on the general principle that the convex hull of a set of data points represents the pareto-optimal extremes of this set. A depth-based algorithm proceeds in an iterative fashion, where during the $k$-th iteration, all points at the corners of the convex hull of the data set are removed. The index of the iteration $k$ also provides an outlier score where smaller values indicate a greater tendency for a data point to be an outlier. These steps are repeated until the data set is empty. The outlier score may be converted to a binary label by reporting all data points with depth at most $r$ as outliers.

```
Algorithm FindDepthOutliers(Data Set: $\mathcal{D}$, Score Threshold: $r$)
begin
  $k = 1$;
  repeat
    Find set $S$ of corners of convex hull of $\mathcal{D}$;
    Assign depth $k$ to points in $S$;
    $\mathcal{D} = \mathcal{D} - S$;
    $k = k + 1$;
  until($D$ is empty);
  Report points with depth at most $r$ as outliers;
end
```

Figure 8.4: Depth-based methods

The value of $r$ may itself need to be determined by univariate extreme value analysis. The steps of the depth-based approach are illustrated in Fig. 8.4.

A pictorial illustration of the depth-based method is illustrated in Fig. 8.5. The process can be viewed as analogous to peeling the different layers of an onion (as shown in Fig. 8.5b) where the outermost layers define the outliers. Depth-based methods try to achieve the same goal as the multivariate method of the previous section, but it generally tends to be less effective both in terms of quality and computational efficiency. From a qualitative perspective, depth-based methods do not normalize for the characteristics of the statistical data distribution, as is the case for multivariate methods based on the Mahalanobis distance. All data points at the corners of a convex hull are treated equally. This is clearly not desirable, and the scores of many data points are indistinguishable because of ties. Furthermore, the fraction of data points at the corners of the convex hull generally increases with dimensionality. For very high dimensionality, it may not be uncommon for the majority of the data points to be located at the corners of the outermost convex hull. As a result, it is no longer possible to distinguish the outlier scores of different data points. The computational complexity of convex-hull methods increases significantly with dimensionality. The combination of qualitative and computational issues associated with this method make it a poor alternative to the multivariate methods based on the Mahalanobis distance.

## 8.3 Probabilistic Models

Probabilistic models are based on a generalization of the multivariate extreme values analysis methods discussed in Sect. 8.2.2. The Mahalanobis distance-based multivariate extreme value analysis method can be viewed as a Gaussian mixture model with a *single* component in the mixture. By generalizing this model to multiple mixture components, it is possible to determine general outliers, rather than multivariate extreme values. This idea is intimately related to the EM-clustering algorithm discussed in Sect. 6.5 of Chap. 6. At an intuitive level, data points that do not naturally fit any cluster in the probabilistic sense may be reported as outliers. The reader is referred to Sect. 6.5 of Chap. 6 for a more detailed discussion of the EM algorithm, though a brief outline is provided here for convenience.

The broad principle of a mixture-based generative model is to assume that the data were generated from a mixture of $k$ distributions with the probability distributions $\mathcal{G}_1 \dots \mathcal{G}_k$ based on the following process:

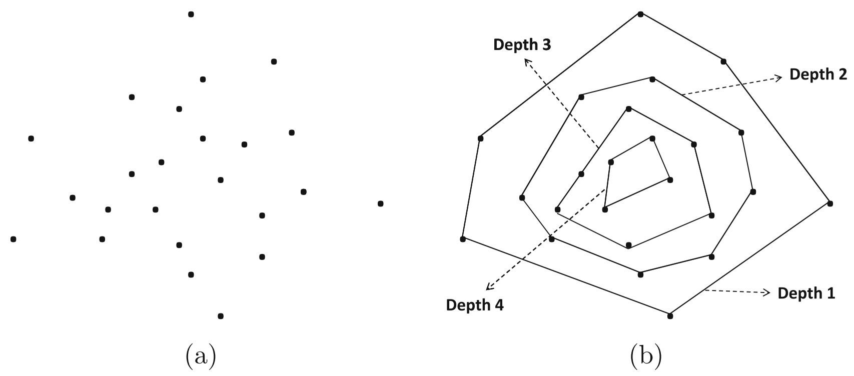


Figure 8.5: Depth-based outlier detection

1. Select a mixture component with prior probability $\alpha_i$, where $i \in \{1 \ldots k\}$. Assume that the $r$th one is selected.
2. Generate a data point from $\mathcal{G}_r$.

This generative model will be denoted by $\mathcal{M}$, and it generates each point in the data set $\mathcal{D}$. The data set $\mathcal{D}$ is used to estimate the parameters of the model. Although it is natural to use Gaussians to represent each component of the mixture, other models may be used if desired. This flexibility is very useful to apply the approach to different data types. For example, in a categorical data set, a categorical probability distribution may be used for each mixture component instead of the Gaussian distribution. After the parameters of the model have been estimated, outliers are defined as those data points in $\mathcal{D}$ that are highly unlikely to be generated by this model. Note that such an assumption exactly reflects Hawkins's definition of outliers, as stated at the beginning of this chapter.

Next, we discuss the estimation of the various parameters of the model such as the estimation of different values of $\alpha_i$ and the parameters of the different distributions $\mathcal{G}_r$. The objective function of this estimation process is to ensure that the full data $\mathcal{D}$ has the maximum likelihood fit to the generative model. Assume that the density function of $\mathcal{G}_i$ is given by $f^i(\cdot)$. The probability (density function) of the data point $\overline{X_j}$ being generated by the model is given by the following:

$$f^{point}(\overline{X_j}|\mathcal{M}) = \sum_{i=1}^{k} \alpha_i \cdot f^i(\overline{X_j}). \tag{8.6}$$

Note that the density value $f^{point}(\overline{X_j}|\mathcal{M})$ provides an estimate of the outlier score of the data point. Data points that are outliers will naturally have low fit values. Examples of the relationship of the fit values to the outlier scores are illustrated in Fig. 8.6. Data points A and B will typically have very low fit to the mixture model and will be considered outliers because the data points A and B do not naturally belong to any of the mixture components. Data point C will have high fit to the mixture model and will, therefore, not be considered an outlier. The parameters of the model $\mathcal{M}$ are estimated using a maximum likelihood criterion, which is discussed below.

For data set $\mathcal{D}$ containing $n$ data points, denoted by $\overline{X_1} \ldots \overline{X_n}$, the probability density of the data set being generated by model $\mathcal{M}$ is the product of the various point-specific

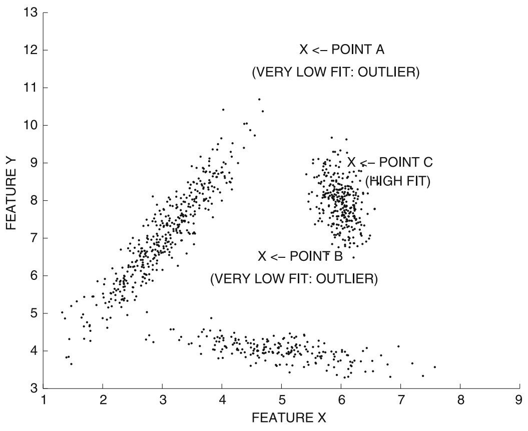


Figure 8.6: Likelihood fit values versus outlier scores

probability densities:

$$f^{data}(\mathcal{D}|\mathcal{M}) = \prod_{j=1}^{n} f^{point}(\overline{X_j}|\mathcal{M}). \tag{8.7}$$

The log-likelihood fit $\mathcal{L}(\mathcal{D}|\mathcal{M})$ of the data set $\mathcal{D}$ with respect to $\mathcal{M}$ is the logarithm of the aforementioned expression, and can be (more conveniently) represented as a sum of values over the different data points:

$$\mathcal{L}(\mathcal{D}|\mathcal{M}) = \log(\prod_{j=1}^{n} f^{point}(\overline{X_j}|\mathcal{M})) = \sum_{j=1}^{n} \log(\sum_{i=1}^{k} \alpha_i \cdot f^i(\overline{X_j})). \tag{8.8}$$

This log-likelihood fit needs to be optimized to determine the model parameters. This objective function maximizes the fit of the data points to the generative model. For this purpose, the EM algorithm discussed in Sect. 6.5 of Chap. 6 is used.

After the parameters of the model have been determined, the value of $f^{point}(\overline{X_j}|\mathcal{M})$ (or its logarithm) may be reported as the outlier score. The major advantage of such mixture models is that the mixture components can also incorporate domain knowledge about the shape of each individual mixture component. For example, if it is known that the data points in a particular cluster are correlated in a certain way, then this fact can be incorporated in the mixture model by fixing the appropriate parameters of the covariance matrix, and learning the remaining parameters. On the other hand, when the available data is limited, mixture models may overfit the data. This will cause data points that are truly outliers to be missed.

## 8.4 Clustering for Outlier Detection

The probabilistic algorithm of the previous section provides a preview of the relationship between clustering and outlier detection. Clustering is all about finding “crowds” of data points, whereas outlier analysis is all about finding data points that are far away from these crowds. Clustering and outlier detection, therefore, share a well-known complementary relationship. A simplistic view is that every data point is either a member of a cluster or an outlier. Clustering algorithms often have an “outlier handling” option that removes data

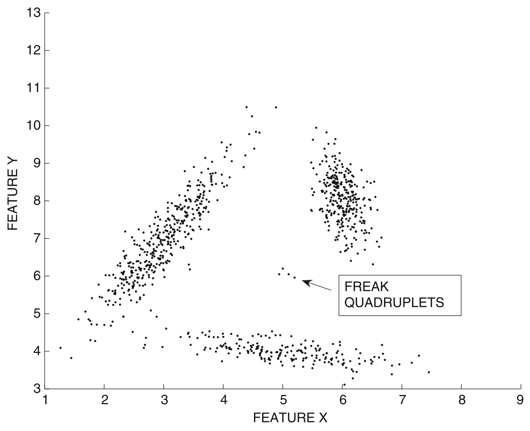


Figure 8.7: Small isolated groups of anomalies

points outside the clusters. The detection of outliers as a side-product of clustering methods is, however, not an appropriate approach because clustering algorithms are not optimized for outlier detection. Data points on the boundary regions of a cluster may also be considered weak outliers but are rarely useful in most application-specific scenarios.

Clustering models do have some advantages as well. Outliers often tend to occur in small clusters of their own. This is because the anomaly in the generating process may be repeated a few times. As a result, a small group of related outliers may be created. An example of a small set of isolated outliers is illustrated in Fig. 8.7. As will be discussed later, clustering methods are generally robust to such scenarios because such groups often do not have the critical mass required to form clusters of their own.

A simple way of defining the outlier score of a data point is to first cluster the data set and then use the raw distance of the data point to its closest cluster centroid. One can, however, do better when the clusters are elongated or have varying density over the data set. As discussed in Chap. 3, the local data distribution often distorts the distances, and, therefore, it is not optimal to use the raw distance. This broader principle is used in multivariate extreme value analysis where the *global* Mahalanobis distance defines outlier scores. In this case, the *local* Mahalanobis distance can be used with respect to the centroid of the closest cluster.

Consider a data set in which $k$ clusters have been discovered with the use of a clustering algorithm. Assume that the $r$th cluster in $d$-dimensional space has a corresponding $d$-dimensional mean vector $\overline{\mu_r}$, and a $d \times d$ covariance matrix $\Sigma_r$. The $(i, j)$th entry of this matrix is the covariance between the dimensions $i$ and $j$ in that cluster. Then, the Mahalanobis distance $Maha(\overline{X}, \overline{\mu_r}, \Sigma_r)$ between a data point $\overline{X}$ and cluster centroid $\overline{\mu_r}$ is defined as follows:

$$Maha(\overline{X}, \overline{\mu_r}, \Sigma_r) = \sqrt{(\overline{X} - \overline{\mu_r})\Sigma_r^{-1}(\overline{X} - \overline{\mu_r})^T}. \tag{8.9}$$

This distance is reported as the outlier score. Larger values of the outlier score indicate a greater outlier tendency. After the outlier score has been determined, univariate extreme value analysis may be used to convert the scores to binary labels.

The justification for using the Mahalanobis distance is exactly analogous to the case of extreme value analysis of multivariate distances, as discussed in Sect. 8.2. The only difference is that the *local cluster-specific* Mahalanobis distances are more relevant to determination of general outliers, whereas global Mahalanobis distances are more relevant to determination of specific types of outliers, such as extreme values. The use of the local Mahalanobis distance also has an interesting connection to the likelihood fit criterion of EM algorithm where the (squared) Mahalanobis distance occurs in the exponent of each Gaussian mixture. Thus, the sum of the inverse exponentiated Mahalanobis distances of a data point to different mixture component means (cluster means) are used to determine the outlier score in the EM algorithm. Such a score can be viewed as a soft version of the score determined by hard clustering algorithms.

Clustering methods are based on global analysis. Therefore, small, closely related groups of data points will not form their own clusters in most cases. For example, the four isolated points in Fig. 8.7 will not typically be considered a cluster. Most clustering algorithms require a minimum critical mass for a set of data points to be considered a cluster. As a result, these points will have a high outlier score. This means that clustering methods are able to detect these small and closely related groups of data points meaningfully and report them as outliers. This is not the case for some of the density-based methods that are based purely on local analysis.

The major problem with clustering algorithms is that they are sometimes not able to properly distinguish between a data point that is *ambient noise* and a data point that is a *truly isolated anomaly*. Clearly, the latter is a much stronger anomaly than the former. Both these types of points will not reside in a cluster. Therefore, the distance to the closest cluster centroid will often not be very representative of their local distribution (or *instance-specific* distribution). In these cases, distance-based methods are more effective.

## 8.5 Distance-Based Outlier Detection

Because outliers are defined as data points that are far away from the "crowded regions" (or clusters) in the data, a natural and *instance-specific* way of defining an outlier is as follows:

*The distance-based outlier score of an object $O$ is its distance to its $k$th nearest neighbor.*

The aforementioned definition, which uses the $k$-nearest neighbor distance, is the most common one. Other variations of this definition are sometimes used, such as the average distance to the $k$-nearest neighbors. The value of $k$ is a user-defined parameter. Selecting a value of $k$ larger than 1 helps identify isolated groups of outliers. For example, in Fig. 8.7, as long as $k$ is fixed to any value larger than 3, all data points within the small groups of closely related points will have a high outlier score. Note that the target data point, for which the outlier score is computed, is itself not included among its $k$-nearest neighbors. This is done to avoid scenarios where a 1-nearest neighbor method will always yield an outlier score of 0.

Distance-based methods typically use a finer granularity of analysis than clustering methods and can therefore distinguish between ambient noise and truly isolated anomalies. This is because ambient noise will typically have a lower $k$-nearest neighbor distance than a truly isolated anomaly. This distinction is lost in clustering methods where the distance to the closest cluster centroid does not accurately reflect the *instance-specific* isolation of the underlying data point.

The price of this better granularity is higher computational complexity. Consider a data set $\mathcal{D}$ containing $n$ data points. The determination of the $k$-nearest neighbor distance requires $O(n)$ time *for each data point*, when a sequential scan is used. Therefore, the determination of the outlier scores of all data points may require $O(n^2)$ time. This is clearly not a feasible option for very large data sets. Therefore, a variety of methods are used to speed up the computation:

1. *Index structures:* Index structures can be used to determine $k$th nearest neighbor distances efficiently. This is, however, not an option, if the data is high dimensional. In such cases, the effectiveness of index structures tends to degrade.

2. *Pruning tricks:* In most applications, the outlier scores of all the data points are not required. It may suffice to return binary labels for the top-$r$ outliers, together with their scores. The outlier scores of the remaining data points are irrelevant. In such cases, it may be possible to terminate a $k$-nearest neighbor sequential scan for an outlier candidate when its current *upper bound estimate* on the $k$-nearest neighbor distance value falls below the $r$th best outlier score found so far. This is because such a candidate is guaranteed to be not among the top-$r$ outliers. This methodology is referred to as the "early termination trick," and it is described in detail later in this section.

In some cases, it is also possible to combine the pruning approach with index structures.

## 8.5.1 Pruning Methods

Pruning methods are used only for the case where the top-$r$ ranked outliers need to be returned, and the outlier scores of the remaining data points are irrelevant. Thus, pruning methods can be used only for the binary-decision version of the algorithm. The basic idea in pruning methods is to reduce the time required for the $k$-nearest neighbor distance computations by ruling out data points quickly that are obviously nonoutliers even with approximate computation.

### 8.5.1.1 Sampling Methods

The first step is to pick a sample $\mathcal{S}$ of size $s \ll n$ from the data $\mathcal{D}$, and compute all pairwise distances between the data points in sample $\mathcal{S}$ and those in database $\mathcal{D}$. There are a total of $n \cdot s$ such pairs. This process requires $O(n \cdot s) \ll O(n^2)$ distance computations. Thus, for each of the sampled points in $\mathcal{S}$, the $k$-nearest neighbor distance is already known exactly. The top $r$th ranked outlier in sample $\mathcal{S}$ is determined, where $r$ is the number of outliers to be returned. The score of the $r$th rank outlier provides a *lower* bound[1] $L$ on the $r$th ranked outlier score over the entire data set $\mathcal{D}$. For the data points in $\mathcal{D} - \mathcal{S}$, only an *upper bound* $V^k(\overline{X})$ on the $k$-nearest neighbor distance is known. This upper bound is equal to the $k$-nearest neighbor distance of each point in $\mathcal{D} - \mathcal{S}$ to the sample $\mathcal{S} \subset \mathcal{D}$. However, if this upper bound $V^k(\overline{X})$ is no larger than the lower bound $L$ already determined, then such a data point $\overline{X} \in \mathcal{D} - \mathcal{S}$ can be excluded from further consideration as a top-$r$ outlier. Typically, this will result in the removal of a large number of outlier candidates from $\mathcal{D} - \mathcal{S}$ immediately, as long as the underlying data set is clustered well. This is because most of the data points in clusters will be removed, as long as at least one point from each cluster is included in the sample $\mathcal{S}$, and at least $r$ points in $\mathcal{S}$ are located in somewhat sparse regions. This can often be achieved with modest values of the sample size $s$ in real-world data sets. After removing these data points from $\mathcal{D} - \mathcal{S}$, the remaining set of points is $\mathcal{R} \subseteq \mathcal{D} - \mathcal{S}$. The $k$-nearest neighbor approach can be applied to a much smaller set of candidates $\mathcal{R}$.

[1]Note that higher $k$-nearest neighbor distances indicate greater outlierness.

The top-$r$ ranked outliers in $\mathcal{R} \cup \mathcal{S}$ are returned as the final output. Depending on the level of pruning already achieved, this can result in a very significant reduction in computational time, especially when $|\mathcal{R} \cup \mathcal{S}| \ll |\mathcal{D}|$.

#### 8.5.1.2 Early Termination Trick with Nested Loops

The approach discussed in the previous section can be improved even further by speeding up the second phase of computing the $k$-nearest neighbor distances of each data point in $\mathcal{R}$. The idea is that the computation of the $k$-nearest neighbor distance of any data point $\overline{X} \in \mathcal{R}$ need not be followed through to termination once it has been determined that $\overline{X}$ cannot possibly be among the top-$r$ outliers. In such cases, the scan of the database $\mathcal{D}$ for computation of the $k$-nearest neighbor of $\overline{X}$ can be terminated early.

Note that one already has an estimate (upper bound) $V^k(\overline{X})$ of the $k$-nearest neighbor distance of every $\overline{X} \in \mathcal{R}$, based on distances to sample $\mathcal{S}$. Furthermore, the $k$-nearest neighbor distance of the $r$th best outlier in $\mathcal{S}$ provides a lower bound on the "cut-off" required to make it to the top-$r$ outliers. This lower-bound is denoted by $L$. This estimate $V^k(\overline{X})$ of the $k$-nearest neighbor distance of $\overline{X}$ is further tightened (reduced) as the database $\mathcal{D} - \mathcal{S}$ is scanned, and the distance of $\overline{X}$ is computed to each point in $\mathcal{D} - \mathcal{S}$. Because this running estimate $V^k(\overline{X})$ is always an upper bound on the true $k$-nearest neighbor distance of $\overline{X}$, the process of determining the $k$-nearest neighbor of $\overline{X}$ can be terminated as soon as $V^k(\overline{X})$ falls below the known lower bound $L$ on the top-$r$ outlier distance. This is referred to as *early termination* and provides significant computational savings. Then, the next data point in $\mathcal{R}$ can be processed. In cases where early termination is not achieved, the data point $\overline{X}$ will almost[2] always be among the top-$r$ (current) outliers. Therefore, in this case, the lower bound $L$ can be tightened (increased) as well, to the new $r$th best outlier score. This will result in even better pruning when the next data point from $\mathcal{R}$ is processed to determine its $k$-nearest neighbor distance value. To maximize the benefits of pruning, the data points in $\mathcal{R}$ should not be processed in arbitrary order. Rather, they should be processed in decreasing order of the initially sampled estimate $V^k(\cdot)$ of the $k$-nearest neighbor distances (based on $\mathcal{S}$). This ensures that the outliers in $\mathcal{R}$ are found early on, and the global bound $L$ is tightened as fast as possible for even better pruning. Furthermore, in the inner loop, the data points $\overline{Y}$ in $\mathcal{D} - \mathcal{S}$ can be ordered in the opposite direction, based on *increasing* value of $V^k(\overline{Y})$. Doing so ensures that the $k$-nearest neighbor distances are updated as fast as possible, and the advantage of early termination is maximized. The nested loop approach can also be implemented without the first phase[3] of sampling, but such an approach will not have the advantage of proper ordering of the data points processed. Starting with an initial lower bound $L$ on the $r$th best outlier score obtained from the sampling phase, the nested loop is executed as follows:

[2]We say "almost," because the very last distance computation for $\overline{X}$ may bring $V(\overline{X})$ below $L$. This scenario is unusual, but might occasionally occur.

[3]Most descriptions in the literature omit the first phase of sampling, which is very important for efficiency maximization. A number of implementations in time-series analysis [306] do order the data points more carefully but not with sampling.

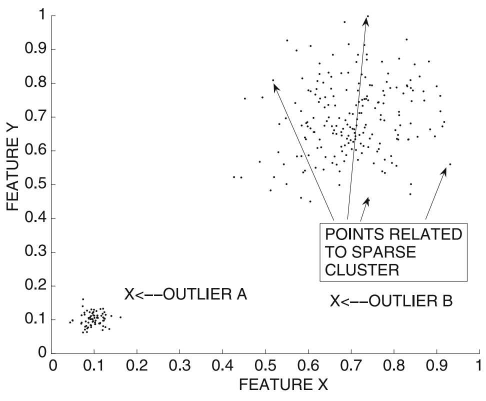
```
for each $\overline{X} \in \mathcal{R}$ do begin
  for each $\overline{Y} \in \mathcal{D} - \mathcal{S}$ do begin
    Update current $k$-nearest neighbor distance estimate $V^k(\overline{X})$ by
      computing distance of $\overline{Y}$ to $\overline{X}$;
    if $V^k(\overline{X}) \leq L$ then terminate inner loop;
  endfor
  if $V^k(\overline{X}) > L$ then
    include $\overline{X}$ in current $r$ best outliers and update $L$ to
      the new $r$th best outlier score;
endfor
```

Note that the $k$-nearest neighbors of a data point $\overline{X}$ do not include the data point itself. Therefore, care must be taken in the nested loop structure to ignore the trivial cases where $\overline{X} = \overline{Y}$ while updating $k$-nearest neighbor distances.

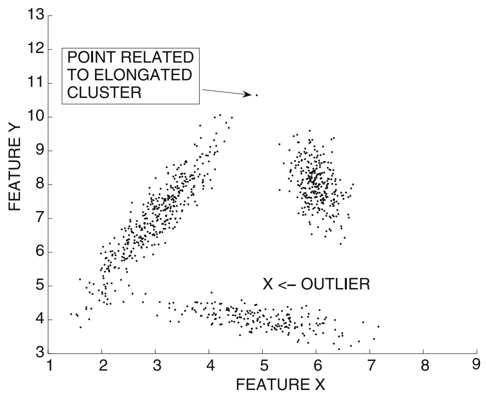
## 8.5.2 Local Distance Correction Methods

Section 3.2.1.8 of Chap. 3 provides a detailed discussion of the impact of the local data distribution on distance computation. In particular, it is shown that straightforward measures, such as the Euclidean distance, do not reflect the *intrinsic* distances between data points when the density and shape of the clusters vary significantly with data locality. This principle was also used in Sect. 8.4 to justify the use of the local Mahalanobis distance for measuring the distances to cluster centroids, rather than the Euclidean distance. One of the earliest methods that recognized this principle in the context of varying data density was the *Local Outlier Factor (LOF)* method. A formal justification is based on the generative principles of data sets, but only an intuitive understanding will be provided here. It should be pointed out that the use of the Mahalanobis distance (instead of the Euclidean distance) for multivariate extreme value analysis (Sect. 8.2.2) is also based on generative principles of the *likelihood* of a data point conforming to the statistical properties of the underlying distribution. The main difference is that the analysis was *global* in that case, whereas the analysis is *local* in this case. The reader is also advised to revisit Sect. 3.2.1.8 of Chap. 3 for a discussion of the impact of data distributions on *intrinsic* distances between data points.

To motivate the principles of local distance correction in the context of outlier analysis, two examples will be used. One of these examples illustrates the impact of varying local distribution density, whereas another example illustrates the impact of varying local cluster shape. Both these aspects can be addressed with different kinds of local normalization of distance computations. In Fig. 8.8a, two different clusters have been shown, one of which is much sparser than the other. In this case, both data points A and B are clearly outliers. While the outlier B will be easily detected by most distance-based algorithms, a challenge arises in the detection of outlier A. This is because the nearest neighbor distance of many data points in the sparser cluster is at least as large as the nearest neighbor distance of outlier A. As a result, depending on the distance-threshold used, a $k$-nearest neighbor algorithm will either falsely report portions of the sparse cluster, or will completely miss outlier A. Simply speaking, the *ranking* of the outliers by distance-based algorithms is an incorrect one. This is because the true distance of points in cluster A should be computed in a *normalized* way, based on its *local* data distribution. This aspect is relevant to the discussion in Sect. 3.2.1.8 of Chap. 3 on the impact of local data distributions on distance function design, and it is important for many distance-based data mining problems. The key



(a) Varying cluster density



(b) Varying cluster shape

Figure 8.8: Impact of local variations in data distribution on distance-based outlier detection

issue here is the generative principle, that data point A is much *less* likely to be generated by its closest (tightly knit) cluster than many slightly isolated data points belonging to the relatively diffuse cluster are likely to be generated by their cluster. Hawkins's definition of outliers, stated at the beginning of this chapter, was formulated on the basis of generative principles. It should be pointed out that the probabilistic EM algorithm of Sect. 8.3 does a much better job at recognizing these generative differences. However, the probabilistic EM method is often not used practically because of overfitting issues with smaller data sets. The *LOF* approach is the first method that recognized the importance of incorporating these generative principles in nonparametric distance-based algorithms.

This point can be emphasized further by examining clusters of different local shape and orientation in Fig. 8.8b. In this case, a distance-based algorithm will report one of the data points along the long axis of one of the elongated clusters, as the strongest outlier, if the 1-nearest neighbor distance is used. This data point is far *more* likely to be generated by its closest cluster, than the outlier marked by "X." However, the latter has a smaller 1-nearest neighbor distance. Therefore, the significant problem with distance-based algorithms is that they do not account for the local generative behavior of the underlying data. In this section, two methods will be discussed for addressing this issue. One of them is *LOF*, and the other is a direct generalization of the global Mahalanobis method for extreme value analysis. The first method can adjust for the generative variations illustrated in Fig. 8.8a, and the second method can adjust for the generative variations illustrated in Fig. 8.8b.

#### 8.5.2.1 Local Outlier Factor (LOF)

The *Local Outlier Factor (LOF)* approach adjusts for local variations in cluster density by normalizing distances with the average point-specific distances in a data locality. It is often understood popularly as a density-based approach, although, in practice, it is a (normalized) distance-based approach where the normalization factor corresponds to the average local data density. This normalization is the key to addressing the challenges posed by the scenario of Fig. 8.8a.

For a given data point $\overline{X}$, let $V^k(\overline{X})$ be the distance to its $k$-nearest neighbor, and let $L_k(\overline{X})$ be the set of points within the $k$-nearest neighbor distance of $\overline{X}$. The set $L_k(\overline{X})$ will typically contain $k$ points, but may sometimes contain more than $k$ points because of ties in the $k$-nearest neighbor distance.

Then, the reachability distance $R_k(\overline{X}, \overline{Y})$ of object $\overline{X}$ with respect to $\overline{Y}$ is defined as the maximum of the distance $Dist(\overline{X}, \overline{Y})$, between the pair $(\overline{X}, \overline{Y})$ and the $k$-nearest neighbor distance of $\overline{Y}$.

$$R_k(\overline{X}, \overline{Y}) = \max\{Dist(\overline{X}, \overline{Y}), V^k(\overline{Y})\} \tag{8.10}$$

The reachability distance is not symmetric between $\overline{X}$ and $\overline{Y}$. Intuitively, when $\overline{Y}$ is in a dense region and the distance between $\overline{X}$ and $\overline{Y}$ is large, the reachability distance of $\overline{X}$ with respect to it is equal to the true distance $Dist(\overline{X}, \overline{Y})$. On the other hand, when the distances between $\overline{X}$ and $\overline{Y}$ are small, then the reachability distance is smoothed out by the $k$-nearest neighbor distance of $\overline{Y}$. The larger the value of $k$, the greater the smoothing. Correspondingly, the reachability distances with respect to different points will also become more similar. The reason for using this smoothing is that it makes the intermediate distance computations more stable. This is especially important when the distances between $\overline{X}$ and $\overline{Y}$ are small, and it will result in greater statistical fluctuations in the raw distances. At a conceptual level, it is possible to define a version of *LOF* directly in terms of raw distances, rather than reachability distances. However, such a version would be missing the stability provided by smoothing.

The *average reachability distance* $AR_k(\overline{X})$ of data point $\overline{X}$ with respect to its neighborhood $L_k(\overline{X})$ is defined as the average of its reachability distances to all objects in its neighborhood.

$$AR_k(\overline{X}) = \text{MEAN}_{\overline{Y} \in L_k(\overline{X})} R_k(\overline{X}, \overline{Y}) \tag{8.11}$$

Here, the MEAN function simply represents the mean value over the entire set $L_k(\overline{X})$. The *Local Outlier Factor* $LOF_k(\overline{X})$ is then equal to the mean ratio of $AR_k(\overline{X})$ to the corresponding values of all points in the $k$-neighborhood of $\overline{X}$.

$$LOF_k(\overline{X}) = \text{MEAN}_{\overline{Y} \in L_k(\overline{X})} \frac{AR_k(\overline{X})}{AR_k(\overline{Y})} \tag{8.12}$$

The use of distance ratios in the definition ensures that the local distance behavior is well accounted for in this definition. As a result, the *LOF* values for the objects in a cluster are often close to 1 when the data points in the cluster are homogeneously distributed. For example, in the case of Fig. 8.8a, the *LOF* values of data points in *both* clusters will be quite close to 1, even though the densities of the two clusters are different. On the other hand, the *LOF* values of *both* the outlying points will be much higher because they will be computed in terms of the ratios to the average neighbor reachability distances. In practice, the maximum value of $LOF_k(\overline{X})$ over a range of different values of $k$ is used as the outlier score to determine the best size of the neighborhood.

One observation about the *LOF* method is that while it is popularly understood in the literature as a density-based approach, it can be more simply understood as a *relative* distance-based approach with smoothing. The smoothing is really a refinement to make distance computations more stable. The basic *LOF* method will work reasonably well on many data sets, even if the raw distances are used instead of reachability distances, for the aforementioned computations of Eq. 8.11.

The *LOF* method, therefore, has the ability to adjust well to regions of varying density because of this relative normalization in the denominator of each term of Eq. 8.12. In the original presentation of the *LOF* algorithm (see bibliographic notes), the *LOF* is defined in terms of a density variable. The density variable is loosely defined as the inverse of the average of the smoothed reachability distances. This is, of course, not a precise definition of density. Density is traditionally defined in terms of the number of data points within a specified area or volume. This book provides exactly the same definition of *LOF* but presents it slightly differently by omitting the intermediate density variable. This is done both for simplicity, and for a definition of *LOF* directly in terms of (normalized) distances. The real connection of *LOF* to data density lies in its insightful ability to *adjust* to varying data density with the use of *relative distances*. Therefore, this book has classified this approach as a (normalized) distance-based method, rather than as a density-based method.

#### 8.5.2.2 Instance-Specific Mahalanobis Distance

The instance-specific Mahalanobis distance is designed for adjusting to varying *shapes* of the distributions in the locality of a particular data point, as illustrated in Fig. 8.8b. The Mahalanobis distance is directly related to shape of the data distribution, although it is traditionally used in the *global* sense. Of course, it is also possible to use the *local* Mahalanobis distance by using the covariance structure of the neighborhood of a data point.

The problem here is that the neighborhood of a data point is hard to define with the Euclidean distance when the shape of the neighborhood cluster is not spherical. For example, the use of the Euclidean distance to a data point is biased toward capturing the circular region around that point, rather than an elongated cluster. To address this issue, an *agglomerative* approach is used to determine the $k$-neighborhood $L_k(\overline{X})$ of a data point $\overline{X}$. First, data point $\overline{X}$ is added to $L_k(\overline{X})$. Then, data points are iteratively added to $L_k(\overline{X})$ that have the smallest distance to *their closest* point in $L_k(\overline{X})$. This approach can be viewed as a special case of single-linkage hierarchical clustering methods, where singleton points are merged with clusters. Single-linkage methods are well-known for creating clusters of arbitrary shape. Such an approach tends to "grow" the neighborhood with the same shape as the cluster. The mean $\overline{\mu_k(X)}$ and covariance matrix $\Sigma_k(\overline{X})$ of the neighborhood $L_k(\overline{X})$ are computed. Then, the *instance-specific Mahalanobis score* $LMaha_k(\overline{X})$ of a data point $\overline{X}$ provides its outlier score. This score is defined as the Mahalanobis distance of $\overline{X}$ to the mean $\overline{\mu_k(X)}$ of data points in $L_k(\overline{X})$.

$$LMaha_k(\overline{X}) = Maha(\overline{X}, \overline{\mu_k(X)}, \Sigma_k(\overline{X})) \tag{8.13}$$

The only difference between this computation and that of the global Mahalanobis distance for extreme value analysis is that the local neighborhood set $L_k(\overline{X})$ is used as the "relevant" data for comparison in the former. While the clustering approach of Sect. 8.4 does use a Mahalanobis metric on the local neighborhood, the computation is subtly different in this case. In the case of clustering-based outlier detection, a preprocessing approach *predefines* a limited number of clusters as the universe of possible neighborhoods. In this case, the neighborhood is constructed in an *instance-specific* way. Different points will have slightly different neighborhoods, and they may not neatly correspond to a predefined cluster. This additional granularity allows more refined analysis. At a conceptual level, this approach computes whether data point $\overline{X}$ can be regarded as an extreme value with respect to its local cluster. As in the case of the *LOF* method, the approach can be applied for different values of $k$, and the highest outlier score for each data point can be reported.

If this approach is applied to the example of Fig. 8.8b, the method will correctly determine the outlier because of the local Mahalanobis normalization with the appropriate (local) covariance matrix for each data point. No distance normalizations are necessary for varying data density (scenario of Fig. 8.8a) because the Mahalanobis distance already performs these local normalizations under the covers. Therefore, such a method can be used for the scenario of Fig. 8.8a as well. The reader is referred to the bibliographic notes for variations of *LOF* that use the concept of varying local cluster shapes with agglomerative neighborhood computation.

## 8.6 Density-Based Methods

Density-based methods are loosely based on similar principles as density-based clustering. The idea is to determine sparse regions in the underlying data in order to report outliers. Correspondingly, histogram-based, grid-based, or kernel density-based methods can be used. Histograms can be viewed as 1-dimensional special cases of grid-based methods. These methods have not, however, found significant popularity because of their difficulty in adjusting to variations of density in different data localities. The definition of density also becomes more challenging with increasing dimensionality. Nevertheless, these methods are used more frequently in the univariate case because of their natural probabilistic interpretation.

### 8.6.1 Histogram- and Grid-Based Techniques

Histograms are simple and easy to construct for univariate data, and are therefore used quite frequently in many application domains. In this case, the data is discretized into bins, and the frequency of each bin is estimated. Data points that lie in bins with very low frequency are reported as outliers. If a continuous outlier score is desired, then the number of *other* data points in the bin for data point $\overline{X}$ is reported as the outlier score for $\overline{X}$. Therefore, the count for a bin does not include the point itself in order to minimize overfitting for smaller bin widths or smaller number of data points. In other words, the outlier score for each data point is one less than its bin count.

In the context of multivariate data, a natural generalization is the use of a grid-structure. Each dimension is partitioned into $p$ equi-width ranges. As in the previous case, the number of points in a particular grid region is reported as the outlier score. Data points that have density less than $\tau$ in any particular grid region are reported as outliers. The appropriate value of $\tau$ can be determined by using univariate extreme value analysis.

The major challenge with histogram-based techniques is that it is often hard to determine the optimal histogram width well. Histograms that are too wide, or too narrow, will not model the frequency distribution well. These are similar issues to those encountered with the use of grid-structures for clustering. When the bins are too narrow, the normal data points falling in these bins will be declared outliers. On the other hand, when the bins are too wide, anomalous data points and high-density regions may be merged into a single bin. Therefore, such anomalous data points may not be declared outliers.

A second issue with the use of histogram techniques is that they are too local in nature, and often do not take the global characteristics of the data into account. For example, for the case of Fig. 8.7, a multivariate grid-based approach may not be able to classify an isolated group of data points as outliers, unless the resolution of the grid structure is calibrated carefully. This is because the density of the grid only depends on the data points inside it, and an isolated group of points may create an artificially dense grid cell when the granularity of representation is high. Furthermore, when the density distribution varies significantly with data locality, grid-based methods may find it difficult to normalize for local variations in density.

Finally, histogram methods do not work very well in high dimensionality because of the sparsity of the grid structure with increasing dimensionality, unless the outlier score is computed with respect to carefully chosen lower dimensional projections. For example, a $d$-dimensional space will contain at least $2^d$ grid-cells, and, therefore, the number of data points expected to populate each cell reduces exponentially with increasing dimensionality. These problems with grid-based methods are well known, and are also frequently encountered in the context of other data mining applications such as clustering.

### 8.6.2 Kernel Density Estimation

Kernel density estimation methods are similar to histogram techniques in terms of building density profiles, though the major difference is that a smoother version of the density profile is constructed. In kernel density estimation, a continuous estimate of the density is generated at a given point. The value of the density at a given point is estimated as the sum of the smoothed values of kernel functions $K_h(\cdot)$ associated with each point in the data set. Each kernel function is associated with a kernel width $h$ that determines the level of smoothing created by the function. The kernel estimation $f(\overline{X})$ based on $n$ data points of dimensionality $d$, and kernel function $K_h(\cdot)$ is defined as follows:

$$f(\overline{X}) = \frac{1}{n} \cdot \sum_{i=1}^{n} K_h(\overline{X} - \overline{X_i}). \tag{8.14}$$

Thus, each discrete point $\overline{X_i}$ in the data set is replaced by a continuous function $K_h(\cdot)$ that peaks at $\overline{X_i}$ and has a variance determined by the smoothing parameter $h$. An example of such a distribution is the Gaussian kernel with width $h$.

$$K_h(\overline{X} - \overline{X_i}) = \left(\frac{1}{\sqrt{2\pi} \cdot h}\right)^d \cdot e^{-||\overline{X}-\overline{X_i}||^2/(2h^2)} \tag{8.15}$$

The estimation error is defined by the kernel width $h$, which is chosen in a data-driven manner. It has been shown that for most smooth functions $K_h(\cdot)$, when the number of data points goes to infinity, the estimate asymptotically converges to the true density value, provided that the width $h$ is chosen appropriately. The density at each data point is computed without including the point itself in the density computation. The value of the density is reported as the outlier score. Low values of the density indicate greater tendency to be an outlier.

Density-based methods have similar challenges as histogram- and grid-based techniques. In particular, the use of a global kernel width $h$ to estimate density may not work very well in cases where there are wide variations in local density, such as those in Figs. 8.7 and 8.8. This is because of the myopic nature of density-based methods, in which the variations in the density distribution are not well accounted for. Nevertheless, kernel-density-based methods can be better generalized to data with local variations, especially if the bandwidth is chosen locally. As in the case of grid-based methods, these techniques are not very effective for higher dimensionality. The reason is that the accuracy of the density estimation approach degrades with increasing dimensionality.

## 8.7 Information-Theoretic Models

Outliers are data points that do not naturally fit the remaining data distribution. Therefore, if a data set were to be somehow compressed with the use of the "normal" patterns in the data distribution, the outliers would increase the minimum code length required to describe it. For example, consider the following two strings:

```
ABABABABABABABABABABABABABABABABAB
ABABACABABABABABABABABABABABABABAB
```

The second string is of the same length as the first and is different at only a single position containing the unique symbol C. The first string can be described concisely as "AB 17 times." However, the second string has a single position corresponding to the symbol C. Therefore, the second string can no longer be described as concisely. In other words, the presence of the symbol C somewhere in the string increases its *minimum description length.* It is also easy to see that this symbol corresponds to an outlier. Information-theoretic models are based on this general principle because they measure the increase in model size required to describe the data as concisely as possible.

Information-theoretic models can be viewed as almost equivalent to conventional deviation-based models, except that the outlier score is defined by the model size for a fixed deviation, rather than the deviation for a fixed model. In conventional models, outliers are always defined on the basis of a "summary" model of normal patterns. When a data point deviates significantly from the estimations of the summary model, then this deviation value is reported as the outlier score. Clearly, a trade-off exists between the size of the summary model and the level of deviation. For example, if a clustering model is used, then a larger number of cluster centroids (model size) will result in lowering the maximum deviation of any data point (including the outlier) from its nearest centroid. Therefore, in conventional models, the same clustering is used to compute deviation values (scores) for the different data points. A slightly different way of computing the outlier score is to fix the maximum allowed deviation (instead of the number of cluster centroids) and compute the number of cluster centroids required to achieve the same level of deviation, with and without a particular data point. It is this *increase* that is reported as the outlier score in the information-theoretic version of the same model. The idea here is that each point can be estimated by its closest cluster centroid, and the cluster centroids serve as a "code-book" in terms of which the data is compressed in a lossy way.

Information-theoretic models can be viewed as a complementary version of conventional models where a different aspect of the space-deviation trade-off curve is examined. Virtually every conventional model can be converted into an information-theoretic version by examining the bi-criteria space-deviation trade-off in terms of space rather than deviation. The bibliographic notes will also provide specific examples of each of the cases below:

1. The probabilistic model of Sect. 8.3 models the normal patterns in terms of generative model parameters such as the mixture means and covariance matrices. The space required by the model is defined by its complexity (e.g., number of mixture components), and the deviation corresponds to the probabilistic fit. In an information-theoretic version of the model, the complementary approach is to examine the size of the model required to achieve a fixed level of fit.

2. A clustering or density-based summarization model describes a data set in terms of cluster descriptions, histograms or other summarized representations. The granularity of these representations (number of cluster centroids, or histogram bins) controls the space, whereas the error in approximating the data point with a central element of the cluster (bin) defines the deviation. In conventional models, the size of the model (number of bins or clusters) is fixed, whereas in the information-theoretic version, the

maximum allowed deviation is fixed, and the required model size is reported as the outlier score.

3. A frequent pattern mining model describes the data in terms of an underlying code-book of frequent patterns. The larger the size of the code-book (by using frequent patterns of lower support), the more accurately the data can be described. These models are particularly popular, and some pointers are provided in the bibliographic notes.

All these models represent the data approximately in terms of individual condensed components representing aggregate trends. In general, outliers increase the length of the description in terms of these condensed components to achieve the same level of approximation. For example, a data set with outliers will require a larger number of mixture parameters, clusters, or frequent patterns to achieve *the same level of approximation*. Therefore, in information-theoretic methods, the components of these summary models are loosely referred to as "code books." Outliers are defined as data points whose removal results in the *largest decrease* in description length for the same error. The actual construction of the coding is often heuristic, and is not very different from the summary models used in conventional outlier analysis. In some cases, the description length for a data set can be *estimated* without explicitly constructing a code book, or building a summary model. An example is that of the *entropy* of a data set, or the *Kolmogorov complexity* of a string. Readers are referred to the bibliographic notes for examples of such methods.

While information-theoretic models are approximately equivalent to conventional models in that they explore the same trade-off in a slightly different way, they do have an advantage in some cases. These are cases where an accurate summary model of the data is hard to *explicitly* construct, and measures such as the entropy or Kolmogorov complexity can be used to *estimate* the compressed space requirements of the data set *indirectly*. In such cases, information-theoretic methods can be useful. In cases where the summary models can be explicitly constructed, it is better to use conventional models because the outlier scores are directly optimized to point-specific deviations rather than the more blunt measure of differential space impact. The bibliographic notes provide specific examples of some of the aforementioned methods.

## 8.8 Outlier Validity

As in the case of clustering models, it is desirable to determine the validity of outliers determined by a particular algorithm. Although the relationship between clustering and outlier analysis is complementary, the measures for outlier validity cannot easily be designed in a similar complementary way. In fact, validity analysis is much harder in outlier detection than data clustering. The reasons for this are discussed in the next section.

### 8.8.1 Methodological Challenges

As in the case of data clustering, outlier analysis is an unsupervised problem. Unsupervised problems are hard to validate because of the lack of external criteria, unless such criteria are synthetically generated, or some rare aspects of real data sets are used as *proxies*. Therefore, a natural question arises, as to whether *internal criteria* can be defined for outlier validation, as is the case for data clustering.

However, internal criteria are rarely used in outlier analysis. While such criteria are well-known to be flawed even in the context of data clustering, these flaws become significant enough to make these criteria unusable for outlier analysis. The reader is advised to refer to Sect. 6.9.1 of Chap. 6 for a discussion of the challenges of internal cluster validity. Most of these challenges are related to the fact that cluster validity criteria are derived from the objective function criteria of clustering algorithms. Therefore, a particular validity measure will favor (or *overfit*) a clustering algorithm using a similar objective function criterion. These problems become magnified in outlier analysis because of the *small sample solution space*. A model only needs to be correct on a few outlier data points to be considered a good model. Therefore, the overfitting of internal validity criteria, which is significant even in clustering, becomes even more problematic in outlier analysis. As a specific example, if one used the $k$-nearest neighbor distance as an internal validity measure, then a pure distance-based outlier detector will always outperform a locally normalized detector such as $LOF$. This is, of course, not consistent with known experience in real settings, where $LOF$ usually provides more meaningful results. One can try to reduce the overfitting effect by designing a validity measure which is different from the outlier detection models being compared. However, this is not a satisfactory solution because significant uncertainty always remains about the impact of hidden interrelationships between such measures and outlier detection models. The main problem with internal measures is that the relative bias in evaluation of various algorithms is *consistently* present, even when the data set is varied. A biased selection of internal measures can easily be abused in algorithm benchmarking.

Internal measures are almost never used in outlier analysis, although they are often used in clustering evaluation. Even in clustering, the use of internal validity measures is questionable in spite of its wider acceptance. Therefore, most of the validity measures used for outlier analysis are based on external measures such as the *Receiver Operating Characteristic* curve.

### 8.8.2 Receiver Operating Characteristic

Outlier detection algorithms are typically evaluated with the use of *external* measures where the *known* outlier labels from a synthetic data set or the *rare* class labels from a real data set are used as the ground-truth. This ground-truth is compared systematically with the outlier score to generate the final output. While such rare classes may not always reflect all the natural outliers in the data, the results are usually reasonably representative of algorithm quality, when evaluated over many data sets.

In outlier detection models, a threshold is typically used on the outlier score to generate a binary label. If the threshold is picked too restrictively to minimize the number of declared outliers then the algorithm will miss true outlier points (false-negatives). On the other hand, if the threshold is chosen in a more relaxed way, this will lead to too many false-positives. This leads to a trade-off between the false-positives and false-negatives. The problem is that the "correct" threshold to use is never known exactly in a real scenario. However, this entire trade-off curve can be generated, and various algorithms can be compared over the entire trade-off curve. One example of such a curve is the *Receiver Operating Characteristic (ROC)* curve.

For any given threshold $t$ on the outlier score, the declared outlier set is denoted by $\mathcal{S}(t)$. As $t$ changes, the size of $\mathcal{S}(t)$ changes as well. Let $\mathcal{G}$ represent the true set (ground-truth set) of outliers in the data set. The *true-positive rate*, which is also referred to as the *recall*, is defined as the percentage of *ground-truth* outliers that have been reported as outliers at threshold $t$.

$$TPR(t) = Recall(t) = 100 * \frac{|\mathcal{S}(t) \cap \mathcal{G}|}{|\mathcal{G}|}$$

Table 8.1: ROC construction with rank of ground-truth outliers

| **Algorithm** | **Rank of ground-truth outliers** |
|---|---|
| Algorithm A | 1, 5, 8, 15, 20 |
| Algorithm B | 3, 7, 11, 13, 15 |
| Random Algorithm | 17, 36, 45, 59, 66 |
| Perfect Oracle | 1, 2, 3, 4, 5 |

The false positive rate $FPR(t)$ is the percentage of the falsely reported positives out of the ground-truth negatives. Therefore, for a data set $\mathcal{D}$ with ground-truth positives $\mathcal{G}$, this measure is defined as follows:

$$FPR(t) = 100 * \frac{|\mathcal{S}(t) - \mathcal{G}|}{|\mathcal{D} - \mathcal{G}|}. \tag{8.16}$$

The ROC curve is defined by plotting the $FPR(t)$ on the $X$-axis, and $TPR(t)$ on the $Y$-axis for varying values of $t$. Note that the end points of the ROC curve are always at $(0, 0)$ and $(100, 100)$, and a random method is expected to exhibit performance along the diagonal line connecting these points. The *lift* obtained above this diagonal line provides an idea of the accuracy of the approach. The area under the ROC curve provides a concrete quantitative evaluation of the effectiveness of a particular method.

To illustrate the insights gained from these different graphical representations, consider an example of a data set with 100 points from which 5 points are outliers. Two algorithms, $A$ and $B$, are applied to this data set that rank all data points from 1 to 100, with lower rank representing greater propensity to be an outlier. Thus, the true-positive rate and false-positive rate values can be generated by determining the ranks of the 5 ground-truth outlier points. In Table 8.1, some hypothetical ranks for the five ground-truth outliers have been illustrated for the different algorithms. In addition, the ranks of the ground-truth outliers for a random algorithm have been indicated. The random algorithm outputs a random outlier score for each data point. Similarly, the ranks for a "perfect oracle" algorithm, which ranks the correct top 5 points as outliers, have also been illustrated in the table. The corresponding ROC curves are illustrated in Fig. 8.9.

What do these curves really tell us? For cases in which one curve strictly dominates another, it is clear that the algorithm for the former curve is superior. For example, it is immediately evident that the oracle algorithm is superior to all algorithms, and the random algorithm is inferior to all the other algorithms. On the other hand, algorithms $A$ and $B$ show domination at different parts of the ROC curve. In such cases, it is hard to say that one algorithm is strictly superior. From Table 8.1, it is clear that Algorithm $A$, ranks three of the correct ground-truth outliers very highly, but the remaining two outliers are ranked poorly. In the case of Algorithm $B$, the highest ranked outliers are not as well ranked as the case of Algorithm $A$, though all five outliers are determined much earlier in terms of rank threshold. Correspondingly, Algorithm $A$ dominates on the earlier part of the ROC curve whereas Algorithm $B$ dominates on the later part. Some practitioners use the area under the ROC curve as a proxy for the overall effectiveness of the algorithm, though such a measure should be used very carefully because all parts of the ROC curve may not be equally important for different applications.

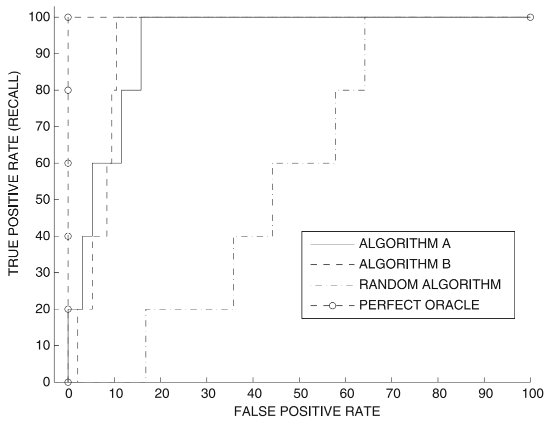


Figure 8.9: Receiver operating characteristic

### 8.8.3 Common Mistakes

A common mistake in benchmarking outlier analysis applications is that the area under the ROC curve is used repeatedly to tune the parameters of the outlier analysis algorithm. Note that such an approach implicitly uses the ground-truth labels for model construction, and it is, therefore, no longer an unsupervised algorithm. For problems such as clustering and outlier detection, it is not acceptable to use external labels in any way to tune the algorithm. In the particular case of outlier analysis, the accuracy can be *drastically* overestimated with such a tuning approach because the relative scores of a small number of outlier points have a very large influence on the ROC curve.

## 8.9 Summary

The problem of outlier analysis is an important one because of its applicability to a variety of problem domains. The common models in outlier detection include probabilistic models, clustering models, distance-based models, density-based models, and information-theoretic models. Among these, distance models are the most popular, but are computationally more expensive. A number of speed-up tricks have been proposed to make these models much faster. Local variations of distance-based models generally tend to be more effective because of their sensitivity to the generative aspects of the underlying data. Information-theoretic models are closely related to conventional models, and explore a different aspect of the space-deviation trade-off than conventional models.

Outlier validation is a difficult problem because of the challenges associated with the unsupervised nature of outlier detection, and the small sample-space problem. Typically, external validation criteria are used. The effectiveness of an outlier analysis algorithm is quantified with the use of the receiver operating characteristic curve that shows the trade-off between the false-positives and false-negatives for different thresholds on the outlier score. The area under this curve provides a quantitative evaluation of the outlier detection algorithm.

## 8.10 Bibliographic Notes

A number of books and surveys have been written on the problem of outlier analysis. The classical books [89, 259] in this area have mostly been written from the perspective of the statistics community. Most of these books were written before the wider adoption of database technology and are therefore not written from a computational perspective. More recently, this problem has been studied extensively by the computer science community. These works consider practical aspects of outlier detection corresponding to the cases where the data may be very large, or may have very high dimensionality. A recent book [5] also studies this area from the perspective of the computer science community. Numerous surveys have also been written that discuss the concept of outliers from different points of view, methodologies, or data types [61, 84, 131, 378, 380]. Among these, the survey by Chandola et al. [131] is the most recent and, arguably, the most comprehensive. It is an excellent review that covers the work on outlier detection quite broadly from the perspective of multiple communities.

The $Z$-value test is used commonly in the statistical literature, and numerous extensions, such as the $t$-value test are available [118]. While this test makes the normal distribution assumption for large data sets, it has been used fairly extensively as a good heuristic even for data distributions that do not satisfy the normal distribution assumption.

A variety of distance-based methods for outlier detection are proposed in [319, 436], and distance-correction methods for outlier detection are proposed in [109]. The determination of arbitrarily-shape clusters in the context of the *LOF* algorithm is explored in [487]. The agglomerative algorithm for discovering arbitrarily shaped neighborhoods, in the section on instance-specific Mahalanobis distance, is based on that approach. However, this method uses a connectivity-outlier factor, rather than the instance-specific Mahalanobis distance. The use of the Mahalanobis distance as a model for outlier detection was proposed in [468], though these methods are global, rather than local. A graph-based algorithm for local outlier detection is discussed in [257]. The *ORCLUS* algorithm also shows how to determine outliers in the presence of arbitrarily shaped clusters [22]. Methods for interpreting distance-based outliers were first proposed in [320].

A variety of information-theoretic methods for outlier detection are discussed in [68, 102, 160, 340, 472]. Many of these different models can be viewed in a complementary way to traditional models. For example, the work in [102] explores probabilistic methods in the context of information-theoretic models. The works in [68, 472] use code books of frequent-patterns for the modeling process. The connection between frequent patterns and compression has been explored in [470]. The use of measures such as entropy and Kolmogorov complexity for outlier analysis is explored in [340, 305]. The concept of coding complexity is explored in [129] in the context of set-based sequences.

Evaluation methods for outlier analysis are essentially identical to the techniques used in information retrieval for understanding precision-recall trade-offs, or in classification for ROC curve analysis. A detailed discussion may be found in [204].

## 8.11 Exercises

1. Suppose a particular random variable has mean 3 and standard deviation 2. Compute the $Z$-number for the values -1, 3. and 9. Which of these values can be considered the most extreme value?

**2.** Define the Mahalanobis-based extreme value measure when the $d$ dimensions are statistically independent of one another, in terms of the dimension-specific standard deviations $\sigma_1 \dots \sigma_d$.

**3.** Consider the four 2-dimensional data points $(0,0)$, $(0,1)$, $(1,0)$, and $(100,100)$. Plot them using mathematical software such as MATLAB. Which data point visually seems like an extreme value? Which data point is reported by the Mahalanobis measure as the strongest extreme value? Which data points are reported by depth-based measure?

**4.** Implement the EM algorithm for clustering, and use it to implement a computation of the probabilistic outlier scores.

**5.** Implement the Mahalanobis $k$-means algorithm, and use it to implement a computation of the outlier score in terms of the local Mahalanobis distance to the closest cluster centroids.

**6.** Discuss the connection between the algorithms implemented in Exercises 4 and 5.

**7.** Discuss the advantages and disadvantages of clustering models over distance-based models.

**8.** Implement a naive distance-based outlier detection algorithm with no pruning.

**9.** What is the effect of the parameter $k$ in $k$-nearest neighbor outlier detection? When do small values of $k$ work well and when do larger values of $k$ work well?

**10.** Design an outlier detection approach with the use of the *NMF* method of Chap. 6.

**11.** Discuss the relative effectiveness of pruning of distance-based algorithms in data sets which are (a) uniformly distributed, and (b) highly clustered with modest ambient noise and outliers.

**12.** Implement the *LOF*-algorithm for outlier detection.

**13.** Consider the set of 1-dimensional data points $\{1, 2, 2, 2, 2, 2, 6, 8, 10, 12, 14\}$. What are the data point(s) with the highest outlier score for a distance-based algorithm, using $k = 2$? What are the data points with highest outlier score using the *LOF* algorithm? Why the difference?

**14.** Implement the instance-specific Mahalanobis method for outlier detection.

**15.** Given a set of ground-truth labels, and outlier scores, implement a computer program to compute the ROC curve for a set of data points.

**16.** Use the objective function criteria of various outlier detection algorithms to design corresponding internal validity measures. Discuss the bias in these measures towards favoring specific algorithms.

**17.** Suppose that you construct a *directed* $k$-nearest neighbor graph from a data set. How can you use the degrees of the nodes to obtain an outlier score? What characteristics does this algorithm share with *LOF*?