MAFIA: An Adaptive Grid-Based Subspace Clustering Approach

Henrik Daniel Christensen[*hench13@student.sdu.dk*]

University of Southern Denmark, SDU
Department of Mathematics and Computer Science

Abstract. The abstract should briefly summarize the contents of the paper in 150–250 words.

Keywords: High-Dimensional Subspace Clustering · MAFIA · Critical Review · Comparative Study.

1 Introduction

Clustering is one of the main techniques within data mining. This technique is a descriptive method that tries to discover unknown patterns within a data set, by partitioning the data objects into *clusters*. Here, each object in a cluster is similar to one another, but different from objects in other clusters. Clustering is widely used in many applications, such as advertising, biology, web search and business intelligence [4, p. 444].

The task of clustering data is a challenging task, first, the data sets are typical large in size, which means that the clustering algorithm must be *scalable*. Additionally, data sets often contains numerous features (*attributes*), which introduces the problem of *curse of dimensionality*, which refers to a several challenges related to high-dimensional data spaces:

First, the issue of *concentration of distances*, where distances between objects in high-dimensional spaces become increasingly similar as dimensionality increases. This means that data points tend to become nearly equidistant from one another, making it difficult for traditional distance-based algorithms to discover clusters.

Secondly, there is the problem of *local feature relevance* and *local feature correlation*, where only a subset of features or different combinations of feature correlations may be relevant for clustering. Consequently, feature reduction techniques like *Principal Component Analysis (PCA)*, which project the original space onto a lower-dimensional subspace, are inadequate because they typically identify only one global subspace that best fits the entire dataset. Also, algorithms that evaluate the entire feature space does not address this issue effectively. [6, p. 43–46]

Instead of relying on a global approach to feature selection, a local approach that addresses the issues of local feature relevance and local feature correlation is necessary. However, we then need to deal with two separate problems, which both

needs to be solved simultaneously. First, is the problem of finding the relevant subspaces of each cluster. Second, is the problem of finding the clusters in each relevant subspace. To solve them efficiently, heuristics needs to be employed into the clustering algorithms. [6, p. 6–7]

For many applications, it is reasonable to focus only on clusters in axis-parallel subspaces, thus restricting the search space to $O(2^d)$ dimensions. These algorithms are called *subspace clustering* (or *projected clustering*) algorithms. These can be further divided into: *top-down-* or *bottom-up* approach. In top-down approach, the relevant subspaces for the clusters are determined by gradually reducing the subspaces, starting from the entire space. In contrast, bottom-up approaches, finds the relevant subspaces for the clusters from the original space starting from 1-dimensional using the *monotonicity property* (or *downward closure property*), see Lemma 1 [2]. [6, p. 8, 11]

For the rest of this paper, the following notation will be adopted: Let $\mathcal{A} = \{A_1, \ldots, A_d\}$ be a set of attributes, and $\mathcal{S} = A_1 \times A_2 \times \ldots A_d$ a d-dimensional numerical space containing n points $p_1, \ldots, p_n$. Let $A_1, \ldots, A_d$ be the dimensions (attributes) of $\mathcal{S}$.

Lemma 1. *If a collection of points C is a cluster in a k-dimensional subspace, then C is also part of a cluster in any $(k-1)$-dimensional projection of this space.*

1.1 Contributions

The primary focus of this paper is to analyze the bottom-up subspace clustering algorithm *MAFIA* [8], which builds upon the pioneering subspace clustering algorithm *CLIQUE* [2]. This paper examines the relationship between these two grid-based algorithms, highlighting their similarities and differences. Additionally, the density-connectivity-based algorithm *SUBCLU* [7] is included for analysis and evaluation, as it offers a contrasting approach to grid-based methods.

The remainder of the paper is structured as follows: Section 2 describes and analyzes the three algorithms in detail. Section 3 evaluates their performance in terms of scalability, considering dataset size, data- and cluster-dimensionality, as well as their clustering quality. Section 5 discusses the findings and explores the contributions of each algorithm to the field of subspace clustering. Finally, Section 6 concludes the main findings and suggests some future work.

2 Subspace Clustering

The main idea of subspace clustering is to identify subspaces of a high dimensional space to allow better clustering than the original (full) space. This is opposed to e.g. PCA, which projects the original space onto a new subspace, which may can be hard to interpret for the user.

Three different density-based subspace clustering algorithms will be discussed. First, the grid-based approach will be discussed, after which the density-based approach will be discussed. Note that, only bottom-up approaches will be considered in this paper.

2.1 CLIQUE

The key idea of grid-based subspace clustering is to partition the $\mathcal{S}$ into axis-parallel grid structure starting in 1-dimensional space. The grids forms hyper-rectangular *units* (cells) for which we find the number of points in each. Only the units that exceeds a certain threshold are retained, called *dense units*. Next, adjacent dense units will be merged to form so called *candidate dense units* (CDUs), which will be used to find clusters in higher dimensional subspaces. The goal is then to describe the clusters using a minimal description in the form of DNF (*Disjunctive Normal Form*) expressions.

CLIQUE is a bottom-up, grid-based subspace clustering algorithm that uses the monotonicity property as its clustering criterion, as described in Lemma 1.

The proof is provided in [2].

The algorithm begins by partitioning the dataset $\mathcal{S}$ into equal-sized intervals (also called windows) of width ε (input parameter), creating axis-parallel rectangular units. It then scans the dataset to identify 1-dimensional (1D) dense units by counting the number of points in each interval using a histogram, as shown in Figure 1. For example, with a grid size of $\varepsilon = 0.2$, three intervals in dimension A_1 exceed the density threshold τ (input parameter), while dimension A_2 reports four dense units. These dense units are known as *candidate dense units* (CDUs).

The algorithm proceeds incrementally: it starts from 1D and moves to 2D, 3D, and so on, until no more CDUs can be generated. At each stage, the algorithm makes another pass over the dataset to determine which CDUs are dense in the higher dimensions, using the $(k-1)$-dimensional dense units identified in the previous stage. Specifically, CDUs in k dimensions are formed by merging dense units in $(k-1)$ dimensions that share the first $(k-2)$ dimensions. This process continues until no further CDUs are generated.

To optimize performance, CLIQUE applies a pruning technique called *coverage* (from [2]) to reduce computation time. This technique prunes subspaces with low coverage (i.e., those containing fewer points). However, this may risk eliminating subspaces that could potentially contain clusters.

Once all dense units are identified, CLIQUE defines clusters as maximal sets of connected dense units. For each k-dimensional subspace, the algorithm computes disjoint sets of connected dense units, where two units are connected if they share $k-1$ dimensions (a common face in the k-dimensional space) or if they are indirectly connected through other units. The algorithm then generates minimal cluster descriptions by covering each set of connected dense units with maximal regions and determining the minimal cover. For instance, in Figure 1, the minimal cover of the clusters is $A \cup B$, expressed as: $((0.2 \leq A_1 < 0.6) \wedge (0.4 \leq A_2 < 0.8)) \vee ((0.4 \leq A_1 < 0.8) \wedge (0.2 \leq A_2 < 0.6))$.

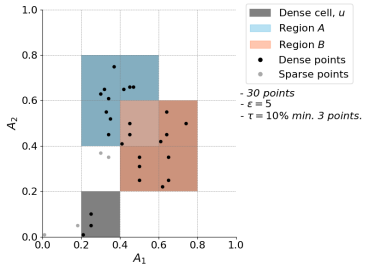
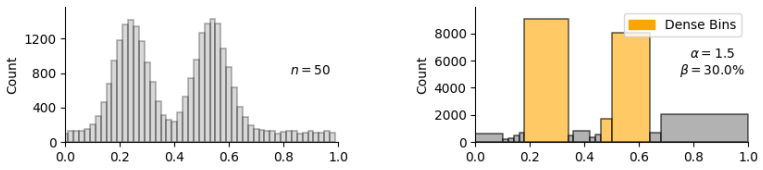


Fig. 1. Illustration of a dense unit u and two overlapping dense regions A and B.

2.2 MAFIA

MAFIA is an extension to CLIQUE where the grid sizes are adaptive meaning that the grid sizes are not fixed but are automatically determined by an algorithm. Moreover, the intention is not to rely on user specified input parameters like CLIQUE. In addition, MAFIA do not implement the pruning technique as noted in [2], this could result in lost information.

The authors of MAFIA therefore introduces the *Adaptive* algorithm, which starts by divide each dimension into a high number of equal-sized bins – default it is $n = 1000$ bins. After dividing the dimension into n bins, it merges the bins from left to right. Two bins are merged together if they are within a percentage of difference β (input parameter). That means, a high beta value result in many merged bins, and vice-versa. If two bins are merged together, the highest bin count of the two are used to further merge with the next bin. An example is shown in Figure 2, where in (a) shows the bins before merging and in (b) shows the bins after merging. The result of the algorithm, as can be seen in the figure, is that bins with similar counts are merged together. At last the algorithm determines a so called *threshold-level* for each of the variable-sized bins. The threshold-level is determined by the size of the bin and a so called *cluster dominance factor* α (input parameter). A higher α will result in a higher threshold-level, meaning that the bin must contain more points to be considered a dense unit, and vice-versa.



(a) Uncombined bins **(b)** Combineed bins

Fig. 2. Illustration of the adaptive grids computation.

Having completed the adaptive grid computation, MAFIA proceeds like CLIQUE by scanning the dataset to identify dense units in each subspace. However, the CDUs are generated differently.

CDU Generation

2.3 SUBCLU

A drawback of grid-based methods is that the quality of clustering depends on the positiong of the grids. In Figure 1, we see that the due to the rigid grid structure we might miss cluster points, that is point not in a dense cell. This problem especially occurs for clusters that are not of rectangular shapes. In contrast, SUBCLU which extends the principles of DBSCAN to subspaces, that is it relies on the locality assumption of points.

The algorithm starts by identifying clusters in 1-dimensional subspaces, applying DBSCAN to each attribute separately. In this step, it looks for dense regions using a parameter ϵ, which defines the radius within which points are considered neighbors, and m, the minimum number of points required for a region to be dense. If clusters are detected in these 1-dimensional subspaces, they are further explored in higher dimensions.

The next phase involves a bottom-up approach where the algorithm combines previously detected k-dimensional subspaces that share $k-1$ attributes to generate $(k+1)$-dimensional candidate subspaces. By extending the clusters found in lower-dimensional subspaces, SUBCLU tests if they persist when new dimensions are added. DBSCAN is again applied in these candidate subspaces with the same ϵ and m values, checking whether clusters detected in the lower-dimensional space still qualify as clusters when higher dimensions are included.

Efficiency is achieved through the monotonicity property of density-connected sets: if no clusters exist in a k-dimensional subspace, then higher-dimensional subspaces derived from it will not contain clusters, allowing the algorithm to prune the search space.

For candidate subspaces where clusters are found, SUBCLU further refines these clusters by choosing subspaces with minimal numbers of objects in the clusters, which reduces the computational load when expanding them into higher dimensions.

3 Evaluation

The evaluation of the clustering algorithms was performed on a Intel i7 1.70 GHz processor (12th gen.) with 16 GB of RAM running Windows 11.

The evaluation of MAFIA was performed using *GPUMAFIA* [1], which was installed on a virtual machine (VM) running Ubuntu. The VM was configured with 4 CPUs and 4 GB of RAM.

CLIQUE and SUBCLU were evaluated on the main machine using *ELKI* [9]. Thus, one should be careful to compare the results of the three algorithms directly, as the execution environment and the implementation may affect the results. However, the growth rate and their clustering of the data sets can be compared.

Throughout the different experiments, a range of input parameters for the three algorithms were tested. The best found were selected. The complete evaluation project can be found in the GitHub repository: https://github.com/henrikdchristensen/SDU-Data-Mining-Exam. Here, additional experiments as well as detailed descriptions of how to generate the data sets and how to install and use GPUMAFIA is described.

3.1 Data set generation

The aim for the synthetic data generation was to be able to produce similar data sets as discussed in [2,8]. That is, hyper-rectangles (axis-parallel) clusters in different subspaces. This was achieved by using *MDCGen*, where different sizes and different densities of the clusters could be determined, as well as which attributes for each cluster that are noise for which values at random is selected from a uniform distribution over the entire range of the attribute.

However, to see the one of the main advantages of SUBCLU compared to CLIQUE and MAFIA, a data set containing a Bezier-shaped cluster was created using *Artificial Cluster* (AC). In addition, a self-populated data set containing a plus-shaped cluster was created as discussed in [8].

Finally, two real-world data sets were selected to evaluate the algorithms in a more realistic setting.

All the data sets were normalized such that each attribute was in the range [0, 1].

3.2 Experimental Results

Scalability with Data Set Size To evaluate the scalability of the algorithms a data set containing 20 dimensions with 5 clusters in 5 different subspaces with 10% noise records was used. The data set size ranges from 10k to 1mio records. The results can be seen in Figure 3. As can be seen, MAFIA is the most scalable algorithm of the three. CLIQUE could handle up to 500k records, while SUBCLU was only able to handle up to 100k records. Similar results in terms of growth rate were reported in all three papers [8,2,7]. That is, linear growth for MAFIA and CLIQUE, and quadratic growth for SUBCLU. The linear behaviour of CLIQUE and MAFIA comes from the fact that the number of passes over the data set depends only on the dimensionality of the embedded cluster. An increase in the size of the data set just means that more data has to be scanned on every pass over the data set while finding dense units resulting in a linear growth rate. SUBCLU, on the other hand, has a quadratic growth rate, as it relies on the DBSCAN algorithm which needs partial range queries that can be costly in terms of running time. Better running time of all three is achieved compared to the three papers. This is probably, due to better hardware.

Note that, the `minpts` in SUBCLU were scaled linearly with the data set size, as the clusters in the dateset were of a fixed size, which means, by increasing the data set size, the density of the clusters increases linearly. However, for the

other two algorithms, there was no need to scale any of input parameters, as they relies on the density of the units for the total amount of records.

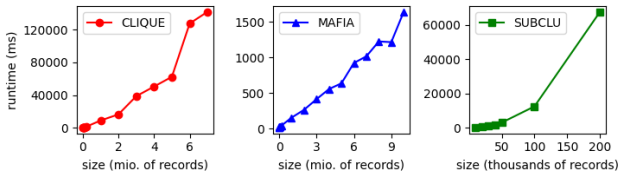
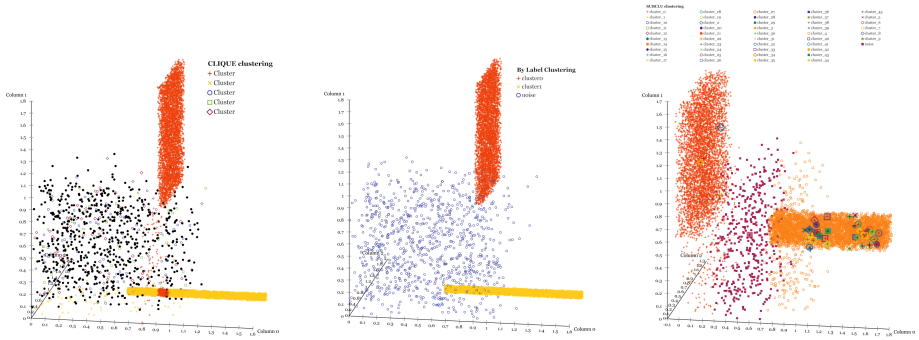


Fig. 3. Scalability with increasing data set size.

Clustering Accuracy Many different synthetic data sets were generated to test the clustering accuracy of the algorithms.

The first data set were a 2-dimensional data set containing a single cluster formed as a skewed plus-sign with 10% noise added. Similarly results as in [8] was achieved, MAFIA detects the 2-dimensional cluster accurately, whereas CLIQUE finds two overlapping clusters, where none of these detects the borders correctly. The accuracy of MAFIA comes, however, with a cost, as it also reports some lower-dimensional clusters as well.

The second data set contains 10 dimensions with contains 2 clusters embedded in a different 4 dimensional subspace. 10% of the data was added as noise records. Similar data set as the one used in [8]. The results can be seen in Figure 4. Here, MAFIA was able to detect the clusters without any additional lower-dimensional clusters. In contrast, CLIQUE reports some overlapping clusters and some of the noise records as clusters. SUBCLU detects also the two clusters, but reports many lower-dimensional clusters and noise records as clusters as well.

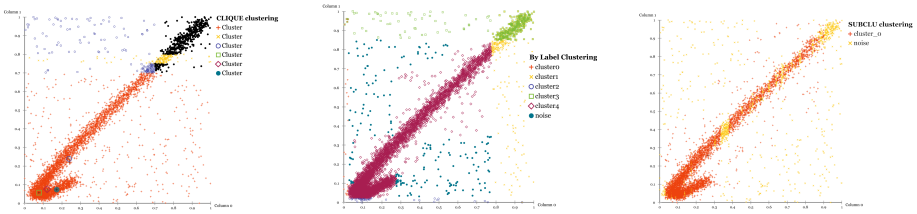


(a) CLIQUE (b) MAFIA (c) SUBCLU

Fig. 4. Two clusters in 4 different subspaces.

The third data set were a 2-dimensional data set containing a single cluster formed as a Bezier curve with 10% noise added to the data set. As expected, SUBCLU outperforms the other two algorithms as can be seen in Figure 5. MAFIA and CLIQUE only partly detects the cluster and reports additional

clusters and noise records as clusters. The result of SUBCLU demonstrates its locality assumption and its use of DBSCAN.



(a) CLIQUE (b) MAFIA (c) SUBCLU

Fig. 5. A single bezier-shaped cluster.

Scalability with Data Dimensionality and Cluster Dimensionality Figure 6 shows the scalability of the CLIQUE and MAFIA with increasing data set dimensionality. The data set contains 1 mio. records with 3 clusters in 5 different subspaces and 10% noise records, similar to the data set in [8]. The data set dimensionality ranges from 10 to 100 dimensions. However, CLIQUE was only able to handle half of the dimensions, as the PC constantly freezes – probably due to the high memory consumption. In [2] it is noted that CLIQUE has a quadratic growth rate in terms of data set dimensionality. To investigate this even further, one could try to reduce the memory consumption by having a smaller amount of records in the data set. Nevertheless, MAFIA is the most scalable algorithm of the two.

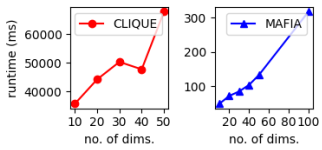


Fig. 6. Scalability with increasing data set dimensionality.

Figure 7 shows the scalability of the CLIQUE and MAFIA with increasing cluster dimensionality. A 20-dimensional data set, containing 500k records with a single cluster were used. The cluster was embedded in increasing number of dimensions starting from 10 to 100 dimensions. 10% of the records was added as noise. The data set is similar to the one used in [8]. Both algorithms heavily suffers from the increasing cluster dimensionality, however, MAFIA is able to handle a higher dimensional cluster than CLIQUE. The reason why both algorithms depends on the cluster dimensionality is due to the fact that a higher cluster dimensionality results in a large subspace coverage and a large number of CDUs. In other words, each pass on the data needs to populate a large number of CDUs, adn increase in cluster dimensionality also increases the number of passes over the data set.

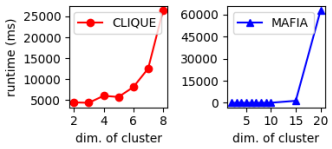
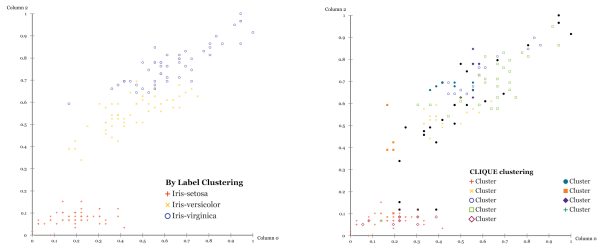


Fig. 7. Scalability with increasing cluster dimensionality.

The results reported for SUBCLU in [7] were also tried to be replicated, however, similar it was not possible evaluate in different scales of data set dimensionality and cluster dimensionality, as the PC constantly freezes – probably due to the high memory consumption. In future work, it would be interesting to investigate this further.

Real Data Sets Two real-world data sets are used to evaluate the algorithms in a more realistic setting. Small data sets were selected for both performance and visualization reasons. Both data sets were normalized such that each attribute was in the range [0, 1].

The first data set is the well-known *Iris* data set [3], which contains 150 records of 3 different iris types (e.g. Setosa, Versicolor, Virginica). The data set contains 4 features (sepal length, sepal width, petal length and petal width). All three algorithms produced some meaningful clusters, however, as can be seen in Figure 8. It is hard to determine which algorithm performs the best in this case.



(a) Original **(b)** CLIQUE

(c) MAFIA **(d)** SUBCLU

Fig. 8. Iris data set.

The second data set is the *Date Fruit* data set [5], which contains 898 records of 7 different date fruit types (e.g. Barhee, Deglet Nour, Sukkary, Rotab). These

were obtained via a computer vision, where 34 features (e.g shape and color) was extracted. Only SUBCLU were able to produce clusters somewhat close to the true clusters but also finds many lower-dimensional clusters and small clusters. CLIQUE and MAFIA either produced way too many clusters or close to one single cluster.

4 Discussion

CLIQUE, MAFIA, and SUBCLU are three prominent subspace clustering algorithms, each offering unique strengths and limitations. CLIQUE, as one of the earliest subspace clustering approaches, uses a grid-based method where the data space is partitioned into equi-sized cells, and dense units are identified based on a density threshold. The advantage of CLIQUE is its simplicity and efficiency in navigating through subspaces using an Apriori-like strategy. However, its reliance on fixed grid boundaries can lead to issues when clusters do not align with these grids, resulting in missed or fragmented clusters. Moreover, the algorithm is sensitive to grid size and density thresholds, which may not be optimal for all datasets.

MAFIA builds upon CLIQUE by introducing adaptive grid sizes, allowing it to dynamically adjust grid boundaries based on data density. This modification enables MAFIA to detect clusters more precisely, providing better resolution than CLIQUE, especially in complex datasets. Additionally, experiments indicate that MAFIA is the most scalable algorithm when it comes to handling large data sizes. However, this scalability comes at a cost; MAFIA is highly sensitive to input parameters, such as grid size and density thresholds, and requires careful tuning to achieve optimal results. While it offers more precise boundary detection than CLIQUE, its dependence on parameters can lead to inconsistent performance across different datasets.

SUBCLU, on the other hand, departs from the grid-based approach and utilizes a density-connectivity principle similar to DBSCAN. This enables it to detect clusters of arbitrary shapes, which is a significant advantage when clusters are irregular or not aligned with the axes. This was demonstrated in experiments using real datasets, where SUBCLU outperformed grid-based methods in identifying clusters with complex boundaries. However, SUBCLU's approach, while flexible, may not be as efficient as MAFIA in terms of data size scalability, particularly for very large datasets.

A critical observation across these algorithms is that their performance can vary significantly depending on the data characteristics. Experiments using synthetic data show that clustering quality is highly use-case dependent, and allowing or disallowing lower-dimensional clusters can affect outcomes. Furthermore, while these synthetic experiments highlight each algorithm's strengths, they also reveal their limitations. It is possible to generate datasets where none of these methods can identify the correct clusters, showing that subspace clustering remains a challenging problem.

All three algorithms struggle with closely clustered data points, often requiring fine-tuning of parameters such as grid size in CLIQUE and MAFIA, or ϵ and m in SUBCLU, to effectively separate clusters. This need for parameter optimization underscores the complexity of subspace clustering and the necessity for tailored solutions depending on the specific data distribution and clustering objectives.

5 Conclusion

The analysis of CLIQUE, MAFIA, and SUBCLU highlights the strengths and limitations of each algorithm in subspace clustering. CLIQUE offers a straightforward, grid-based approach suitable for detecting axis-aligned clusters, but its reliance on fixed grids and sensitivity to parameters can limit performance. MAFIA improves upon CLIQUE by adapting grid sizes, providing better resolution and scalability for large datasets, though it still remains somewhat dependent on parameter tuning. SUBCLU, using a density-based method, excels in identifying arbitrarily shaped clusters, especially in real-world data, but it may not scale as efficiently as CLIQUE and MAFIA.

Overall, the choice of algorithm depends on the use case, data size, and the shape of clusters. The experiments demonstrated that these algorithms perform well under controlled conditions, but in highly complex data sets, none of the algorithms detects clusters accurately. Hence, while each method has its advantages, they all require careful adjustment of parameters and may need further refinement for optimal performance in diverse datasets.

References

1. Adinetz, A., Kraus, J., Meinke, J., Pleiter, D.: Gpumafia: Efficient subspace clustering with mafia on gpus pp. 838–849 (08 2013). https://doi.org/10.1007/978-3-642-40047-6_83
2. Agrawal, R., Gehrke, J., Gunopulos, D., Raghavan, P.: Automatic subspace clustering of high dimensional data for data mining applications p. 94–105 (1998). https://doi.org/10.1145/276304.276314, https://doi.org/10.1145/276304.276314
3. Fisher, R.A.: Iris. UCI Machine Learning Repository (1936). https://doi.org/https://doi.org/10.24432/C56C76
4. Han, J., Kamber, M., Pei, J.: Data Mining: Concepts and techniques. Elsevier (6 2011)
5. Koklu, M., Kursun, R., Taspinar, Y.S., Cinar, I.: Classification of date fruits into genetic varieties using image analysis. Mathematical Problems in Engineering **2021**(1), 4793293 (2021). https://doi.org/https://doi.org/10.1155/2021/4793293, https://onlinelibrary.wiley.com/doi/abs/10.1155/2021/4793293
6. Kriegel, H.P., Kröger, P., Zimek, A.: Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. ACM Trans. Knowl. Discov. Data **3**(1) (Mar 2009). https://doi.org/10.1145/1497577.1497578, https://doi.org/10.1145/1497577.1497578

7. Kröger, P., Kriegel, H.P., Kailing, K.: Density-connected subspace clustering for high-dimensional data **246-257** (04 2004). https://doi.org/10.1137/1.9781611972740.23
8. Nagesh, H., Goil, S., Choud, A., Choudhary, P.: Adaptive grids for clustering massive data sets (01 2002). https://doi.org/10.1137/1.9781611972719.7
9. Schubert, E.: Automatic indexing for similarity search in ELKI **13590**, 205–213 (2022). https://doi.org/10.1007/978-3-031-17849-8_16, https://doi.org/10.1007/978-3-031-17849-8_16