Chapter 5

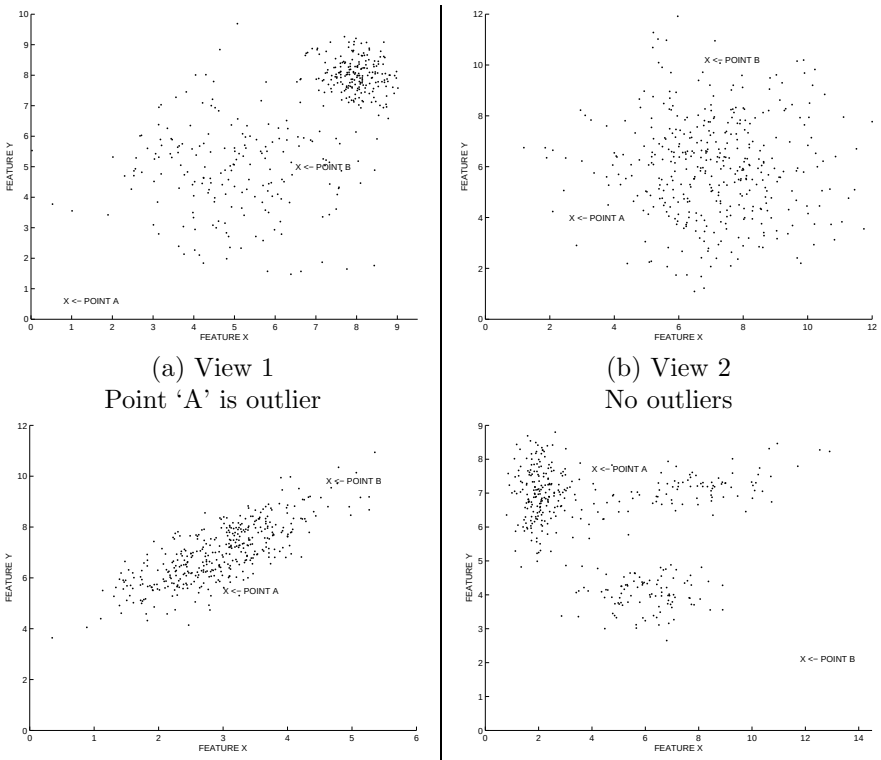
High-Dimensional Outlier Detection: The Subspace Method

"In view of all that we have said in the foregoing sections, the many obstacles we appear to have surmounted, what casts the pall over our victory celebration? It is the curse of dimensionality, a malediction that has plagued the scientist from the earliest days."– Richard Bellman

5.1 Introduction

Many real data sets are very high dimensional. In some scenarios, real data sets may contain hundreds or thousands of dimensions. With increasing dimensionality, many of the conventional outlier detection methods do not work very effectively. This is an artifact of the well-known *curse of dimensionality*. In high-dimensional space, the data becomes sparse, and the true outliers become masked by the noise effects of multiple irrelevant dimensions, when analyzed in *full dimensionality*.

A main cause of the dimensionality curse is the difficulty in defining the *relevant* locality of a point in the high-dimensional case. For example, proximity-based methods define locality with the use of distance functions on all the dimensions. On the other hand, all the dimensions may not be relevant for a specific test point, which also affects the quality of the underlying distance functions [263]. For example, all pairs of points are almost equidistant in high-dimensional space. This phenomenon is referred to as *data sparsity* or *distance concentration*. Since outliers are defined as data points in sparse regions, this results in a poorly discriminative situation where all data points are situated in almost equally sparse regions in full dimensionality. The challenges arising from the dimensionality curse are not specific to outlier detection. It is well known that many problems such as clustering and similarity search experience qualitative challenges with increasing dimensionality [5, 7, 121, 263]. In fact, it has been suggested that almost any algorithm that is based on the notion of proximity would degrade qualitatively in higher-dimensional space, and would therefore need to



(a) View 1
Point 'A' is outlier

(b) View 2
No outliers

(c) View 3
No outliers

(d) View 4
Point 'B' is outlier

Figure 5.1: The outlier behavior is masked by the irrelevant attributes in high dimensions.

be re-defined in a more meaningful way [8]. The impact of the dimensionality curse on the outlier detection problem was first noted in [4].

In order to further explain the causes of the ineffectiveness of full-dimensional outlier analysis algorithms, a motivating example will be presented. In Figure 5.1, four different 2-dimensional views of a hypothetical data set have been illustrated. Each of these views corresponds to a disjoint set of dimensions. It is evident that point 'A' is exposed as an outlier in the first view of the data set, whereas point 'B' is exposed as an outlier in the fourth view of the data set. However, neither of the data points 'A' and 'B' are exposed as outliers in the second and third views of the data set. These views are therefore *noisy* from the perspective of measuring the outlierness of 'A' and 'B.' In this case, three of the four views are quite non-informative and noisy for exposing any *particular* outlier 'A' or 'B.' In such cases, the outliers are lost in the random distributions within these views, when the distance measurements are performed in *full* dimensionality. This situation is often naturally magnified with increasing dimensionality. For data sets of very high dimensionality, it is possible that only a very small fraction of the views may be informative for the outlier analysis process.

What does the aforementioned pictorial illustration tell us about the issue of locally relevant dimensions? The physical interpretation of this situation is quite intuitive in practical scenarios. An object may have several measured quantities, and significantly abnormal behavior of this object may be reflected only in a small subset of these quantities. For ex-

ample, consider an airplane mechanical fault-detection scenario in which the results from different tests are represented in different dimensions. The results of thousands of different airframe tests on the same plane may mostly be normal, with some noisy variations, which are not significant. On the other hand, some deviations in a small subset of tests may be significant enough to be indicative of anomalous behavior. When the data from the tests are represented in full dimensionality, the anomalous data points will appear normal in virtually all views of the data except for a very small fraction of the dimensions. Therefore, aggregate proximity measures are unlikely to expose the outliers, since the noisy variations of the vast number of normal tests will mask the outliers. Furthermore, when different objects (instances of different airframes) are tested, different tests (subsets of dimensions) may be relevant for identifying the outliers. In other words, the outliers are often embedded in *locally* relevant subspaces.

What does this mean for full-dimensional analysis in such scenarios? When full-dimensional distances are used in order to measure deviations, the dilution effects of the vast number of "normally noisy" dimensions will make the detection of outliers difficult. In most cases, this will show up as distance-concentration effects from the noise in the other dimensions. This may make the computations more erroneous. Furthermore, the additive effects of the noise present in the large number of different dimensions will interfere with the detection of actual deviations. Simply speaking, *outliers are lost in low-dimensional subspaces, when full-dimensional analysis is used, because of the masking and dilution effects of the noise in full dimensional computations* [4].

Similar effects are also experienced for other distance-based methods such as clustering and similarity search. For these problems, it has been shown [5, 7, 263] that by examining the behavior of the data in subspaces, it is possible to design more meaningful clusters that are specific to the particular subspace in question. This broad observation is generally true of the outlier detection problem as well. Since the outliers may only be discovered in low-dimensional subspaces of the data, it makes sense to explore the lower dimensional subspaces for deviations of interest. Such an approach filters out the additive noise effects of the large number of dimensions and results in more robust outliers. An interesting observation is that such lower-dimensional projections can often be identified even in data sets with missing attribute values. This is quite useful for many real applications, in which feature extraction is a difficult process and full feature descriptions often do not exist. For example, in the airframe fault-detection scenario, it is possible that only a subset of tests may have been applied, and therefore the values in only a subset of the dimensions may be available for outlier analysis. This model is referred to as *projected outlier detection* or, alternatively, *subspace outlier detection* [4].

The identification of relevant subspaces is an extraordinarily challenging problem. This is because the number of possible projections of high-dimensional data is exponentially related to the dimensionality of the data. An effective outlier detection method would need to search the data points and dimensions in *an integrated way*, so as to reveal the most relevant outliers. This is because different subsets of dimensions may be relevant to different outliers, as is evident from the example in Figure 5.1. This further adds to the computational complexity.

An important observation is that subspace analysis is generally more difficult in the context of the outlier detection problem than in the case of problems such as clustering. This is because problems like clustering are based on aggregate behavior, whereas outliers, by definition, are rare. Therefore, in the case of outlier analysis, statistical aggregates on individual dimensions in a given locality often provide *very weak* hints for the subspace exploration process as compared to aggregation-based problems like clustering. When such

weak hints result in the omission of relevant dimensions, the effects can be much more drastic than the inclusion of irrelevant dimensions, especially in the interesting cases when the number of locally relevant dimensions is a small fraction of the full data dimensionality. A common mistake is to assume that the complementarity relationship between clustering and outlier analysis can be extended to the problem of local subspace selection. In particular, blind adaptations of dimension selection methods from earlier subspace clustering methods, which are unaware of the nuances of subspace analysis principles across different problems, may sometimes miss important outliers. In this context, it is also crucial to recognize the difficulty in identifying relevant subspaces for outlier analysis. In general, selecting a single relevant subspace for each data point can cause unpredictable results, and therefore it is important to combine the results from *multiple* subspaces. In other words, subspace outlier detection is inherently posed as an ensemble-centric problem.

Several classes of methods are commonly used:

- **Rarity-based:** These methods attempt to discover the subspaces based on rarity of the underlying distribution. The major challenge here is computational, since the number of rare subspaces is far larger than the number of dense subspaces in high dimensionality.

- **Unbiased:** In these methods, the subspaces are sampled in an unbiased way, and scores are combined across the sampled subspaces. When subspaces are sampled from the original set of attributes, the approach is referred to as *feature bagging* [344]. In cases in which arbitrarily oriented subspaces are sampled, the approach is referred to as *rotated bagging* [32] or *rotated subspace sampling*. In spite of their extraordinary simplicity, these methods often work well.

- **Aggregation-based:** In these methods, aggregate statistics such as cluster statistics, variance statistics, or non-uniformity statistics of global or local subsets of the data are used to quantify the relevance of subspaces. Unlike rarity-based statistics, these methods quantify the statistical properties of global or local reference sets of points instead of trying to identify rarely populated subspaces directly. Since such methods only provide weak (and error-prone) *hints* for identifying relevant subspaces, multiple subspace sampling is crucial.

This chapter is organized as follows. Axis-parallel methods for subspace outlier detection are studied in section 5.2. The underlying techniques discuss how multiple subspaces may be combined to discover outliers. The problem of identifying outliers in generalized subspaces (i.e., arbitrarily oriented subspaces) is discussed in section 5.3. Recent methods for finding outliers in nonlinear subspaces are also discussed in this section. The limitations of subspace analysis are discussed in section 5.4. The conclusions and summary are presented in section 5.5.

5.2 Axis-Parallel Subspaces

The first work on subspace outlier detection [4] proposed a model in which outliers were defined by axis-parallel subspaces. In these methods, an outlier is defined in a subset of features from the original data. Clearly, careful quantification is required for comparing the scores from various subspaces, especially if they are of different dimensionality and use different scales of reference. Furthermore, methods are required for quantifying the

effectiveness of various subspaces in exposing outliers. There are two major variations in the approaches used by axis-parallel methods:

- In one class of methods, points are examined one by one and their relevant outlying subspaces are identified. This is inherently an instance-based method. This type of approach is computationally expensive because a significant amount of computational time may be required for determining the outlier subspaces of each point. However, the approach provides a more fine-grained analysis, and it is also useful for providing *intensional knowledge*. Such intensional knowledge is useful for describing *why* a specific data point is an outlier.

- In the second class of methods, outliers are identified by building a subspace model up front. Each point is scored with respect to the model. In some cases, each model may correspond to a single subspace. Points are typically scored by using an ensemble score of the results obtained from different models. Even in cases in which a single (global) subspace is used in a model for scoring all the points, the combination score often enhances the local subspace properties of the scores because of the ability of ensemble methods to reduce *representational bias* [170] (cf. section 6.4.3 of Chapter 6).

The fine-grained analysis of the first class of methods is often computationally expensive. Because of the computationally intensive and fine-grained nature of this analysis, it is often harder to fully explore the use of multiple subspaces for analysis. This can sometimes have a detrimental effect on the accuracy as well. The second class of methods has clear computational benefits. This computational efficiency can be leveraged to explore a larger number of subspaces and provide more robust results. Many of the methods belonging to the second category, such as feature bagging, rotated bagging, subspace histograms, and isolation forests, are among the more successful and accurate methods for subspace outlier detection.

The advantages of ensemble-based analysis are very significant in the context of subspace analysis [31]. Since the outlier scores from different subspaces may be very different, it is often difficult to fully trust the score from a single subspace, and the combination of scores is crucial. This chapter will explore several methods that leverage the advantages of combining multiple subspaces.

5.2.1 Genetic Algorithms for Outlier Detection

The first approach for subspace outlier detection [4] was a genetic algorithm. Subspace outliers are identified by finding *localized regions of the data in low-dimensional space* that have abnormally low density. A genetic algorithm is employed to discover such local subspace regions. The outliers are then defined by their membership in such regions.

5.2.1.1 Defining Abnormal Lower-Dimensional Projections

In order to identify abnormal lower-dimensional projections, it is important to provide a proper statistical definition of an abnormal lower-dimensional projection. An abnormal lower-dimensional projection is one in which the density of the data is exceptionally lower than average. In this context, the methods for extreme-value analysis introduced in Chapter 2 are useful.

A grid-based approach is used in order to identify rarely populated local subspace regions. The first step is to create grid regions with data discretization. Each attribute is

divided into ϕ ranges. These ranges are created on an equi-depth basis. Thus, each range contains a fraction $f = 1/\phi$ of the records. The reason for using equi-depth ranges as opposed to equi-width ranges is that different localities of the data may have different densities. Therefore, such an approach partially adjusts for the local variations in data density during the initial phase. These ranges form the units of locality that are used in order to define sparse subspace regions.

Consider a k-dimensional cube that is created by selecting grid ranges from k different dimensions. If the attributes are statistically independent, the expected fraction of the records in that k-dimensional region is f^k. Of course, real-world data is usually far from statistically independent and therefore the actual distribution of points in a cube would differ significantly from this expected value. Many of the local regions may contain very few data points and most of them will be empty with increasing values of k. In cases where these abnormally sparse regions are non-empty, the data points inside them might be outliers.

It is assumed that the total number of points in the database is denoted by N. Under the aforementioned independence assumption, the presence or absence of any point in a k-dimensional cube is a Bernoulli random variable with probability f^k. Then, from the properties of Bernoulli random variables, we know that the expected fraction and standard deviation of the points in a k-dimensional cube is given by $N \cdot f^k$ and $\sqrt{N \cdot f^k \cdot (1 - f^k)}$, respectively. Furthermore, if the number of data points N is large, the central limit theorem can be used to *approximate* the number of points in a cube by a normal distribution. Such an assumption can help in creating meaningful measures of abnormality (sparsity) of the cube. Let $n(\mathcal{D})$ be the number of points in a k-dimensional cube $\mathcal{D}$. The sparsity coefficient $S(\mathcal{D})$ of the data set $\mathcal{D}$ can be computed as follows:

$$S(\mathcal{D}) = \frac{n(D) - N \cdot f^k}{\sqrt{N \cdot f^k \cdot (1 - f^k)}} \tag{5.1}$$

Only sparsity coefficients that are negative are indicative of local projected regions for which the density is lower than expectation. Since $n(\mathcal{D})$ is assumed to fit a normal distribution, the normal distribution tables can be used to quantify the probabilistic level of significance of its deviation. Although the independence assumption is never really true, it provides a good practical *heuristic* for estimating the point-specific abnormality.

5.2.1.2 Defining Genetic Operators for Subspace Search

An exhaustive search of all the subspaces is impractical because of exponential computational complexity. Therefore, a selective search method, which prunes most of the subspaces, is required. The nature of this problem is such that there are no upward- or downward-closed properties[1] on the grid-based subspaces satisfying the sparsity condition. Such properties are often leveraged in other problems like frequent pattern mining [36]. However, unlike frequent pattern mining, in which one is looking for patterns with high frequency, the problem of finding sparsely-populated subsets of dimensions has the flavor of finding a needle in haystack. Furthermore, even though particular regions may be well populated on certain subsets of dimensions, it is possible for them to be very sparsely populated when such dimensions are combined. For example, in a given data set, there may be a large number of individuals clustered at the age of 20, and a modest number of individuals with high levels of severity of Alzheimer's disease. However, *very rare* individuals would satisfy

[1] An upward-closed pattern is one in which all supersets of the pattern are also valid patterns. A downward-closed set of patterns is one in which all subsets of the pattern are also members of the set.

both criteria, because the disease does not affect young individuals. From the perspective of outlier detection, a 20-year old with early-onset Alzheimer is a very interesting record. However, the interestingness of the pattern is not even hinted at by its lower-dimensional projections. Therefore, the best projections are often created by an unknown combination of dimensions, whose lower-dimensional projections may contain very few hints for guiding subspace exploration. One solution is to change the measure in order to force better closure or pruning properties; however, forcing the choice of the measure to be driven by algorithmic considerations is often a recipe for poor results. In general, it is not possible to predict the effect of combining two sets of dimensions on the outlier scores. Therefore, a natural option is to develop search methods that can identify such hidden combinations of dimensions. An important observation is that one can view the problem of finding sparse subspaces as an optimization problem of minimizing the count of the number of points in the identified subspaces. However, since the number of subspaces increases exponentially with data dimensionality, the work in [4] uses genetic algorithms, which are also referred to as evolutionary search methods. Such optimization methods are particularly useful in unstructured settings where there are few hard rules to guide the search process.

Genetic algorithms, also known as evolutionary algorithms [273], are methods that imitate the process of organic evolution in order to solve poorly structured optimization problems. In evolutionary methods, every solution to an optimization problem can be represented as an individual in an evolutionary system. The measure of fitness of this "individual" is equal to the objective function value of the corresponding solution. As in biological evolution, an individual has to compete with other individuals that are alternative solutions to the optimization problem. Therefore, one always works with a multitude (i.e., *population*) of solutions at any given time, rather than a single solution. Furthermore, new solutions can be created by recombination of the properties of older solutions, which is the analog of the process of biological reproduction. Therefore, appropriate operations are defined in order to imitate the recombination and mutation processes in order to complete the simulation. A mutation can be viewed as a way of exploring closely related solutions for possible improvement, much as one would do in a hill-climbing approach.

Clearly, in order to simulate this biological process, we need some type of concise representation of the solutions to the optimization problem. This representation enables a concrete algorithm for simulating the algorithmic processes of recombination and mutation. Each feasible solution is represented as a string, which can be viewed as the chromosome representation of the solution. The process of conversion of feasible solutions into strings is referred to as its *encoding*. The effectiveness of the evolutionary algorithm often depends crucially on the choice of encoding because it implicitly defines all the operations used for search-space exploration. The measure of fitness of a string is evaluated by the *fitness function*. This is equivalent to an evaluation of the objective function of the optimization problem. Therefore, a solution with a better objective function value can be viewed as the analog of a *fitter* individual in the biological setting. When evolutionary algorithms simulate the process of biological evolution, it generally leads to an improvement in the average objective function of all the solutions (population) at hand much as the biological evolution process improves fitness over time. Furthermore, because of the perpetual (selection) bias towards fitter individuals, diversity in the population of solutions is lost. This loss of diversity resembles the way in which convergence works in other types of iterative optimization algorithms. De Jong [163] defined convergence of a particular position in the string as the stage at which 95% of the population had the same value for that position. The population is said to have converged when all positions in the string representation have converged.

The evolutionary algorithm views subspace projections as possible solutions to the op-

timization problem. Such projections can be easily represented as strings. Since the data is discretized into a grid structure, we can assume that the identifiers of the various grid intervals in any dimension range from 1 to ϕ. Consider a d-dimensional data point for which the grid intervals for the d different dimensions are denoted by $(m_1, \ldots m_d)$. The value of each m_i can take on any of the value from 1 through ϕ, or it can take on the value $*$, which indicates a "don't care" value. Thus, there are a total of $\phi+1$ possible values of m_i. Consider a 4-dimensional data set with $\phi = 10$. Then, one possible example of a solution to the problem is given by the string *3*9. In this case, the ranges for the second and fourth dimension are identified, whereas the first and third are left as "don't cares." The evolutionary algorithm uses the dimensionality of the projection k as an input parameter. Therefore, for a d-dimensional data set, the string of length d will contain k specified positions and $(d - k)$ "don't care" positions. This represents the string encoding of the k-dimensional subspace. The fitness for the corresponding solution may be computed using the sparsity coefficient discussed earlier. The evolutionary search technique starts with a population of p random solutions and iteratively uses the processes of selection, crossover, and mutation in order to perform a combination of hill climbing, solution recombination and random search over the space of possible projections. The process is continued until the population converges to a global optimum according to the *De Jong convergence criterion* [163]. At each stage of the algorithm, the m best projection solutions (most negative sparsity coefficients) are tracked in running fashion. At the end of the algorithm, these solutions are reported as the best projections in the data. The following operators are defined for selection, crossover, and mutation:

- **Selection:** The copies of a solution are replicated by ordering them by rank and biasing them in the population in the favor of higher ranked solutions. This is referred to as *rank selection*.

- **Crossover:** The crossover technique is key to the success of the algorithm, since it implicitly defines the subspace exploration process. One solution is to use a uniform two-point crossover in order to create the recombinant children strings. The two-point crossover mechanism works by determining a point in the string at random called the crossover point, and exchanging the segments to the right of this point. However, such a blind recombination process may create poor solutions too often. Therefore, an optimized crossover mechanism is defined. In this case, it is guaranteed that both children solutions correspond to a k-dimensional projection as the parents, and the children typically have high fitness values. This is achieved by examining a subset of the different possibilities for recombination and selecting the best among them. The basic idea is to select k dimensions greedily from the space of (at most) $2 \cdot k$ distinct dimensions included in the two parents. A detailed description of this optimized crossover process is provided in [4].

- **Mutation:** In this case, random positions in the string are flipped with a predefined mutation probability. Care must be taken to ensure that the dimensionality of the projection does not change after the flipping process.

At termination, the algorithm is followed by a postprocessing phase. In the postprocessing phase, all data points containing the abnormal projections are reported by the algorithm as the outliers. The approach also provides the relevant projections which provide the *causality* for the outlier behavior of a data point. Thus, this approach has a high degree of interpretability.

5.2.2 Finding Distance-Based Outlying Subspaces

After the initial proposal of the basic subspace outlier detection framework [4], one of the earliest methods along this line was the *HOS-Miner* approach. Several different aspects of the broader ideas associated with *HOS-Miner* are discussed in [605, 606, 607]. A first discussion of the *HOS-Miner* approach was presented in [605]. According to this work, the definition of the outlying subspace for a given data point $\overline{X}$ is as follows:

Definition 5.2.1 *For a given data point $\overline{X}$, determine the set of subspaces such that the sum of its k-nearest neighbor distances in that subspace is at least δ.*

This approach does not normalize the distances with the number of dimensions. Therefore, a subspace becomes more likely to be outlying with increasing dimensionality. This definition also exhibits closure properties in which any subspace of a non-outlying subspace is also not outlying. Similarly, every superset of an outlying subspace is also outlying. Clearly, only *minimal* subspaces that are outliers are interesting. The method in [605] uses these closure properties to prune irrelevant or uninteresting subspaces. Although the aforementioned definition has desirable closure properties, the use of a fixed threshold δ across subspaces of different dimensionalities seems unreasonable. Selecting a definition based on algorithmic convenience can often cause poor results. As illustrated by the earlier example of the young Alzheimer patient, true outliers are often hidden in subspaces of the data, which cannot be inferred from their lower- or higher-dimensional projections.

An X-Tree is used in order to perform the indexing for performing the k-nearest neighbor queries in different subspaces efficiently. In order to further improve the efficiency of the learning process, the work in [605] uses a random sample of the data in order to learn about the subspaces before starting the subspace exploration process. This is achieved by estimating a quantity called the *Total Savings Factor (TSF)* of the outlying subspaces. These are used to regulate the search process for specific query points and prune the different subspaces in an ordered way. Furthermore, the TSF values of different subspaces are dynamically updated as the search proceeds. It has been shown in [605] that such an approach can be used in order to determine the outlying subspaces of specific data points efficiently. Numerous methods for using different kinds of pruning properties and genetic algorithms for finding outlying subspaces are presented in [606, 607].

5.2.3 Feature Bagging: A Subspace Sampling Perspective

The simplest method for combining outliers from multiple subspaces is the use of *feature bagging* [344], which is an ensemble method. Each base component of the ensemble uses the following steps:

- Randomly select an integer r from $\lfloor d/2 \rfloor$ to $(d-1)$.
- Randomly select r features (without replacement) from the underlying data set in iteration t in order to create an r-dimensional data set D_t in the tth iteration.
- Apply the outlier detection algorithm O_t on the data set D_t in order to compute the score of each data point.

In principle, one could use a different outlier detection algorithm in each iteration, provided that the scores are normalized to Z-values after the process. The normalization is also necessary to account for the fact that different subspace samples contain a different number of features. However, the work in [344] uses the LOF algorithm for all the iterations. Since the

LOF algorithm returns inherently normalized scores, such a normalization is not necessary. At the end of the process, the outlier scores from the different algorithms are combined in one of two possible ways:

- *Breadth-first approach:* In this approach, the ranking of the algorithms is used for combination purposes. The top-ranked outliers over all the different executions are ranked first, followed by the second-ranked outliers (with repetitions removed), and so on. Minor variations could exist because of tie-breaking between the outliers within a particular rank.

- *Cumulative-sum approach:* The outlier scores over the different algorithm executions are summed up. The top ranked outliers are reported on this basis. One can also view this process as equivalent to the averaging combination function in an ensemble method (cf. Chapter 6).

It was experimentally shown in [344] that such methods are able to ameliorate the effects of irrelevant attributes. In such cases, full-dimensional algorithms are unable to distinguish the true outliers from the normal data, because of the additional noise.

At first sight, it would seem that random subspace sampling [344] does not attempt to optimize the discovery of relevant subspaces at all. Nevertheless, it does have the paradoxical merit that it is relatively efficient to sample subspaces, and therefore a large number of subspaces can be sampled in order to improve robustness. Even though each detector selects a global subspace, the *ensemble-based combined score* of a given point is able to implicitly benefit from the locally-optimized subspaces. This is because different points may obtain favorable scores in different subspace samples, and the ensemble combination is often able to identify all the points that are favored in a sufficient number of the subspaces. This phenomenon can also be formally explained in terms of the notion of how ensemble methods reduce representational bias [170] (cf. section 6.4.3.1 of Chapter 6). In other words, ensemble methods provide an implicit route for converting global subspace exploration into local subspace selection and are therefore inherently more powerful than their individual components. An ensemble-centric perspective on feature bagging is provided in section 6.4.3.1.

The robustness resulting from multiple subspace sampling is clearly a very desirable quality, as long as the combination function at the end recognizes the differential behavior of different subspace samples for a given data point. In a sense, this approach implicitly recognizes the difficulty of detecting relevant and rare subspaces, and therefore samples as many subspaces as possible in order to reveal the rare behavior. From a conceptual perspective, this approach is similar to that of harnessing the power of many weak learners to create a single strong learner in classification problems. The approach has been shown to show consistent performance improvement over full-dimensional methods for many real data sets in [344]. This approach may also be referred to as the *feature bagging method* or *random subspace ensemble method*. Even though the original work [344] uses LOF as the base detector, the average k-nearest neighbor detector has also been shown to work [32].

5.2.4 Projected Clustering Ensembles

Projected clustering methods define clusters as sets of points together with sets of dimensions in which these points cluster well. As discussed in Chapter 4, clustering and outlier detection are complementary problems. Therefore, it is natural to investigate whether projected or subspace clustering methods can also be used for outlier detection. Although the

relevant subspaces for clusters are not always relevant for outlier detection, there is still a weak relationship between the two. By using ensembles, it is possible to strengthen the types of outliers discovered using this approach.

As shown in the *OutRank* work [406], one can use ensembles of projected clustering algorithms [5] for subspace outlier detection. In this light, it has been emphasized in [406] that the use of *multiple* projected clusterings is essential because the use of a single projected clustering algorithm provides very poor results. The basic idea in *OutRank* is to use the following procedure repeatedly:

- Use a randomized projected clustering method like PROCLUS [5] on the data set to create a set of projected clusters.

- Quantify the outlier score of each point based on its similarity to the cluster to which it belongs. Examples of relevant scores include the size, dimensionality, (projected) distance to cluster centroid, or a combination of these factors. The proper choice of measure is sensitive to the specific clustering algorithm that is used.

This process is applied repeatedly, and the scores are averaged in order to yield the final result. The use of a sufficiently randomized clustering method in the first step is crucial for obtaining good results with this ensemble-centric approach.

There are several variations one might use for the scoring step. For a distance-based algorithm like PROCLUS, it makes sense to use the same distance measure to quantify the outlier score as was used for clustering. For example, one can use the Manhattan segmental distance of a data point from its nearest cluster centroid in the case of PROCLUS. The Manhattan segmental distance is estimated by first computing the Manhattan distance of the point to the centroid of the cluster in its relevant subspace and then dividing by the number of dimensions in that subspace. However, this measure ignores the number of points and dimensionality of the clusters; furthermore, it is not applicable for pattern-based methods with overlapping clusters. A natural approach is the *individual weighting* measure. For a point, the fraction of the number of points in its cluster to maximum cluster size is computed, and the fraction of the number of dimensions in its cluster subspace to the maximum cluster-subspace dimensionality is also computed. A simple outlier score is to add these two fractions over all clusters in which the point occurs and then divide by the total number of clusters in the data. A point that is included in many large and high-dimensional subspace clusters is unlikely to be an outlier. Therefore, points with smaller scores are deemed as outliers. A similar method for binary and categorical data is discussed in section 8.5.1 of Chapter 8. Two other measures, referred to as cluster coverage and subspace similarity, are proposed in [406]. The work in [406] experimented with a number of different clustering algorithms and found that using multiple randomized runs of PROCLUS yields the best results. This variation is referred to as *Multiple-Proclus*. The cluster coverage and subspace similarity measures were used to achieve these results, although reasonable results were also achieved with the individual weighting measure.

The key point to understand about such clustering-based methods is that the type of outlier discovered is sensitive to the underlying clustering, although an ensemble-centric approach is essential for success. Therefore, a locality-sensitive clustering ensemble will make the outliers locality-sensitive; a subspace-sensitive clustering ensemble will make the outliers subspace-sensitive, and a correlation-sensitive clustering ensemble will make the outliers correlation-sensitive.

5.2.5 Subspace Histograms in Linear Time

A linear-time implementation of subspace histograms with hashing is provided in [476] and is referred to as *RS-Hash*. The basic idea is to repeatedly construct grid-based histograms on data samples of size s and combine the scores in an ensemble-centric approach. Each histogram is constructed on a randomly chosen subspace of the data. The dimensionality of the subspace and size of the grid region is specific to its ensemble component. In the testing phase of the ensemble component, all N points in the data are scored based on the logarithm of the number of points (from the training sample) in its grid region. The approach is repeated with multiple samples of size s. The point-specific scores are averaged over different ensemble components to create a final result, which is highly robust. The variation in the dimensionality and size of the grid regions is controlled with am integer dimensionality parameter r and a fractional grid-size parameter $f \in (0, 1)$, which vary randomly over different ensemble components. We will describe the process of random selection of these parameters later. For now, we assume (for simplicity) that the values of these parameters are fixed.

The sample S of size $s \ll N$ may be viewed as the training data of a single ensemble component, and each of the N points is scored in this component by constructing a subspace histogram on this training sample. First, a set V of r dimensions is randomly sampled from the d dimensions, and all the scoring is done on histograms built in this r-dimensional subspace. The minimum value min_j and the maximum value max_j of the jth dimension are determined from this sample. Let x_{ij} denote the jth dimension of the ith point. All $s \cdot r$ values x_{ij} in the training sample, such that $j \in V$, are normalized as follows:

$$x'_{ij} \Leftarrow \frac{x_{ij} - min_j}{max_j - min_j} \tag{5.2}$$

One can even use $x'_{ij} \Leftarrow x_{ij}/(max_j - min_j)$ for implementation simplicity. At the time of normalization, we also create the following r-dimensional *discretized* representation of the training points, where for each of the r dimensions in V, we use a grid-size of width $f \in (0, 1)$. Furthermore, to induce diversity across ensemble components, the placement of the grid partitioning points is varied across ensemble components. In a given ensemble component, a grid partitioning point is not placed at 0, but at a value of $-\alpha_j$ for dimension j. This can be achieved by setting the discretized identifier of point i and dimension j to $\lfloor (x'_{ij} + \alpha_j)/f \rfloor$. The value of α_j is fixed within an ensemble component up front at a value randomly chosen from $(0, f)$. This r-dimensional discretized representation provides the identity of the r-dimensional bounding box of that point. A hash table maintains a count of each of the bounding boxes encountered in the training sample. For each of the s training points, the count of its bounding box is incremented by hashing this r-dimensional discrete representation, which requires constant time. In the testing phase, the discretized representation of each of the N points is again constructed using the aforementioned process, and its count is retrieved from the hash table constructed on the training sample. Let $n_i \leq s$ denote this count of the ith point. For points included in the training sample S, the outlier score of the ith point is $\log_2(n_i)$, whereas for points not included in the training sample S, the score is $\log_2(n_i + 1)$. This process is repeated over multiple ensemble components (typically 100), and the average score of each point is returned. Low scores represent outliers.

It is noteworthy that the values of min_j and max_j used in the testing phase of an ensemble component are the same as those estimated from the training sample. Thus, the training phase requires only $O(s)$ time, and the value of s is typically a small constant such as 1000. The testing phase of each ensemble component requires $O(N)$ time, and the overall

algorithm requires $O(N + s)$ time. The constant factors also tend to be very small and the approach is extremely fast.

We now describe the process of randomly selecting f, r, and α_j up front in each ensemble component. The value of f is selected uniformly at random from $(1/\sqrt{s}, 1 - 1/\sqrt{s})$. Subsequently, the value of r is set uniformly at random to an integer between $1 + 0.5 \cdot [\log_{\max\{2,1/f\}}(s)]$ and $\log_{\max\{2,1/f\}}(s)$. The value of each α_j is selected uniformly at random from $(0, f)$. This variation of grid size (with f), dimensionality (with r), and placement of grid regions (with α_j) provides additional diversity, which is helpful to the overall ensemble result. The work in [476] has also shown how the approach may be extended to data streams. This variant is referred to as *RS-Stream*. The streaming variant requires greater sophistication in the hash-table design and maintenance, although the overall approach is quite similar.

5.2.6 Isolation Forests

The work in [367] proposes a model called *isolation forests*, which shares some intuitive similarity with another ensemble technique known as *random forests*. Random forests are among the most successful models used in classification and are known to outperform the majority of classifiers in a variety of problem domains [195]. However, the unsupervised way in which an isolation forest is constructed is quite different, especially in terms of how a data point is scored. As discussed in section 5.2.6.3, the isolation forest is also a special case of the *extremely randomized clustering forest (ERC-Forest)* for clustering [401].

An isolation forest is an ensemble combination of a set of *isolation trees*. In an isolation tree, the data is recursively partitioned with axis-parallel cuts at randomly chosen partition points in randomly selected attributes, so as to isolate the instances into nodes with fewer and fewer instances until the points are isolated into singleton nodes containing one instance. In such cases, the tree branches containing outliers are noticeably less deep, because these data points are located in sparse regions. Therefore, the distance of the leaf to the root is used as the outlier score. The final combination step is performed by averaging the path lengths of the data points in the different trees of the isolation forest.

Isolation forests are intimately related to subspace outlier detection. The different branches correspond to different local subspace regions of the data, depending on how the attributes are selected for splitting purposes. The smaller paths correspond to lower dimensionality[2] of the subspaces in which the outliers have been isolated. The less the dimensionality that is required to isolate a point, the stronger the outlier that point is likely to be. In other words, isolation forests work under the implicit assumption that it is more likely to be able to isolate outliers in subspaces of lower dimensionality created by random splits. For instance, in our earlier example on Alzheimer's patients, a *short* sequence of splits such as $Age \leq 30, Alzheimer = 1$ is likely to isolate a rare individual with early-onset Alzheimer's disease.

The training phase of an isolation forest constructs multiple isolation trees, which are unsupervised equivalents of decision trees. Each tree is binary, and has at most N leaf nodes for a data set containing N points. This is because each leaf node contains exactly one data point by default, but early termination is possible by parameterizing the approach with

[2]Although it is possible to repeatedly cut along the same dimension, this becomes less likely in higher-dimensional data sets. In general, the path length is highly correlated with the dimensionality of the subspace used for isolation. For a data set containing $2^8 = 256$ points (which is the recommended subsample size), the average depth of the tree will be 8, but an outlier might often be isolated in less than three or four splits (dimensions).

a height parameter. In order to construct the isolation tree from a data set containing N points, the approach creates a root node containing all the points. This is the initial state of the isolation tree $\mathcal{T}$. A candidate list $\mathcal{C}$ (for further splitting) of nodes is initialized as a singleton list containing the root node. Then, the following steps are repeated to create the isolation tree $\mathcal{T}$ until the candidate list $\mathcal{C}$ is empty:

1. Select a node R from $\mathcal{C}$ randomly and remove from $\mathcal{C}$.

2. Select a random attribute i and split the data in R into two sets R_1 and R_2 at a random value a chosen along that attribute. Therefore, all data points in R_1 satisfy $x_i \leq a$ and all data points in R_2 satisfy $x_i > a$. The random value a is chosen uniformly at random between the minimum and maximum values of the ith attribute among data points in node R. The nodes R_1 and R_2 are children of R in $\mathcal{T}$.

3. **(Step performed for each $i \in \{1, 2\}$):** If R_i contains more than one point then add it to $\mathcal{C}$. Otherwise, designate the node as an isolation-tree leaf.

This process will result in the creation of a binary tree that is typically not balanced. Outlier nodes will tend to be isolated more quickly than non-outlier nodes. For example, a point that is located very far away from the remaining data along all dimensions would likely be separated as an isolated leaf in the very first iteration. Therefore, the length of the path from the root to the leaf is used as the outlier score. Note that the path from the root to the leaf defines a subspace of varying dimensionality. Outliers can typically be isolated in much lower-dimensional subspaces than normal points. Therefore, the number of edges from the root to a node is equal to its outlier score. Smaller scores correspond to outliers. This inherently randomized approach is repeated multiple times and the scores are averaged to create the final result. The ensemble-like approach is particularly effective and it often provides results of high quality. The basic version of the isolation tree, when grown to full height, is *parameter-free*. This characteristic is always a significant advantage in unsupervised problems like outlier detection. The *average-case* computational complexity of the approach is $\theta(N\log(N))$, and the space complexity is $O(N)$ for each isolation tree.

One can always improve computational efficiency and often improve accuracy with subsampling. The subsampling approach induces further diversity, and gains some of the advantages inherent in outlier ensembles [31]. In this case, the isolation tree is constructed using a subsample in the *training phase*, although all points are scored against the subsample in the *testing phase*. The training phase returns the tree structure together with the split conditions (such as $x_i \leq a$ and $x_i > a$) at each node. Out-of-sample points are scored in the testing phase by using the split conditions computed during the training phase. Much like the testing phase of a decision tree, the appropriate leaf node for an out-of-sample point is identified by traversing the appropriate path from the root to the leaf with the use of the split conditions, which are simple univariate inequalities.

The use of a subsample results in better computational efficiency and better diversity. It is stated in [367] that a subsample size of 256 works well in practice, although this value might vary somewhat with the data set at hand. Note that this results in *constant* computational and memory requirements for building the tree, irrespective of data set size. The testing phase requires average-case complexity of $\theta(\log(N))$ for each data point if the entire data set is used. On the other hand, the testing phase requires constant time for *each point*, if subsamples of size 256 are used. Therefore, if subsampling is used with a constant number of samples and a constant number of trials, the running time is constant for the training phase, $O(N)$ for the testing phase, and the space complexity is constant as well.

The isolation forest is an efficient method, which is noteworthy considering the fact that most subspace methods are computationally intensive.

The final combination step is performed by averaging the path lengths of a data point in the different trees of the isolation forest. A detailed discussion of this approach from an ensemble-centric point of view is provided in section 6.4.5 of Chapter 6. Practical implementations of this approach are available on the Python library *scikit-learn* [630] and R library *SourceForge* [631].

5.2.6.1 Further Enhancements for Subspace Selection

The approach is enhanced by additional pre-selection of feature with a *Kurtosis measure*. The Kurtosis of a set of feature values $x_1 \ldots x_N$ is computed by first standardizing them to $z_1 \ldots z_N$ with zero mean and unit standard deviation:

$$z_i = \frac{x_i - \mu}{\sigma} \tag{5.3}$$

Here, μ is the mean and σ is the standard deviation of $x_1 \ldots x_N$. Then, the Kurtosis is computed as follows:

$$K(z_1 \ldots z_N) = \frac{\sum_{i=1}^{N} z_i^4}{N} \tag{5.4}$$

Features that are very non-uniform will show a high level of Kurtosis. Therefore, the Kurtosis computation can be viewed as a feature selection measure for anomaly detection. The work in [367] *preselects* a subset of attributes based on the ranking of their univariate Kurtosis values and then constructs the random forest after (globally) throwing away those features. Note that this results in global subspace selection; nevertheless the random split approach is still able to explore different local subspaces, albeit randomly (like feature bagging). A generalization of the Kurtosis measure, referred to as *multidimensional* Kurtosis, is discussed in section 1.3.1 of Chapter 1. This measure evaluates subsets of features jointly using Equation 5.4 on the Mahalanobis distances of points in that subspace rather than ranking features with univariate Kurtosis values. This measure is generally considered to be more effective than univariate Kurtosis but it is computationally expensive because it needs to be coupled with feature subset exploration in a structured way. Although the generalized Kurtosis measure has not been used in the isolation forest framework, it has the potential for application within settings in which computational complexity is not as much of a concern.

5.2.6.2 Early Termination

A further enhancement is that the tree is not grown to full height. The growth of a node is stopped as soon as a node contains either duplicate instances or it exceeds a certain threshold height. This threshold height is set to 10 in the experiments of [367]. In order to estimate the path length of points in such nodes, an additional credit needs to be assigned to account for the fact that the points in these nodes have not been materialized to full height. For a node containing r instances, its additional credit $c(r)$ is defined as the expected path length in a binary search tree with r points [442]:

$$c(r) = \ln(r-1) - \frac{2(r-1)}{r} + 0.5772 \tag{5.5}$$

Note that this credit is added to the path length of that node from the root to compute the final outlier score. Early termination is an efficiency-centric enhancement, and one can choose to grow the trees to full length if needed.

5.2.6.3 Relationship to Clustering Ensembles and Histograms

The isolation forest can be viewed as a type of clustering ensemble as discussed in section 5.2.4. The isolation tree creates hierarchical projected clusters from the data, in which clusters are defined by their bounding boxes. A bounding box of a cluster (node) is defined by the sequence of axis-parallel splits from the root to that node. However, compared to most projected clustering methods, the isolation tree is *extremely* randomized because of its focus on ensemble-centric performance. The isolation forest is a decision-tree-based approach to clustering. Interestingly, the basic isolation forest can be shown to be a variation of an earlier clustering ensemble method, referred to as *extremely randomized clustering forests (ERC-Forests)* [401]. The main difference is that the ERC-Forest uses multiple trials at each node to enable a small amount of supervision with class labels; however, by setting the number of trials to 1 and growing the tree to full length, one can obtain an unsupervised isolation tree as a special case. Because of the *space*-partitioning (rather than *point*-partitioning) methodology used in ERC-Forests and isolation forests, these methods also share some intuitive similarities with histogram- and density-based methods. An isolation tree creates hierarchical and randomized grid regions, whose expected volume reduces by a factor of 2 with each split. The path length in an isolation tree is therefore a rough surrogate for the negative logarithm of the (fractional) volume of a maximal grid region containing a single data point. This is similar to the notion of log-likelihood density used in traditional histograms. Unlike traditional histograms, isolation forests are not parameterized by grid-width and are therefore more flexible in handling data distributions of varying density. Furthermore, the flexible shapes of the histograms in the latter naturally define local subspace regions.

The measures for scoring the outliers with projected clustering methods [406] also share some intuitive similarities with isolation forests. One of the measures in [406] uses the sum of the subspace dimensionality of the cluster and the number of points in the cluster as an outlier score. Note that the subspace dimensionality of a cluster is a rough proxy for the path length in the isolation tree. Similarly, some variations of the isolation tree, referred to as half-space trees [532], use fixed-height trees. In these cases, the number of points in the relevant node for a point is used to define its outlier score. This is similar to clustering-based outlier detection methods in which the number of points in the nearest cluster is often used as an important component of the outlier score.

5.2.7 Selecting High-Contrast Subspaces

The feature bagging method [344] discussed in section 5.2.3 randomly samples subspaces. If many dimensions are irrelevant, at least a few of them are likely to be included in each subspace sample. At the same time, information is lost because many dimensions are dropped. These effects are detrimental to the accuracy of the approach. Therefore, it is natural to ask whether it is possible to perform a pre-processing in which a smaller number of *high-contrast* subspaces are selected. This method is also referred to as *HiCS*, as it selects high-contrast subspaces.

In the work proposed in [308], the outliers are found only in these high-contrast subspaces, and the corresponding scores are combined. Thus, this approach decouples the sub-

space search as a generalized pre-processing approach from the outlier ranking of the individual data points. The approach discussed in [308] is quite interesting because of its pre-processing approach to finding relevant subspaces in order to reduce the irrelevant subspace exploration. Although the high-contrast subspaces are obtained using aggregation-based statistics, these statistics are only used as hints in order to identify multiple subspaces for greater robustness. The assumption here is that rare patterns are *statistically more likely* to occur in subspaces where there is significant non-uniformity and contrast. The final outlier score combines the results over different subspaces to ensure that at least a few relevant subspaces will be selected. The insight in the work of [308] is to combine *discriminative* subspace selection with the score aggregation of feature bagging in order to determine the relevant outlier scores. Therefore, the only difference from feature bagging is in how the subspaces are selected; the algorithms are otherwise identical. It has been shown in [308] that this approach performs better than the feature bagging method. Therefore, an overview of the *HiCS* method is as follows:

1. The first step is to select discriminative subspaces using an *Apriori*-like [37] exploration, which is described at the end of this section. These are *high-contrast subspaces*. Furthermore, the subspaces are also pruned to account for redundancy among them.

   - An important part of this exploration is to be able to *evaluate* the quality of the candidate subspaces during exploration. This is achieved by quantifying the contrast of a subspace.

2. Once the subspaces have been identified, an exactly similar approach to the feature bagging method is used. The LOF algorithm is executed after projecting the data into these subspaces and the scores are combined as discussed in section 5.2.3.

Our description below will therefore focus only on the first step of *Apriori*-like exploration and the corresponding quantification of the contrast. We will deviate from the natural order of presentation and first describe the contrast computation because it is germane to a proper understanding of the *Apriori*-like subspace exploration process.

Consider a subspace of dimensionality p, in which the dimensions are indexed as $\{1 \ldots p\}$ (without loss of generality). The conditional probability $P(x_1|x_2 \ldots x_p)$ for an attribute value x_1 is the same as its unconditional probability $P(x_1)$ for the case of uncorrelated data. High-contrast subspaces are likely to violate this assumption because of non-uniformity in data distribution. In our earlier example of the young Alzheimer patients, this corresponds to the unexpected rarity of the *combination* of youth and the disease. In other words $P(Alzheimer = 1)$ is likely to be very different from $P(Alzheimer = 1|Age \leq 30)$. The idea is that subspaces with such unexpected non-uniformity are more *likely* to contain outliers, although it is treated only as a weak hint for pre-selection of one of multiple subspaces. The approach in [308] generates candidate subspaces using an Apriori-like approach [37] described later in this section. For each candidate subspace of dimensionality p (which might vary during the Apriori-like exploration), it repeatedly draws pairs of "samples" from the data in order to estimate $P(x_i)$ and $P(x_i|x_1 \ldots x_{i-1}, x_{i+1} \ldots x_p)$ and test whether they are different. A "sample" is defined by (i) the selection of a particular attribute i from $\{1 \ldots p\}$ for testing, and (ii) the construction of a random rectangular region in p-dimensional space for testing. Because of the construction of a rectangular region for testing, each x_i refers to a 1-dimensional range of values (e.g., $Age \in (10, 20)$) in the ith dimension. The values of $P(x_i)$ and $P(x_i|x_1 \ldots x_{i-1}, x_{i+1} \ldots x_p)$ are computed in this random rectangular region. After M pairs of samples of $P(x_i)$ and $P(x_i|x_1 \ldots x_{i-1}, x_{i+1} \ldots x_p)$ have been drawn, it is

determined whether the independence assumption is violated with hypothesis testing. A variety of tests based on the Student's t-distribution can be used to measure the deviation of a subspace from the basic hypothesis of independence. This provides a measure of the non-uniformity of the subspace and therefore provides a way to measure the quality of the subspaces in terms of their propensity to contain outliers.

A bottom-up *Apriori*-like [37] approach is used to identify the relevant projections. In this bottom-up approach, the subspaces are continuously extended to higher dimensions for non-uniformity testing. Like *Apriori*, only subspaces that have sufficient contrast are extended for non-uniformity testing as potential candidates. The non-uniformity testing is performed as follows. For each candidate subspace of dimensionality p, a random rectangular region is generated in p dimensions. The width of the random range along each dimension is selected so that the 1-dimensional range contains $N \cdot \alpha^{(1/p)}$ points, where $\alpha < 1$. Therefore, the entire p-dimensional region is expected to contain $N \cdot \alpha$ points. The ith dimension is used for hypothesis testing, where the value of i is chosen at random from $\{1 \ldots p\}$. One can view the index i as the test dimension. Let the set of points in the intersection of the ranges along the remaining $(p-1)$ dimensions be denoted by S_i. The fraction of points in S_i that lies within the upper and lower bounds of the range of dimension i provides an estimate of $P(x_i|x_1 \ldots x_{i-1}, x_{i+1} \ldots x_d)$. The statistically normalized deviation of this value from the unconditional value of $P(x_i)$ is computed using hypothesis testing and it provides a deviation estimate for that subspace. The process is repeated multiple times over different random slices and test dimensions; then, the deviation values over different tests are averaged. Subspaces with large deviations are identified as high-contrast subspaces. At the end of the *Apriori*-like phase, an additional pruning step is applied to remove redundant subspaces. A subspace of dimensionality p is removed, if another subspace of dimensionality $(p+1)$ exists (among the reported subspaces) with higher contrast.

The approach decouples subspace identification from outlier detection and therefore all the relevant subspaces are identified up front as a preprocessing step. After the subspaces have been identified, the points are scored using the LOF algorithm in each such subspace. Note that this step is very similar to feature bagging, except that we are restricting ourselves to more carefully chosen subspaces. Then, the scores of each point across various subspaces are computed and averaged to provide a unified score of each data point. In principle, other combination functions like maximization can be used. Therefore, one can adapt any of the combination methods used in feature bagging. More details of the algorithm for selecting relevant subspaces are available in [308].

The *HiCS* technique is notable for the intuitive idea that statistical selection of relevant subspaces is more effective than choosing random subspaces. The main challenge is in discovering the high-contrast subspaces, because it is computationally intensive to use an *Apriori*-like algorithm in combination with sample-based hypothesis testing. Many straightforward alternatives exist for finding high-contrast subspaces, which might be worth exploring. For example, one can use the multidimensional Kurtosis measure discussed in section 1.3.1 of Chapter 1 in order to test the relevance of a subspace for high-dimensional outlier detection. This measure is simple to compute and also takes the interactions between the dimensions into account because of its use of the Mahalanobis distance.

5.2.8 Local Selection of Subspace Projections

The work in [402] uses *local* statistical selection of relevant subspace projections in order to identify outliers. In other words, the selection of the subspace projections is optimized to specific data points, and therefore the locality of a given data point matters in the

selection process. For each data point $\overline{X}$, a set of subspaces is identified, which are considered *high-contrast* subspaces from the perspective of outlier detection. However, this exploration process uses the high-contrast behavior as statistical *hints* in order to explore *multiple* subspaces for robustness, since a single subspace may be unable to completely capture the outlierness of the data point.

The *OUTRES* method [402] examines the density of lower-dimensional subspaces in order to identify relevant projections. The basic hypothesis is that for a given data point $\overline{X}$, it is desirable to determine subspaces in which the data is sufficiently non-uniformly distributed in its locality. In order to characterize the distribution of the locality of a data point, the work in [402] computes the local density of data point $\overline{X}$ in subspace S as follows:

$$den(S, \overline{X}) = |\mathcal{N}(\overline{X}, S)| = |\{\overline{Y} : dist_S(\overline{X}, \overline{Y}) \leq \epsilon\}| \tag{5.6}$$

Here, $dist_S(\overline{X}, \overline{Y})$ represents the Euclidean distance between data point $\overline{X}$ and $\overline{Y}$ in subspace S. This is the simplest possible definition of the density, although other more sophisticated methods such as kernel density estimation [496] are used in *OUTRES* in order to obtain more refined results. Kernel density estimation is also discussed in Chapter 4. A major challenge here is in comparing the subspaces of varying dimensionality. This is because the density of the underlying subspaces reduces with increasing dimensionality. It has been shown in [402], that it is possible to obtain comparable density estimates across subspaces of different dimensionalities by selecting the bandwidth of the density estimation process according to the dimensionality of the subspace.

Furthermore, the work in [402] uses statistical techniques in order to meaningfully compare different subspaces. For example, if the data is uniformly distributed, the number of data points lying within a distance ϵ of the data point should be regulated by the fractional volume of the data in that subspace. Specifically, the fractional parameter defines a binomial distribution characterizing the number of points in that volume, if that data were to be uniformly distributed. Of course, one is really interested in subspaces that deviate significantly from this behavior. The (local) relevance of the subspace for a particular data point $\overline{X}$ is computed using statistical testing. The two hypotheses are as follows:

- Hypothesis H_0: The local subspace neighborhood $\mathcal{N}(\overline{X}, S)$ is uniformly distributed.
- Hypothesis H_1: The local subspace neighborhood $\mathcal{N}(\overline{X}, S)$ is not uniformly distributed.

The Kolmogorov-Smirnoff goodness-of-fit test [512] is used to determine which of the aforementioned hypotheses is true. It is important to note that this process provides an idea of the *usefulness* of a subspace, and is used in order to enable a *filtering condition* for removing irrelevant subspaces from the process of computing the outlier score of a specific data point. A subspace is defined as relevant, if it passes the hypothesis condition H_1. In other words, outlier scores are computed using a combination of subspaces which *must* satisfy this relevance criterion. This test is combined with an ordered subspace exploration process in order to determine the relevant subspaces $S_1 \ldots S_k$. This exploration process will be described later in detail (cf. Figure 5.2).

In order to combine the point-wise scores from multiple *relevant* subspaces, the work in [402] uses the product of the outlier scores obtained from different subspaces. Thus, if $S_1 \ldots S_k$ are the different abnormal subspaces found for data point $\overline{X}$, and if $O(S_i, \overline{X})$ is its outlier score in subspace S_i, then the overall outlier score $OS(\overline{X})$ is defined as follows:

$$OS(\overline{X}) = \prod_i O(S_i, \overline{X}) \tag{5.7}$$

Algorithm *OUTRES*(Data Point: $\overline{X}$, Subspace: S);
begin
  for each attribute i not in S **do**
    if $S_i = S \cup \{i\}$ passes Kolmogorov-Smirnoff non-uniformity test **then**
    begin
      Compute $den(S_i, \overline{X})$ using Equation 5.6 or kernel density estimation;
      Compute $dev(S_i, \overline{X})$ using Equation 5.8;
      Compute $O(S_i, \overline{X})$ using Equation 5.9;
      $OS(\overline{X}) = O(S_i, \overline{X}) \cdot OS(\overline{X})$;
      $OUTRES(\overline{X}, S_i)$;
    end
end

Figure 5.2: The *OUTRES* algorithm

The details of the computation of $O(S_i, \overline{X})$ will be provided later, although the basic assumption is that *low scores* represent a greater tendency to be an outlier. The advantage of using the product over the sum in this setting is that the latter is dominated by the high scores, as a result of which a few subspaces containing normal behavior will dominate the sum. On the other hand, in the case of the product, the outlier behavior in a small number of subspaces will be greatly magnified. This is particularly appropriate for the problem of outlier detection. It is noteworthy that the product-wise combination can also be viewed as the sum of the logarithms of the scores.

In order to define the outlier score $O(S_i, \overline{X})$, a subspace is considered significant for particular objects only if its density is at least two standard deviations less than the mean value. This is essentially a condition for that subspace to be considered deviant. Thus, the deviation $dev(S_i, \overline{X})$ of the data point $\overline{X}$ in subspace S_i is defined as the ratio of the deviation of the density of the object from the mean density in the neighborhood of $\overline{X}$, divided by two standard deviations.

$$dev(S_i, \overline{X}) = \frac{\mu - den(S_i, \overline{X})}{2 \cdot \sigma} \tag{5.8}$$

The values of μ and σ are computed over data points in the neighborhood of $\overline{X}$ and therefore this computation provides a *local* deviation value. Note that deviant subspaces will have $dev(S_i, \overline{X}) > 1$. The outlier score of a data point in a subspace is the ratio of the density of the point in the space to its deviation, if it is a deviant subspace. Otherwise the outlier score is considered to be 1, and it does not affect the overall outlier score in the product-wise function in Equation 5.7 for combining the scores of data point $\overline{X}$ from different subspaces. Thus, the outlier score $O(S_i, \overline{X})$ is defined as follows:

$$O(S_i, \overline{X}) = \begin{cases} \frac{den(S_i, \overline{X})}{dev(S_i, \overline{X})} & \text{if } dev(S_i, \overline{X}) > 1 \\ 1 & \text{otherwise} \end{cases} \tag{5.9}$$

The entire recursive approach of the *OUTRES* algorithm (cf. Figure 5.2) uses the data point $\overline{X}$ as input, and therefore the procedure needs to be applied separately *for scoring each candidate data point*. In other words, this approach is inherently an instance-based method (like nearest-neighbor detectors), rather than one of the methods that selects subspaces

up front in a decoupled way (like feature bagging or *HiCS*). An observation in [402] is that subspaces that are either very low dimensional (e.g., 1-dimensional subspaces) or very high dimensional are not very informative for outlier detection. Informative subspaces can be identified by careful attribute-wise addition to candidate subspaces. In the recursive exploration, an additional attribute is included in the subspace for statistical testing. When an attribute is added to the current subspace S_i, the non-uniformity test is utilized to determine whether or not that subspace should be used. If it is not relevant, then the subspace is discarded. Otherwise, the outlier score $O(S_i, \overline{X})$ in that subspace is computed for the data point, and the current value of the outlier score $OS(\overline{X})$ is updated by multiplying $O(S_i, \overline{X})$ with it. Since the outlier scores of subspaces that do not meet the filter condition are set to 1, they do not affect the density computation in this multiplicative approach. The procedure is then recursively called in order to explore the next subspace. Thus, such a procedure potentially explores an exponential number of subspaces, although the real number is likely to be quite modest because of pruning. In particular, the non-uniformity test prunes large parts of the recursion tree during the exploration. The overall algorithm for subspace exploration for a given data point $\overline{X}$ is illustrated in Figure 5.2. Note that this pseudocode assumes that the overall outlier score $OS(\overline{X})$ is like a global variable that can be accessed by all levels of the recursion and it is initialized to 1 before the first call of *OUTRES*. The initial call to *OUTRES* uses the empty subspace as the argument value for S.

5.2.9 Distance-Based Reference Sets

A distance-based method for finding outliers in lower-dimensional projections of the data was proposed in [327]. In this approach, instead of trying to find local subspaces of abnormally low density over the whole data, a local analysis is provided specific to each data point. For each data point $\overline{X}$, a *reference set* of points $S(\overline{X})$ is identified. The reference set $S(\overline{X})$ is generated as the top-k closest points to the candidate with the use of shared nearest-neighbor distances [287] (see section 4.3.3).

After this reference set $S(\overline{X})$ has been identified, the relevant subspace for $S(\overline{X})$ is determined as the set $Q(\overline{X})$ of dimensions in which the variance is small. The specific threshold on the variance is set to a user-defined fraction of the average dimension-specific variance of the points in $S(\overline{X})$. Thus, this approach analyzes the statistics of individual dimensions independently of one another during the crucial step of subspace selection. The Euclidean distance of $\overline{X}$ is computed to the mean of the reference set $S(\overline{X})$ *in the subspace defined by* $Q(\overline{X})$. This is denoted by $G(\overline{X})$. The value of $G(\overline{X})$ is affected by the number of dimensions in $Q(\overline{X})$. The *subspace outlier degree* $SOD(\overline{X})$ of a data point is defined by normalizing this distance $G(\overline{X})$ by the number of dimensions in $Q(\overline{X})$:

$$SOD(\overline{X}) = \frac{G(\overline{X})}{|Q(\overline{X})|}$$

The approach of using the variance of *individual* dimensions for selecting the subspace set $Q(\overline{X})$ is a rather naive generalization derived from subspace clustering methods, and is a rather questionable design choice. This is because the approach completely ignores the interactions among various dimensions. In many cases, such as the example of the young Alzheimer patient discussed earlier, the unusual behavior is manifested in *the violations of dependencies among* dimensions rather than the variances of the individual dimensions. Variances of individual dimensions tell us little about the dependencies among them. Many

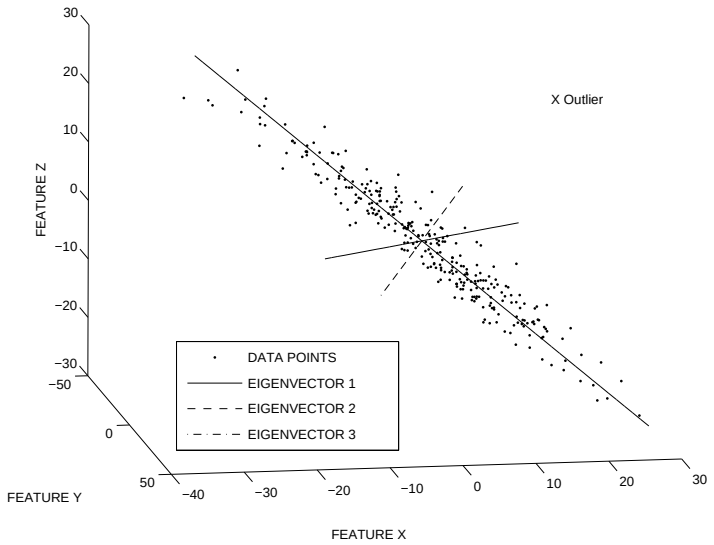


Figure 5.3: The example of Figure 3.4 re-visited: Global PCA can discover outliers in cases, where the entire data is aligned along lower dimensional manifolds.

other insightful techniques like *HiCS* [308, 402], which use biased subspace selection, almost always use the dependencies among dimensions as a key selection criterion.

Another problem is the use of a *single* subspace to score outliers. Since the subspace selection criterion ignores dependencies, it can cause the removal of relevant dimensions. In the interesting cases, where the number of relevant dimensions is limited, the negative effects of removing a single relevant dimension can be even more drastic than keeping many irrelevant dimensions. The particularly problematic factor in using a single subspace is that if a mistake is made in subspace selection, there is virtually no chance of recovering from the mistake. In general, these types of methods are almost always outperformed by the various subspace ensemble methods discussed in this chapter (feature bagging [344], rotated bagging [32], *RS-Hash* [476], and isolation forests [367]).

5.3 Generalized Subspaces

Although axis-parallel methods are effective for finding outliers in most real settings, they are not very useful for finding outliers in cases where the points are aligned along arbitrary lower-dimensional manifolds of the data. For example, in the case of Figure 5.4, no 1-dimensional feature from the 2-dimensional data can find the outliers. On the other hand, it is possible to find *localized* 1-dimensional correlated subspaces so that most of the data aligns along these localized 1-dimensional subspaces, and the remaining deviants can be classified as outliers. Although this particular data set seems to be relatively easy for outlier detection because of its low dimensionality, this problem can become more challenging with increasing dimensionality.

These algorithms are generalizations of the following two classes of algorithms:

- The PCA-based linear models discussed in Chapter 3 find the *global* regions of correlation in the data. For example, in the case of Figure 5.3, the outliers can be effectively

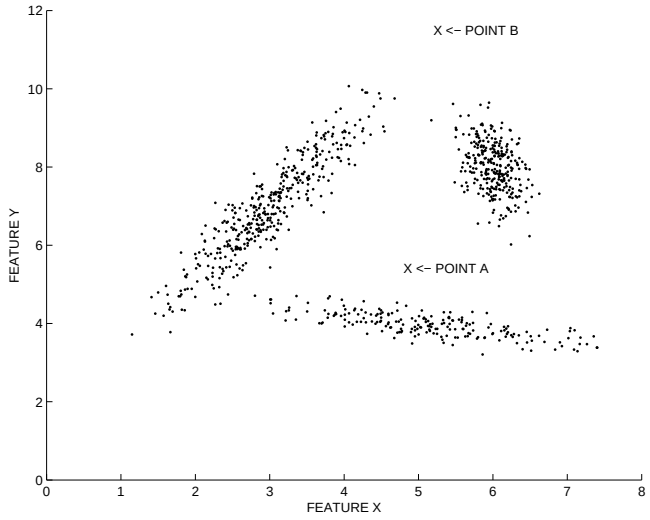


Figure 5.4: The example of Figure 2.9 revisited: Outliers are best discovered by determining deviations from local PCA-based clusters. Neither axis-parallel subspace outliers nor global-PCA can capture such clusters.

identified by determining these global directions of correlation. However, no such *global* directions of correlation exist in Figure 5.4.

- The axis-parallel subspace outliers discussed earlier in this chapter can find deviants, when the data is naturally aligned along low dimensional axis-parallel subspace clusters. However, this is not the case in Figure 5.4, in which the data is aligned along arbitrary directions of correlation.

The goal in generalized subspace analysis is to combine the ideas in these two types of algorithms. In other words, it is desired to determine the arbitrarily oriented subspaces *simultaneously* with outlier discovery. In the following, we will discuss several methods for finding such generalized subspaces. We will also discuss some methods for discovering *nonlinear* subspaces, in which the data is distributed along *local nonlinear manifolds*.

5.3.1 Generalized Projected Clustering Approach

This problem can be partially addressed with the use of generalized projected clustering methods, where the clusters are identified in arbitrarily aligned subspaces of the data [7]. The method discussed in [7] has a built-in mechanism in order to determine the outliers *in addition to* the clusters. Such outliers are naturally data points that do not align with the clusters. However, the approach is not particularly optimized for finding the outliers, because the primary purpose of the method is to determine the clusters. The outliers are discovered as a side-product of the clustering algorithm, rather than as the primary goal. Therefore, the approach may sometimes discover the weaker outliers, which correspond to the noise in the data. Clearly, methods are required for properly distinguishing between the strong and weak outliers by using an outlier scoring mechanism to distinguish between various points. The simplest approach is to compute the local Mahalanobis distance of every candidate outlier to each cluster centroid. The computation of the local Mahalanobis

distance of a point to a cluster centroid uses only the mean and covariance matrix of that cluster. The computation of the local Mahalanobis distance is described in section 4.2 of Chapter 4 (cf. Equation 4.2). For any given point, its smallest Mahalanobis distance (i.e., distance to its nearest centroid) is reported as its outlier score.

To improve robustness, one should use randomized methods to cluster the data in multiple ways, and average the point-wise scores from the different models. It is *very important* to combine scores from multiple subspaces, because the individual subspaces discovered using subspace clustering do not tell us much about the relevant subspaces for outlier detection. However, the outliers discovered using clustering often inherit the properties of the underlying clusters when an ensemble method is used. Therefore, using clusters in subspaces of the data yields outliers that are subspace sensitive. The work in [406] provides a specific example in the axis-parallel setting, where it shows that the combination of scores from multiple clusterings with the use of the *Multiple-Proclus* method greatly improves over the performance of a single application of the clustering algorithm.

Many other generalized projected clustering methods can be used and a detailed survey may be found in [23] (Chapter 9). Many of these methods are quite efficient. One advantage of this approach is that once the clusters have been identified up front, the scoring process is very efficient. Furthermore, the model-building process usually requires less than $O(N^2)$ time, which is required by most distance-based detectors.

5.3.2 Leveraging Instance-Specific Reference Sets

In order to determine the outliers that are optimized to the locality of a particular data point, it is critical to determine localized subspaces that are optimized to the candidate data point $\overline{X}$ being scored. The determination of such subspaces is non-trivial, since it often cannot be inferred from locally aggregate properties of the data, for detecting the behavior of *rare* instances. A method was recently proposed in [328] for finding outliers in generalized subspaces with the use of reference sets. The main difference from earlier generalized subspace clustering methods is that local reference sets are *specific to the various data points*, whereas clusters provide a fixed set of reference sets that are used to score all points. The price of this flexibility is that the running time of finding *each* point-specific reference set is $O(N)$. Therefore, the approach requires $O(N^2)$ time for scoring.

For a given data point $\overline{X}$, this method finds the full-dimensional k-nearest neighbors of $\overline{X}$. This provides a reference set S with mean vector $\overline{\mu}$. The PCA approach of Chapter 3 is applied to the covariance matrix $\Sigma(S)$ of the *local* reference set S in order to determine the key eigenvectors $\overline{e_1} \ldots \overline{e_d}$, in increasing order of variance, with corresponding eigenvalues $\lambda_1 \leq \lambda_2 \ldots \leq \lambda_d$. The discussion in section 3.3 of Chapter 3 performs these same steps [493] except that they are performed on a *global* basis, rather than on a local reference set S. Even if all d dimensions are included, it is possible to create a normalized outlier score of a data point $\overline{X}$ to the centroid $\overline{\mu}$ of the data with the use of local eigenvalue scaling, as discussed in Chapter 3:

$$Score(\overline{X}) = \sum_{j=1}^{d} \frac{|(\overline{X} - \overline{\mu}) \cdot \overline{e_j}|^2}{\lambda_j} \tag{5.10}$$

As discussed in section 2.2.2.2 of Chapter 2, this can be approximately modeled as a χ^2 distribution with d degrees of freedom for each data point, and the outlier scores of the different data points can be reasonably compared to one another. Such an approach is used in [493] in the context of global data analysis. The survey paper of Chandola *et al.* [125]

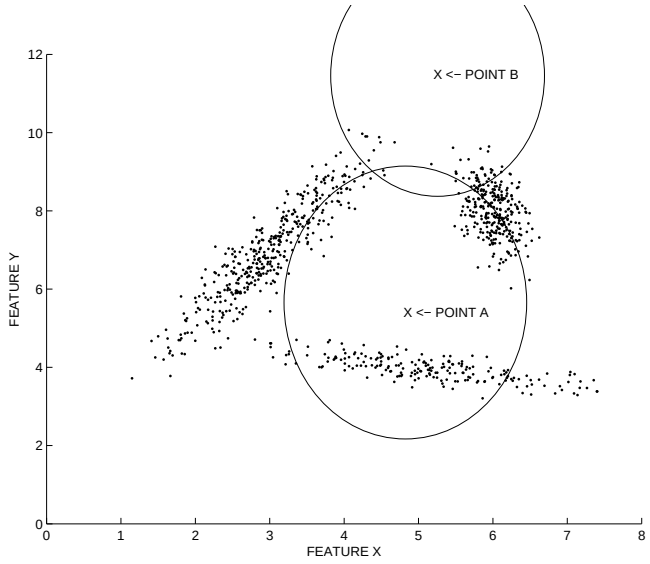


Figure 5.5: Local reference set may sometimes contain points from multiple generating mechanisms

provides a simpler exposition. Note that this approach can be viewed as a local version of soft PCA and it is not necessary to use subspaces.

The work in [328] is different from this basic approach in that it emphasizes the pre-selection of a subset of the eigenvectors for scoring. This is primarily because the work is positioned as a generalized subspace outlier detection method. As we will discuss later, this is not necessarily desirable.

The δ eigenvectors[3] with the smallest eigenvalues are selected for score computation. Correspondingly, the pruned score is defined on the basis of the smallest $\delta \leq d$ eigenvectors:

$$Score(\overline{X}, \delta) = \sum_{j=1}^{\delta} \frac{|(\overline{X} - \overline{\mu}) \cdot \overline{e_j}|^2}{\lambda_j} \tag{5.11}$$

How should the value of δ be fixed for a particular data point $\overline{X}$? The score is a χ^2-distribution with δ-degrees of freedom. It was observed in [328] that the value of δ can be parameterized, by treating the χ^2 distribution as a special case of the Γ distribution.

$$Score(\overline{X}, \delta) \sim \Gamma(\delta/2, 2)$$

The optimal value of δ is selected specifically for each data point, by selecting the value of δ in order to determine the maximal unlikely deviation based on this model. This is done by using the cumulative density function of the aforementioned distribution. While this value can be directly used as an outlier score, it was also shown in [328], how this score may be converted into a more intuitive probability value.

This approach, however, has several weaknesses. These weaknesses arise from the lack of robustness in using a single subspace as relevant set of dimensions, from the way in which

[3]The work in [328] uses δ as the number of *longest* eigenvectors, which is only a notational difference, but is noted here to avoid confusion.

the reference set is constructed, and from how the hard pruning methodology is used in score computation. We discuss each of these issues in detail below:

- A *single subspace* has been used by this approach for finding the outliers with the use of the local reference set S. If the local reference set S is not accurately determined, then this will not provide the proper directions of local correlation. The use of a single subspace is risky, especially with the use of weak aggregation-based hints, because it is often possible to unintentionally remove relevant subspaces. This can have drastic effects. The use of multiple subspaces may be much more relevant in such scenarios, such as the methods proposed in [32, 308, 344, 367, 402, 406].

- There is an inherent circularity in identifying the reference set with the use of full-dimensional k-nearest neighbor distances, especially if the distances are not meaningfully defined in full dimensionality. The choice of points in the reference set and the choice of the subspace clearly impact each other in a circular way. This is a classical "chicken-and-egg" problem in subspace analysis, which was first pointed out in [5]. The analysis in such cases needs to be *simultaneous* rather than *sequential*. As is well known, the most robust techniques for handling circularity in virtually all problem domains (e.g., projected clustering methods) use iterative methods, so that the point-specific and dimension-specific aspects of the problem are able to interact with one another. This is however, not the case in [328], in which a sequential analysis is used.

  In particular, it may happen that many locally irrelevant features may be used during the determination of the local reference set, when full-dimensional distances are used. This set could therefore contain data points from multiple generating mechanisms, as illustrated in Figure 5.5. When the number of irrelevant features is unknown, a specific number of points in the reference set will not be able to avoid this problem. The use of a smaller reference set size can reduce the chance of this happening to some extent, but can never guarantee it, especially when many irrelevant features are used. On the other hand, reducing the reference set size can also result in a correlation hyperplane, whose eigenvalue statistics overfit an artificially small set of reference points. In fact, the real challenge in such problems is in properly selecting the reference set; this issue has been trivialized by this approach.

- It is not necessary to select a particular set of eigenvectors in a hard way, since the eigenvalues in the denominator of Equation 5.10 already provide a soft weighting to their relative importance (or relevance). For example, if for a large value of λ_i, a data point shows even larger deviations along that direction, such an outlier would either be missed by dimension pre-selection, or would include other less relevant dimensions. An example is the outlier B in Figure 5.5, which is aligned along the longer eigenvector, and therefore the longest eigenvector is the *most informative* about its outlier behavior. In particular, the method of selecting the δ smallest eigenvectors implicitly assumes that the relevance of the attributes are ordered by eigenvalue magnitude. While this may generally be true for aggregation-based clustering algorithms, it is very often not true in outlier analysis because of the unusual nature of outliers. The possibility of outliers aligning along long eigenvectors is not uncommon at all, since two highly correlated attributes may often show highly deviant behavior of a similarly correlated nature. This example also shows, how *brittle* the rare nature of outlier analysis is to aggregation-based measures. This is because of the varying causes of rarity, which cannot be fully captured in aggregation statistics. This observation exemplifies the fact that straightforward generalizations of subspace selection methods from clustering

(based on aggregates), are often not appropriate or optimized for (the rare nature of) outlier analysis. One advantage of using all the dimensions is that it reduces to a local Mahalanobis distance with the same dimensionality, and allows better comparability in the scores across different outliers. In such cases, intuitive probability values may be derived more simply from the $\chi^2(d)$ distribution.

The high-dimensional case is an extremely difficult one, and it is understandable that no given method will be able to solve these problems perfectly.

5.3.3 Rotated Subspace Sampling

The rotated subspace sampling method was recently proposed in [32] as an ensemble method, which improves over feature bagging [344]. This approach is also referred to as *rotated bagging*. Just as feature bagging is designed for discovering outliers in axis-parallel subspaces, the rotated bagging method is designed for discovering outliers in generalized subspaces. As in the case of feature bagging, this approach is an ensemble method and it can use any off-the-shelf outlier detection algorithm (e.g., LOF) as the *base detector*.

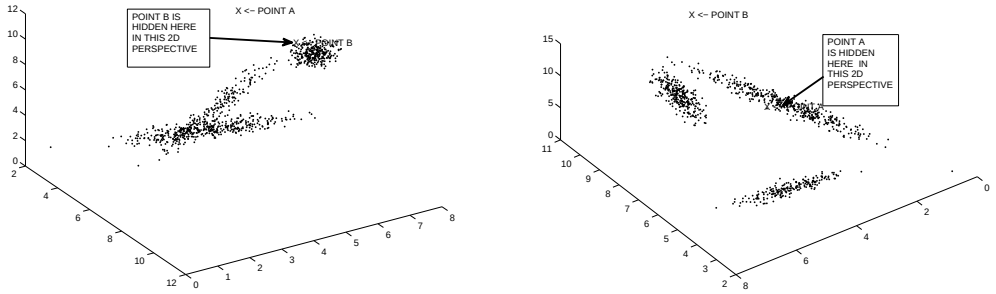
The basic idea in rotated bagging is to sample randomly rotated subspaces in lower-dimensional space, and score each point in this low-dimensional space. The scores from various subspaces can be combined to provide the final result. In particular, the approach uses subspaces of dimensionality $r = 2 + \lceil \sqrt{d}/2 \rceil$, which is much lower the typical dimensionality of the subspace used in feature bagging. This is because the axis rotation enables the capturing of information from all dimensions to varying degrees. The ability to use lower-dimensional projections is also useful for inducing diversity and thereby improving the quality of the overall *ensemble* score. The rotated bagging algorithm works as follows:

1. Determine a randomly rotated axis system in the data.
2. Sample $r = 2 + \lceil \sqrt{d}/2 \rceil$ directions from rotated axis system. Project data along these r directions.
3. Run the base outlier detector on projected data and store the scores of each point.

The component scores can be combined by either (a) using the average score of a point across different projections, or (b) using the maximum score of a point across different projections. Other combination functions are discussed in Chapter 6. It is important to use standardization on the scores before the combination.

In order to determine $r = 2 + \lceil \sqrt{d}/2 \rceil$ randomly rotated mutually orthogonal directions, a $d \times r$ random matrix Y is generated, such that each value in the matrix is uniformly distributed in $[-1, 1]$. Let the tth column of Y be denoted by $\overline{y_t}$. Then, the r random orthogonal directions $\overline{e_1} \ldots \overline{e_r}$ are generated using a straightforward Gram-Schmidt orthogonalization of $\overline{y_1} \ldots \overline{y_r}$ as follows:

1. $t = 1$; $\overline{e_1} = \frac{\overline{y_1}}{|\overline{y_1}|}$
2. $\overline{e_{t+1}} = \overline{y_{t+1}} - \sum_{j=1}^{t} (\overline{y_{t+1}} \cdot \overline{e_j}) \overline{e_j}$
3. Normalize $\overline{e_{t+1}}$ to unit norm.
4. $t = t + 1$
5. if $t < r$ go to step 2



(a) Point 'A' is exposed and 'B' is hidden in this 2d view

(b) Point 'B' is exposed and 'A' is hidden in this 2d view

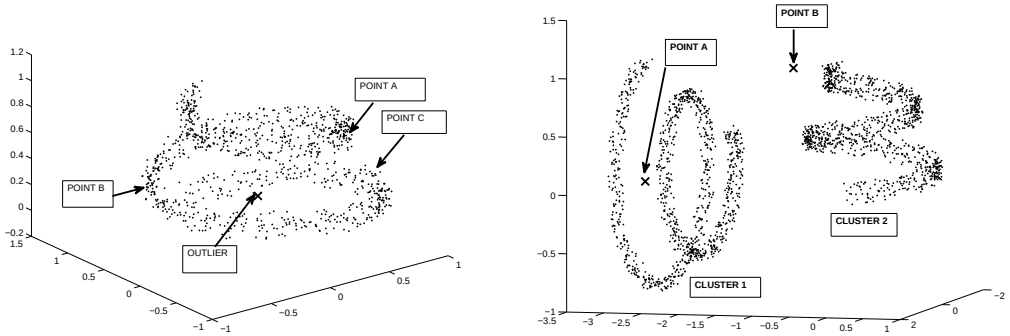
Figure 5.6: In this 3-dimensional data set, points 'A' and 'B' are exposed in different 2-dimensional views (projections). However, the averaging or maximization score-combination will expose both 'A' and 'B.'

Let the resulting $d \times r$ matrix with columns $\overline{e_1} \ldots \overline{e_r}$ be denoted by E. The $N \times d$ data set D is transformed and projected to these orthogonal directions by computing the matrix product DE, which is an $N \times r$ matrix of r-dimensional points. The results in [32] show that the approach improves over feature bagging. Further improvements may be obtained by combining it with other ensemble techniques. Rotated bagging uses a more general model to describe the outliers than feature bagging in terms of *arbitrary subspaces* and therefore using an ensemble is even more important. It is often difficult to discover a particular local subspace that is relevant to a particular data point. The great power of ensembles lies in the fact that an averaged ensemble combination of many global subspace selections is often able to discover the locally relevant subspaces. In other words, the ensemble is inherently more powerful than its individual members. This point can be explained from the perspective of how such types of averaged models use ensembles to reduce representational bias (cf. section 6.4.3.1 of Chapter 6). Furthermore, the maximization combination function often does even better in terms of reducing representational bias.

An example of a 3-dimensional data set is illustrated in Figure 5.6. Here, we have shown different 2-dimensional views of the data. It is clear that in the case of Figure 5.6(a), outlier 'A' is exposed, whereas in the case of Figure 5.6(b), outlier 'B' is exposed. However, if one were to use the average or maximization combination of the scores obtained by running the detector on these two views, *both* points 'A' and 'B' might be scored highly in the combination (and therefore discovered as outliers). This is an example of how ensemble methods overcome the *representational bias* in individual detectors in order to provide a more general model. Another example of this power of ensemble methods to overcome representational bias is provided in Figure 6.4 of Chapter 6. An ensemble-centric description of rotated bagging is also provided in section 6.4.4 of Chapter 6.

5.3.4 Nonlinear Subspaces

The most difficult case of subspace outlier detection occurs in cases in which the manifolds exist in lower-dimensional subspaces of arbitrary shape. Examples of such patterns in the



(a) Global data distribution (b) Local data distributions

Figure 5.7: Arbitrarily shaped clusters can exist in lower-dimensional manifolds (Revisiting Figure 4.3 of Chapter 4)

data are illustrated in Figure 5.7. This figure is the same as Figure 4.3 of Chapter 4. In fact, the clustering method proposed in section 4.2.1 provides one of the ways in which outliers can be discovered in such nonlinear settings. Since the discussion in this section is roughly based on the framework established in section 4.2.1 of Chapter 4, the reader is advised to revisit that section before proceeding further.

The arbitrary shape of the clusters presents a number of unique challenges. There are several cases in Figures 5.7(a) and (b), in which the outliers are placed in the interior of non-convex patterns. Without an appropriate *local nonlinear transformation*, it becomes more challenging to discover such outliers. This extraction is achieved with spectral methods, which can be viewed as a special class of kernel methods that is friendly to discovering clusters of arbitrary shape.

As in the case of section 4.2.1, a spectral embedding of the data is extracted. This is achieved by constructing a k-nearest neighbor graph and computing its top eigenvectors. Because of the way in which the spectral embedding is constructed, it already adapts itself to local nonlinear patterns in the data, and Euclidean distances can be used on the transformed data set. One difference from the method of section 4.2.1 is that an additional step is used to reduce the noise in the spectral embedding [475]. The main problem in spectral embedding is caused by the presence of bad edges in the neighborhood graph, which connect disjoint clusters. Therefore, the similarity graph needs to be corrected to improve the quality of the embedding. In particular, an iterative approach is used to construct the spectral embedding in which the similarity matrix (used to construct the embedding) is corrected for erroneous computations between pairs of points. The spectral embedding is used to correct the similarity matrix, and the corrected similarity matrix is used to construct the spectral embedding. The basic idea here is that the spectral embedding from a similarity matrix S will itself create a embedding from which a new similarity matrix S' can be constructed. In the new representation, weakly related points move away from each other on a relative basis (like points A and C in Figure 5.7(a)), and strongly related points move towards each other. Therefore, the Hadamard product of S and S' is used to adjust S:

$$S \Leftarrow S \circ S' \tag{5.12}$$

This has the effect of correcting the original similarity matrix S to de-emphasize noisy

similarities on a *relative* basis. The new similarity matrix is used to again create a new embedding. These steps are repeated for a small number of iterations and the final embedding is used to score the points. This process is described in detail in [475]. Each of the new dimensions of the embedding are normalized to unit variance. A k-nearest neighbor detector is executed on the embedding in order to return the final outlier score. By using a k-nearest neighbor detector in the transformed space, one is effectively using a data-dependent distance function that is sensitive to the nonlinear subspace patterns in the original space. Although the work in [475] does not discuss it, the approach can be strengthened with the use of averaged scores from multiple clusterings with different parameters.

5.3.5 Regression Modeling Techniques

It is also possible to use regression models for subspace outlier detection by identifying points violating attribute-wise dependencies (e.g., the young Alzheimer patient discussed earlier). The basic idea is to decompose a d-dimensional unsupervised outlier detection problem into a set of d regression modeling problems. One can predict each attribute with the remaining $(d-1)$ attributes using an off-the-shelf regression model. For each instance, the squared errors of predicting the various attributes are aggregated to create the outlier score. This approach is discussed in detail in section 7.7 of Chapter 7. When certain types of base detectors like random forests are used, the approach can be interpreted as a local subspace outlier detection method. These connections are discussed in section 7.7.

5.4 Discussion of Subspace Analysis

The major reason for difficulty in high-dimensional outlier analysis is because of the masking effects of the locally noisy and irrelevant nature of some of the dimensions, which is often also manifested in the concentration of distances. It has been claimed in a recent survey [620] that the literature only focuses on the distance-concentration issue and "abstains from" discussing the impact of *locally relevant dimensions* caused by differential generating mechanisms (thereby claiming the latter as a new insight of the survey [620]). This is an incorrect and particularly surprising assertion, because *both* the aspects of local feature selection (relevance) and distance concentration have been studied extensively in the literature together with their connections to one another. The original work in [4] (and virtually every subsequent work [308, 344, 402]) provides a pictorial illustration (similar to Figure 5.1) and a fairly detailed discussion of how (locally) irrelevant attributes mask outliers in different feature-specific views of the data. In the context of such a pictorial illustration, the following is stated in [4] about locally irrelevant attributes:

> "Thus, by using full dimensional distance measures it would be difficult to determine outliers effectively because of the averaging behavior of the noisy and irrelevant dimensions. Furthermore, it is impossible to prune off specific features a priori, since different points may show different kinds of abnormal patterns, each of which use different features or views."

The ineffectiveness of *global* feature selection in high-dimensional data in fact forms the motivating reason for subspace analysis, which can be considered a *local* feature selection method, or a *local* dimensionality reduction method [7, 121]. These connections of local subspace analysis to the ineffectiveness of global feature selection in high-dimensional data were explicitly discussed in detail in the motivational discussion of one of the earliest works

on subspace analysis [5, 263]. At this point, these results are well known and established[4] wisdom. While it is possible to reduce the distance concentration effects by carefully calibrating the fraction of informative dimensions, such cases are (usually) not interesting for subspace analysis.

Although it is most definitely true that noisy and irrelevant attributes mask the outliers, the observation is certainly not new, and the two factors of distance concentration and local feature relevance are closely related. In fact, even the experimental simulations in the survey [620] show that concentration effects tend to co-occur in settings where there are too many irrelevant attributes. This is, of course, not a hard rule; nevertheless, it is a significant issue for distance-based detectors. The interesting cases for subspace analysis (typically) show some levels of both properties. Even limited levels of distance concentration impact the effectiveness of full-dimensional distance-based algorithms, and this impact is therefore important to examine in outlier analysis. It should be noted that noisy and irrelevant attributes are more likely to lead to concentration of distances. For example, for the case of uniformly distributed data, where all attributes are noisy, the concentration effect is extreme, and an outlier deviating along *a relatively small number of dimensions* will be hard to discover by full-dimensional methods. In such cases, from a full-dimensional distance-based or density-based perspective, all data points have almost equally good outlier scores, and this can be equivalently understood in terms of *either* locally irrelevant features or distance concentration effects. Of course, real data sets are not uniformly distributed, but *both* irrelevant features and concentration effects are present to varying degrees in different data sets. The general assumption for subspace analysis is that the addition of more dimensions often does not add *proportionally* more information for a particular outlier. The challenging outliers are often defined by the behavior of a small number of dimensions, and when the point-specific information does not increase substantially with data dimensionality, even modest concentration effects will have a negative impact on full-dimensional algorithms. The more the number of irrelevant attributes there are, the more erroneous the computations for full-dimensional distance-based methods are likely to be. An extreme example at the other end of the spectrum is where an outlier shows informative and deviant behavior in every dimension, and therefore outlier characteristics grow *stronger* with increasing dimensionality. However, in this rather uninteresting case, since the outlier shows *both* many relevant features *and* also typically does not conform to the distance concentration behavior of the remaining data, a trivial full-dimensional distance-based algorithm would discover it easily in most cases. In general, cases where the informative dimensions also increase significantly with data dimensionality, are not as interesting for subspace analysis because the full-dimensional masking behavior becomes less prominent in this easier case. Subspace analysis does not exclude the possibility that the more obvious deviants may also be found by full-dimensional analysis.

There are many high-dimensional data sets, where one can perform effective outlier detection by simpler full-dimensional algorithms. The effectiveness of a particular algorithm on a full-dimensional data set depends on the application scenario and the way in which the features are extracted. If the features are extracted with a particular anomaly detection scenario in mind, then a full-dimensional algorithm is likely to work well. In spite of this fact, subspace analysis will often discover outliers that are not easily found by other full-dimensional algorithms. Indeed, subspace analysis should not be viewed as a stand-alone method, but inherently as an ensemble method to improve the performance of various types

[4]Some of the earliest methods even refer to these classes of techniques as local dimensionality reduction [121] in order to emphasize the enhanced and differential local feature selection effect, which arises as a result of different generating mechanisms.

of base detectors. Whether one develops a specific base method for high-dimensional outlier detection (like subspace histograms) or whether one wraps it around existing detectors (like feature and rotated bagging), there are clear benefits to incorporating this principle in outlier detection. Furthermore, subspace analysis provides useful insights about the causality of the anomaly; one can use the relevant local subspaces to extract an understanding about the specific combination of relevant features. A simple example is the case of the young Alzheimer patient discussed earlier in this chapter.

Outliers, by their very rare nature, may often be hidden in small combinations of dimensions in a high-dimensional data set. Subspace analysis is particularly interesting for such scenarios. On the other hand, when more dimensions do add (significantly) more information, then this becomes an easy case for analysis, which no longer remains interesting. In more difficult cases, the vast majority of noisy dimensions make it difficult to discriminate between data points from a density-based or data-sparsity perspective. Subspace analysis works particularly well in these cases when the outliers exist in a modest number of locally relevant dimensions. This observation also points to a particularly difficult case in which the number of locally irrelevant dimensions increases with data dimensionality, and a very large number of dimensions also continue to be *weakly* relevant (but not strongly relevant). This situation often occurs in data sets containing thousands of dimensions. Such data sets remain an unsolved case for all classes of full-dimensional and subspace methods.

To summarize, the following principles need to be kept in mind while designing subspace methods:

- A direct exploration of rare regions is possible, though it is computationally challenging because of combinatorial explosion [4].
- Aggregation-based methods provide only weak hints about the relevant subspaces. The main power of these methods lies only in the ensemble-centric setting by combining the results from different subspaces.
- The individual component of an ensemble should be designed with efficiency considerations in mind. This is because efficient components enable the practical use of a larger number of components for better accuracy.

One interesting observation is that even when weak base detectors are used, combining them often leads to very strong results. The success of methods like feature bagging, subspace histograms, and rotated bagging is based on this fact. Note that in each case, the underlying base detector is not particularly strong; yet the final outlier detection results are very effective. Recent advancements in the field of high-dimensional outlier detection and ensemble analysis have been very significant. In spite of these advancements, many high-dimensional data sets continue to remain a challenge for outlier analysis.

5.5 Conclusions and Summary

Subspace methods for outlier detection are used in cases, where the outlier tendency of a data point is diluted by the noise effects of a large number of locally non-informative dimensions. In such cases, the outlier analysis process can be sharpened significantly by searching for subspaces in which the data points deviate significantly from the normal behavior. The most successful methods identify multiple relevant subspaces for a candidate outlier, and then combine the results from different subspaces in order to create a more robust ensemble-based ranking.

Outlier analysis is the most difficult problem among all classes of subspace analysis problems. This difficulty arises out of the rare nature of outliers, which makes direct statistical analysis more difficult. Since subspace analysis and local feature selection are related, it is noteworthy that even for global feature selection, there are few known methods for outlier analysis, as compared to clustering and classification algorithms. The reason is simple: enough statistical evidence is often not available for the analysis of rare characteristics. Robust statistics is all about *more* data, and outliers are all about *less* data and statistical non-conformity with most of the data! Regions and subspaces containing statistical conformity tell us very little about the complementary regions of non-conformity in the particular case of high-dimensional subspace analysis, since the *potential* domain of the latter is much larger than the former. In particular, a local subspace region of the greatest aggregate conformity does not necessarily reveal anything about the rare point with the greatest statistical non-conformity.

Although many recent ensemble methods for subspace analysis have shown great success, a particularly difficult case is one in which a large number of dimensions are weakly relevant (but not strongly relevant), and even more dimensions are locally irrelevant. These cases often occur in data sets containing thousands of dimensions, and remain unsolved by existing methods. While it is doubtful that the more difficult variations of the problem will ever be fully solved, or will work completely in all situations, the currently available techniques work in many important scenarios. There are many merits in being able to design such methods, because of the numerous insights they can provide in terms of identifying the causes of abnormality. The main challenge is that outlier analysis is so brittle, that it is often impossible to make confident assertions about inferences drawn from aggregate data analysis. The issue of efficiency seems to be closely related to that of effectiveness in high-dimensional outlier analysis. This is because the search process for outliers is likely to require exploration of multiple local subspaces of the data in order to ensure robustness. With increasing advances in the computational power of modern computers, there is as yet hope that this area will become increasingly tractable for analysis.

5.6 Bibliographic Survey

In the context of high-dimensional data, two distinct lines of research exist, one of which investigates the *efficiency* of high-dimensional outlier detection [58, 219, 557], and the other investigates the more fundamental issue of the *effectiveness* of high-dimensional outlier detection [4]. Unfortunately, the distinction between these two lines of work is sometimes blurred in the literature, even though these are clearly different lines of work with very different motivations. It should be noted that the methods discussed in [58, 219, 557] are all *full-dimensional methods*, because outliers are defined on the basis of their full-dimensional deviation. Although the method of [557] uses projections for indexing, this is used only as an approximation to improve the efficiency of the outlier detection process.

In the high-dimensional case, the efficiency of (full-dimensional) outlier detection also becomes a concern, because most outlier detection methods require repeated similarity searches in high dimensions in order to determine the nearest neighbors. The efficiency of these methods degrades because of two factors: (i) the computations now use a larger number of dimensions, and (ii) the effectiveness of pruning methods and indexing methods degrades with increasing dimensionality. The solution to these issues still remains unresolved in the vast similarity search literature. Therefore, it is unlikely that *significantly* more efficient similarity computations could be achieved in the context of high-dimensional

outlier detection, although some success has been claimed for improving the efficiency of high-dimensional outlier detection in methods proposed in [58, 219, 557]. On the whole, it is unclear how these methods would compare to the vast array of techniques available in the similarity-search literature for indexing high-dimensional data. This chapter does *not* investigate the efficiency issue at all, because the efficiency of a *full-dimensional* outlier-detection technique is not important, if it does not even provide meaningful outliers. Therefore, the focus of the chapter is on methods that *re-define* the outlier detection problem in the context of lower-dimensional projections.

The problem of subspace outlier detection was first proposed in [4]. In this paper, an evolutionary algorithm was proposed to discover the lower dimensional subspaces in which the outliers may exist. The method for distance-based outlier detection with subspace outlier degree was proposed in [327]. Another distance-based method for subspace outlier detection was proposed in [411]. Some methods have also been proposed for outlier analysis by randomly sampling subspaces and combining the scores from different subspaces [344, 367]. In particular, the work in [344] attempts to combine the results from these different subspaces in order to provide a more robust evaluation of the outliers. These are essentially *ensemble-based* methods that attempt to improve detection robustness by bagging the results from different sets of features. The major challenge of these methods is that random sampling may not work very well in cases where the outliers are hidden in specific subspaces of the data. The work in [308] can be considered a generalization of the broad approach in [344], where only high-contrast subspaces are selected for the problem of outlier detection. The use of information-theoretic measures for biased subspace selection is discussed in [413].

The work on isolation forests is related to earlier work on using random forests and clustering forests for clustering, similarity computation, sparse encoding, and outlier detection [99, 401, 491, 555]. In particular, the work in [401] creates extremely randomized clustering forests (ERC-Forests) for clustering and coding. The isolation forest can be viewed as a special case of the ERC-Forest in which the number of trials at each node is set to 1 and the trees are grown to full height. Many variations of the isolation forest [367], such as half-space trees [532], have been proposed. The main difference between an isolation tree and a half-space tree is that the latter is a fully balanced tree of a fixed height, and splits are performed by picking points that are half-way between the minimum and maximum ranges of each attribute. Furthermore, the minimum and maximum ranges of each attribute are defined in a perturbed way to induce diversity. The number of data points in the leaf node of a test instance is multiplied with the number of leaf nodes to obtain its outlier score in a single ensemble component. These scores are averaged over different half-space trees. The multiplication of each component score with the number of leaf nodes is helpful in cases where different trees have different depth [547]. Recently, the subspace histogram technique, referred to as *RS-Hash*, has been proposed in [476]. This techniques averages the log-density in grid regions of varying sizes and shapes in order to provide the final score. The approach uses randomized hashing for efficiency and requires linear time. Such methods can also be used for streaming data.

The reverse problem of finding outlying subspaces *from* specific points was studied in [605, 606, 607]. In these methods, a variety of pruning and evolutionary methods were proposed in order to speed up the search process for outlying subspaces. The work in [59] also defines the exceptional properties of outlying objects, both with respect to the entire population (global properties), and also with respect to particular sub-populations to which it belongs (local properties). Both these methods provide different but meaningful insights about the underlying data. A genetic algorithm for finding the outlying subspaces in high-dimensional data is provided in [606]. In order to speed up the fitness function evaluation,

methods are proposed to speed up the computation of the k-nearest neighbor distance with the use of bounding strategies. A broader framework for finding outlying subspaces in high-dimensional data is provided in [607]. A method that uses two-way search for finding outlying subspaces was proposed in [582]. In this method, full-dimensional methods are first used to determine the outliers. Subsequently, the key outlying subspaces from these outlier points are detected and reported. A method for using rules in order to explain the context of outlier objects is proposed in [405].

A number of ranking methods for subspace outlier exploration have been proposed in [402, 403, 404]. In these methods, outliers are determined in multiple subspaces of the data. Different subspaces may either provide information about different outliers or about the same outliers. Therefore, the goal is to combine the information from these different subspaces in a robust way in order to report the final set of outliers. The *OUTRES* algorithm proposed in [402] uses recursive subspace exploration in order to determine all the subspaces relevant to a particular data point. The outlier scores from these different subspaces are combined in order to provide a final value. A tool-kit for ranking subspace outliers was presented in [403]. A more recent method for using multiple views of the data for subspace outlier detection is proposed in [406]. Methods for subspace outlier detection in multimedia databases were proposed in [64].

Most of the methods for subspace outlier detection perform the exploration in axis-parallel subspaces of the data. This is based on the complementary assumption that the dense regions or clusters are hidden in axis-parallel subspaces of the data. However, it has been shown in recent work that the dense regions may often be located in arbitrarily oriented subspaces of the data [7]. Such clustering methods can be used in conjunction with the methodology discussed in section 5.2.4 to discover outliers. Another work in [328] provides a solution based on local reference sets rather than clusters. The work in [32] proposes a rotated bagging approach; this can be viewed as the analog of the feature bagging approach for the arbitrarily oriented case. Finally, a method for finding outliers in the context of nonlinear subspaces was proposed in [475].

Recently, the problem of outlier detection has also been studied in the context of dynamic data and data streams. The SPOT method was proposed in [604], which is able to determine projected outliers from high-dimensional data streams. This approach employs a window-based time model and decaying cell summaries to capture statistics from the data stream. The most sparse subspaces are obtained by a variety of supervised and unsupervised learning processes. These are used to identify the projected outliers. A multi-objective genetic algorithm is employed for finding outlying subspaces from training data.

The problem of high-dimensional outlier detection has also been extended to other application-specific scenarios such as astronomical data [261], uncertain data [26], transaction data [255], and supervised data [619]. In the uncertain scenario, high-dimensional data is especially challenging, because the noise in the uncertain scenario greatly increases the sparsity of the underlying data. Furthermore, the level of uncertainty in the different attributes is available. This helps determine the importance of different attributes for outlier detection purposes. Subspace methods for outlier detection in uncertain data are proposed in [26]. Supervised methods for high-dimensional outlier detection are proposed in [619]. In this case, a small number of examples are identified and presented to users. These are then used in order to learn the critical projections that are relevant to the outlierness of an object. The learned information is then leveraged in order to identify the relevant outliers in the underlying data.

5.7 Exercises

1. Which of the following data points is an outlier in some well chosen two-dimensional projection: { $(1, 8, 7)$, $(2, 8, 8)$, $(5, 1, 2)$, $(4, 1, 1)$, $(3, 1, 8)$ }

2. Download the *Arrythmia* data set from the UCI Machine Learning Repository [203]. Write a computer program to determine all distance-based outliers in different 2-dimension projections. Are the outliers the same in different projections?

3. In the *Arrythmia* data set mentioned in the previous exercise, examine the *Age*, *Height* and *Weight* attributes of the *Arrythmia data* set both independently and in combination. Draw a scatter plot of each of the 1-dimensional distributions and different 2-dimensional combinations. Can you visually see any outliers?

4. Write a computer program to determine the subspace outlier degree of each data point in the *Arrythmia* data set for all 1-dimensional projections and 2-dimensional projections. Which data points are declared outliers?

5. Write a computer program to perform subspace sampling of the *Arrythmia* data set, using the approach of [344] by sampling 2-dimensional projections. How many subspaces need to be sampled in order to robustly identify the outliers found in Exercise 2 over different executions of your computer program.

6. Consider a data set with d-dimensions, in which exactly 3 specific dimensions behave in an abnormal way with respect to an observation. How many minimum number of random subspaces of dimensionality $(d/2)$ will be required in order to include all 3 dimensions in one of the sampled subspaces with probability at least 0.99? Plot the number of required samples for different values of $d > 6$.