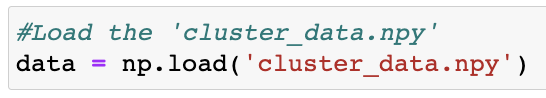
**Hasan Enes Guray**

**19489124**

* The "cluster\_data.npy" dataset for use as input in the homework is loaded into the notebook with NumPy's load formula:



Text

Description automatically generated

* Since the dataset consists of 2 variables, it is possible to visualize it in a 2D plane. The dataset is visualized with the scatter plot method, with the scatter formula of the matplotlib library:

Chart, scatter chart

Description automatically generated

* Thanks to the plot\_clusters formula created, the desired clustering models can be visualized on the desired data set by specifying their parameters. However, it is the clustering method of the model that determines the success of each clustering model, as well as the determination of the parameters. The parameters that can be tuned for the clustering models to be studied are as follows:

**K-Means:** n\_clusters, init, n\_init, max\_iter, tol, verbose, random\_state, algorithm [1]  
**Affinity Propogation:** damping, max\_iter, convergence\_iter, copy, preference, affinity, verbose, random\_state [2]  
**Mean Shift:** bandwith, seeds, bin\_seeding, min\_bin\_freq, cluster\_all, n\_jobs, max\_iter [3]

**Spectral Clustering:** n\_clusters, eigen\_solver, n\_components, random\_state, n\_init, affinity, eigen\_tol, assign\_labels, degree, coef0, verbose, gamma, n\_neighbors, kernel\_params, n\_jobs [4]

**Agglomerative Clustering:** n\_clusters, affinity, metric, connectivity, compute\_full\_tree, linkage, distance\_threshold, compute\_distances [5]

**HDBSCAN:** min\_cluster\_size, min\_samples, memory, cluster\_selection\_epsilon, alpha, cluster\_selection\_method, allow\_single\_cluster [6]

* **K-Means:**

It is possible to visualize and visually analyze a bivariate clustering algorithm. When the dataset is examined without the need for any extra method, 6 clusters that are close to each other are separated from the others.

Also, the optimum number of clusters can be reached by using the elbow method, thanks to the inertia or within-cluster sum of squares (WCSS) score provided by K-Means. The WCSS is the sum of squared distances between each data point and its nearest centroid. The plot should have a distinctive "elbow" shape, and the number of clusters at the elbow point can be chosen as the optimal number of clusters. The lower the value of WCSS or inertia, the better the clustering. As a result of this process, it was seen that 3 clusters were optimum.

On the other hand, when the optimum number of clusters was examined with the distortion score, the result of 4 was obtained. The distortion score only measures the tightness of individual clusters, while the inertia score considers both the tightness of individual clusters and the distance between them.[7][8] Therefore, the inertia score is often considered a more comprehensive measure for evaluating the quality of clustering results, especially when the number of clusters is not fixed. However, the efforts to find the most optimal parameters continued with 6 clusters by trusting the eyes.

Afterward, when the other parameters were tuned by examining the inertia score, silhouette score, and visual analysis thanks to the sklearn library’s formulas the following parameters, which are different from the default one, were reached: plot\_clusters(data, cluster.KMeans, (), {'n\_clusters':6, "random\_state":959, "max\_iter":1000, "init":"random", "n\_init":30, "tol":0.0002, "verbose":0, "copy\_x":True, "algorithm":"lloyd"})

* **Affinity Propagation:**

Contrary to the K-Means clustering model, the number of clusters cannot be directly determined with parameters in Affinity Propagation. Moreover, inertia is not a built-in metric in the sklearn.cluster module, but it can be calculated manually using the cluster assignments and cluster centers obtained from the fitted model. In this model, while preference and dumping parameters play a role in determining the number of clusters, it provides a clearer separation of other parameters. Therefore, since it is known from the K-Means model that there are 6 clusters, the dumping and preference parameters that enable the formation of 6 clusters were examined one by one. Then, the optimum values ​​were determined by visual evaluation, silhouette score, and inertia score; the remaining parameters were also examined by visual observation. As a result, the parameters that provided the most successful clustering were: plot\_clusters(data, cluster.AffinityPropagation, (), {'preference':-10.0, 'damping':0.7, 'max\_iter': 400, 'convergence\_iter':20, 'copy':True, 'affinity':'euclidean', 'verbose': False, 'random\_state':300})

* **Mean Shift:**

As in the Affinity Propagation model, the number of clusters cannot be determined directly in the Mean Shift model. The parameters that determine the number of clusters are bandwidth and min\_bin\_freq. The estimate\_bandwith formula in the cluster section of the sklearn library was used to determine the bandwidth parameter. With the increase of the min\_bin\_freq parameter, the clustering speed also increases, but when certain limits are exceeded, the number of clusters decreases. Therefore, the min\_bin\_freq value was chosen to create the model in the fastest way without reducing the number of clusters to 5. On the other hand, enabling bin\_seeding will accelerate the algorithm as it will initialize fewer seeds. Therefore, setting the bin\_seeding parameter to True greatly speeds up the setup of the model. Unlike other models, the entire data in the dataset may not be included in the clustering process, thanks to the cluster\_all parameter in the Mean Shift model. The points located at more than certain distances from the determined cluster centers are separated as non-clustered. Clustering in this way can be more effective, especially if there is a highly scattered dataset as in the given dataset, because including every data in a cluster may result in incorrect clustering results. As in Affinity Propagation, the inertia score for Mean Shift is not among the metrics in the sklearn library, however, the inertia score for the model can be obtained by applying the method used in Affinity Propagation. As a result, all parameters were evaluated separately by visual observation, inertia, and silhouette scores, and the most optimum result was obtained the following parameters were used: plot\_clusters(data, cluster.MeanShift, (), {'bandwidth':cluster.estimate\_bandwidth(data, quantile=0.12, n\_samples=200), 'seeds':None, 'bin\_seeding':True, 'min\_bin\_freq':3, 'cluster\_all':False, 'n\_jobs':None, 'max\_iter':300})

* **Spectral Clustering:**

Thanks to the n\_clusters parameter of the Spectral Clustering model, the number of clusters can be determined directly. In addition, the success of the cluster models created with the silhouette\_score can also be measured. Therefore, first of all, the n\_clusters parameter was determined as 6, which was previously determined. Then the optimum random\_state parameter was obtained via the metrics and visual observation. As in Affinity Propagation, the inertia score for Spectral Clustering is not among the metrics in the sklearn library however, the inertia score for the model can be obtained by applying the method created in Affinity Propagation. Afterward, assign\_labels, n\_components, and affinity parameters were optimized. Finally, while trying to find the optimum values ​​of other parameters with the same method, the performance could not be increased more and this result was obtained: plot\_clusters(data, cluster.SpectralClustering, (), {'n\_clusters':6, 'random\_state':20, 'eigen\_solver':None, 'n\_components':None, 'assign\_labels':'discretize', 'gamma':1.0, n\_neighbors':10, 'n\_init':10, 'affinity':'rbf', 'kernel\_params':None, 'n\_jobs':None, 'eigen\_tol':'auto', 'degree':3, 'coef0':1, 'verbose':False})

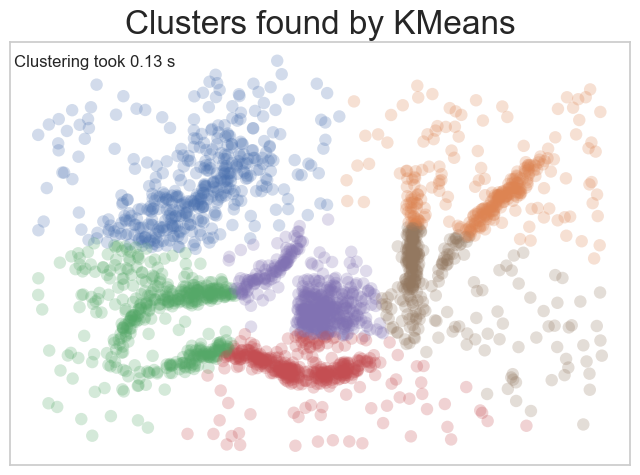
* **Agglomerative Clustering:**

Thanks to the n\_clusters parameter of the Agglomerative Clustering model, the number of clusters can be determined directly. In addition, the success of the cluster models created with the metrics and visual analysis can also be measured. Moreover, although the inertia score cannot be measured directly with a function, it can also be measured thanks to the extra function created for Affinity Propagation. Therefore, first of all, the n\_clusters parameter was determined as 6, which was previously set. Then, the inertia and silhouette scores of the models created with different combinations of all the remaining parameters were calculated. As a result, for the optimum model, the model with the highest silhouette score and the lowest inertia score was visually examined separately and this result was reached by choosing the more successful one: plot\_clusters(data, cluster.AgglomerativeClustering, (), {'n\_clusters':6, 'linkage':'ward', 'connectivity':connectivity\_func, 'affinity':'euclidean', 'metric':None, 'memory':None, 'compute\_full\_tree':'auto', 'distance\_threshold':None, 'compute\_distances':True}). (connectivity\_func = lambda data: kneighbors\_graph(data, 5, mode='connectivity', include\_self=True))

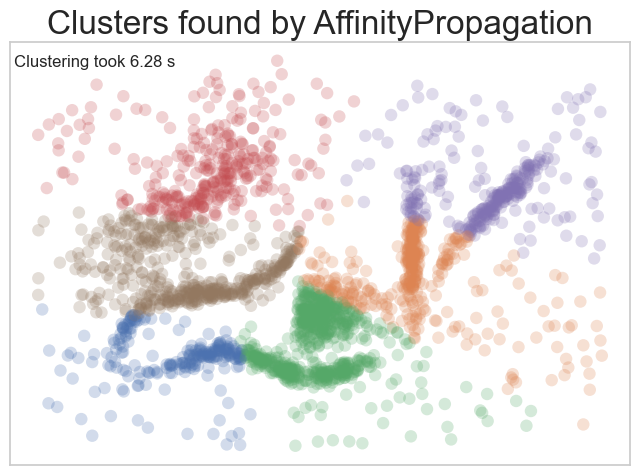
* **HDBSCAN:**

In the HDBSCAN clustering model, no parameter can directly define the number of clusters. The number of clusters is determined by the parameters min\_cluster\_size, cluster\_selection\_method, min\_samples, and cluster\_selection\_epsilon. As with other models, the inertia score can be calculated directly with the formulas in the library. However, the cluster\_persistence function returns an array of persistence values, where each value corresponds to a cluster assigned by the HDBSCAN model. These values can be interpreted as a measure of cluster stability, where higher values indicate more stable clusters. A variant of the inertia score is obtained by summing up the persistence values for all non-noise clusters. As in other models, in this model, silhouette and inertia scores were examined together with the number of clusters, and optimum parameters were decided. The silhouette score or inertia score is just one of several metrics, which can be used to evaluate the quality of the clustering solution. It is important to also visually inspect the clusters and apply domain knowledge to interpret the results. Since the HDBSCAN model does not include the entire dataset in the clustering process, it prefers to use less of the dataset to achieve more successful results when measured only with metrics. Although this method seems successful, it ignores most of the dataset. Therefore, a more successful clustering process can be applied when the HDBSCAN model is analyzed with visual analysis rather than metrics. As a result of the analysis, the optimum result was obtained as follows: plot\_clusters(data, hdbscan.HDBSCAN, (), {'min\_cluster\_size':11, 'cluster\_selection\_method':'eom', 'min\_samples':15, 'cluster\_selection\_epsilon':0.01, 'alpha': 1.0, 'allow\_single\_cluster':False})

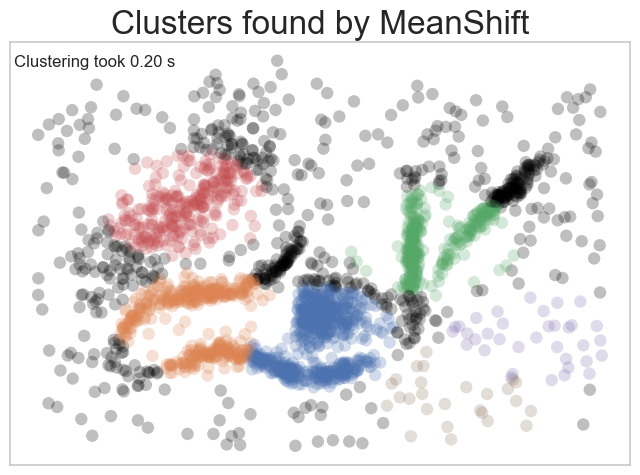
* K-Means



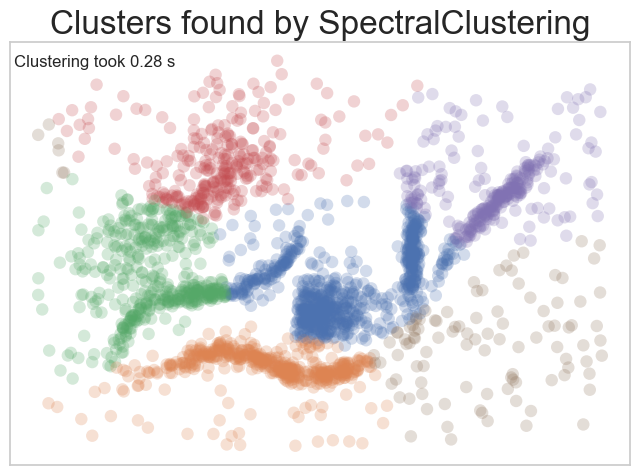
* Affinity Propagation



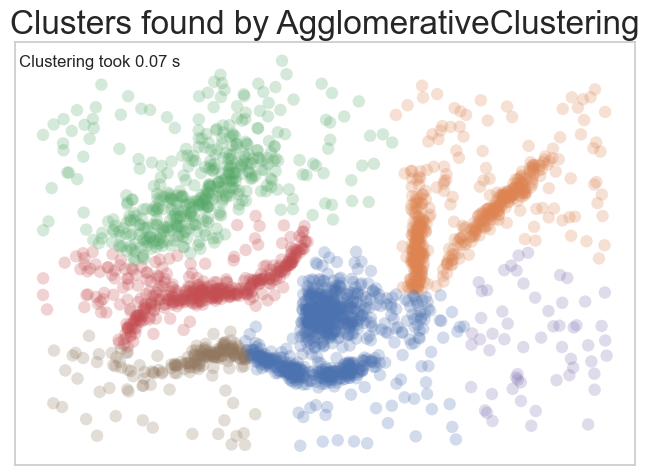
* Mean Shift



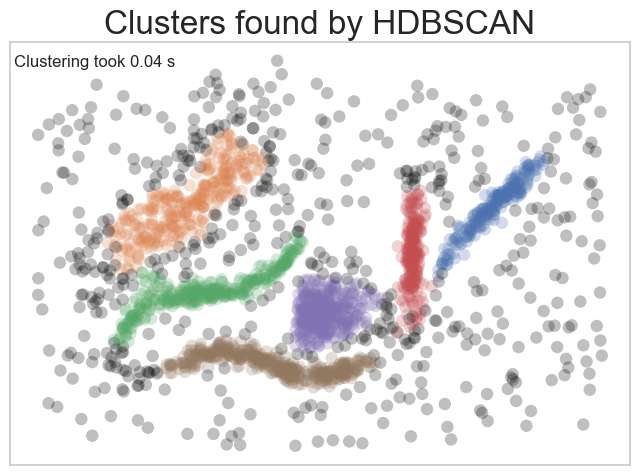
* Spectral Clustering



* Agglomerative Clustering



* HDBSCAN



Clustering algorithms are a modeling technique that necessitates a meticulous analysis and appraisal not only based on quantitative data but also through visual inspections. Additionally, incorporating domain expertise can considerably enhance the meaningfulness and effectiveness of the analyses. In this technical report, the dataset was examined without any domain knowledge and prior knowledge of the optimal number of clusters, utilizing state-of-the-art techniques to attain the most favorable outcomes.

Choosing the right parameters is crucial for effective clustering. K-Means, Spectral Clustering, and Agglomerative Clustering rely on the n\_clusters parameter, which can be determined through the elbow method or visual inspection. Spectral Clustering is also affected by setting the n\_components parameter. Affinity Propagation needs preference and damping parameters for successful clustering. Mean Shift needs fine-tuning of the bandwidth and min\_bin\_freq parameters, along with the use of bin\_seeding and min\_bin\_freq parameters for speeding up. HDBSCAN needs critical parameters such as min\_cluster\_size, min\_samples, and cluster\_selection\_method for optimal performance. Other parameters examined throughout the study are also very important for the success of the clustering process, but these parameters are critical. Inertia, silhouette scores, and visual analysis help optimize parameters. It's also recommended to set the max\_iter parameter above the convergence point and fix the random\_state parameter, if applicable, for stability.

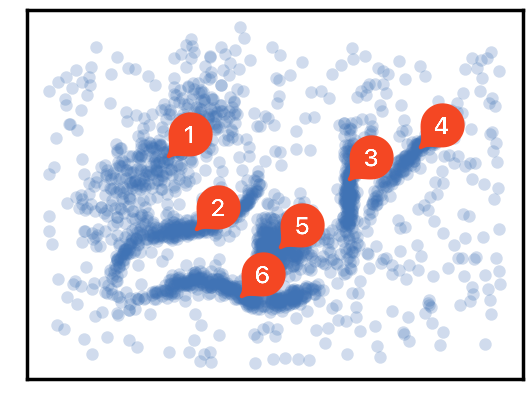
Clustering algorithms can be evaluated using two metrics: inertia score and silhouette score. The former measures the sum of squared distances between points and their cluster centroid, while the latter measures the distance between points in one cluster and their nearest neighboring cluster. When the silhouette score is high, it signifies that the clusters are well separated, but if it is low, it suggests that the clusters may be overlapping or not properly separated.[12] To achieve the best clustering results, the ideal clustering algorithm must balance the intra-cluster (measured by inertia score) and inter-cluster (measured by silhouette score) distances. The quality of the clustering results is determined by the distances between points in the dataset.[13] Through a process of hyperparameter tuning and visual analysis of six clustering models, this study has shown that a balance between intra-cluster and inter-cluster distances yields the best clustering outcomes:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Clustering Algorithms | Cluster Number | Inertia Score | Silhouette Score | All clustered? | Time(s) |
| K-Means | 6 | 44.03 | 0.42 | Yes | 0.13 |
| Affinity Propagation | 6 | 44.33 | 0.38 | Yes | 6.28 |
| Mean Shift | 6 | 55.46 | 0.22 | No | 0.21 |
| Mean Shift | 6 | 53.64 | 0.40 | Yes | 0.2 |
| Spectral Clustering | 6 | 45.81 | 0.39 | Yes | 0.28 |
| Agglomerative Clustering | 6 | 47.72 | 0.39 | Yes | 0.07 |
| HDBSCAN | 6 | 1.26 | 0.46 | No | 0.04 |

After examining the algorithms individually, we found that HDBSCAN outperformed the other models due to its low Inertia score, high silhouette score, and fast clustering process. HDBSCAN is the best algorithm for our dataset without domain-specific information. However, if the entire dataset needs to be used for clustering, K-Means, and Agglomerative Clustering are recommended. K-Means has the best inertia and silhouette score, while Agglomerative Clustering is the fastest algorithm. It should be kept in mind that the choice of algorithm depends on the dataset's characteristics. HDBSCAN is the most effective algorithm when the entire dataset is not needed, but K-Means and Agglomerative Clustering are better options for optimal results and faster processing, respectively, when the entire dataset is required.

After the analysis with numerical data, algorithms should be examined in terms of 4 basic rules of EDA clustering:

* **Don’t be wrong!:** In clustering algorithms, K-means and Affinity Propagation tend to group data points into clusters that may not accurately reflect their true membership. Moreover, these methods assume the shape of clusters to be globular. While Mean Shift strives for globular clusters, its outcomes may fall short of optimality. Spectral Clustering, although not strictly assuming globular clusters, still partitions the data, leading to the creation of noisy clusters. In contrast, Agglomerative Clustering does not impose assumptions about globular clusters but presupposes that all data points belong to clusters, rendering noise exclusion challenging. The HDBSCAN algorithm represents the most resilient option, given its lack of assumptions regarding globular clusters and noise grouping, thereby conferring an advantage in avoiding spurious clustering outcomes.
* **Intuitive Parameters:** During the process of hyperparameter tuning, utilizing metrics such as Inertia and Silhouette score, along with visual analysis, can prove to be essential tools in elevating the comprehensiveness and effectiveness of clustering algorithms. Notably, K-Means, Spectral Clustering, and Agglomerative Clustering algorithms offer the convenience of directly determining the number of clusters via a specific parameter, which proves particularly advantageous when the target number of clusters is predetermined. In Affinity Propagation, the preference and damping parameters determine the number of clusters and labeling methodologies; however, the efficacy and scope of these parameters remain limited. Despite having fewer parameters, the HDBSCAN algorithm distinguishes itself by enabling the specification of minimum cluster size. In contrast, the Mean Shift algorithm possesses a diverse range of parameters that impact the clustering process, thereby facilitating a range of distinct clustering strategies.
* **Stable Clusters:** The stability of various clustering algorithms is contingent upon the characteristics of the data and the specific hyperparameters employed. K-Means is prone to instability and necessitates multiple runs to achieve consistent clustering outcomes. Conversely, Affinity Propagation exhibits greater stability owing to its deterministic nature. Mean Shift's performance can be highly variable depending on the choice of bandwidth parameter and the random initialization. Spectral Clustering is comparatively more stable than K-Means, but still vulnerable to similar concerns. In contrast, Agglomerative Clustering is highly stable across diverse parameter choices and multiple runs, and its dendogram provides a consistent representation of how the data is clustered. Finally, HDBSCAN is the most stable alternative due to its ability to adapt to variable density clustering and its robustness to subsampling, resulting in dependable results over a range of parameter selections.
* **Performance:** The six clustering algorithms have varying levels of performance and efficiency. The K-Means algorithm is quick and suitable for large datasets, with the potential for further optimization. The Affinity Propagation algorithm can be slow and challenging for larger datasets. The Mean Shift algorithm has good scalability but may be slow in practice, while the Spectral Clustering algorithm is slower than K-Means but can handle large datasets accurately. The Agglomerative Clustering algorithm's performance depends on the implementation used. HDBSCAN algorithm is particularly effective for larger datasets due to its optimized parameter tuning and high clustering capability. Therefore, while the Affinity Propagation algorithm is the slowest and not ideal for large datasets, K-Means, and HDBSCAN algorithms are the fastest algorithms.



After clustering with optimal parameters, six different scatter plots were generated, which were visually determined initially. The hyperparameter tuning operations were focused on achieving these clusters. Upon examining the K-Means and Mean Shift algorithm clusters, the 1st Cluster was formed accurately, but other clusters were mixed. The Affinity Propagation algorithm formed only a significant portion of the 2nd Cluster. The Spectral Clustering algorithm separated only the 6th cluster correctly. The Agglomerative Clustering algorithm successfully formed the 1st and 2nd clusters. However, the HDBSCAN algorithm outperformed all others, creating clusters in close proximity to the visually predetermined clusters. HDBSCAN is the most effective algorithm for numerical data and demonstrated superior performance based on visual analysis.

In conclusion, clustering algorithms are an essential tool for data analysis, and their success depends on selecting the appropriate parameters for the given dataset. The analysis shows that the most successful algorithm varies depending on the purpose of clustering, the data, and the metrics used. Therefore, a combination of domain knowledge, visual analysis, and appropriate metrics is necessary to achieve a comprehensive and successful clustering result.

**References**

[1] *Sklearn.cluster.KMeans* (no date) *scikit-learn*. scikit-learn. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html (Accessed: April 24, 2023).

[2] *Sklearn.cluster.AffinityPropagation* (no date) *scikit-learn*. scikit-learn. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AffinityPropagation.html (Accessed: April 24, 2023).

[3] *Sklearn.cluster.MeanShift* (no date) *scikit-learn*. scikit-learn. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.MeanShift.html (Accessed: April 24, 2023).

[4] *Sklearn.cluster.SpectralClustering* (no date) *scikit-learn*. scikit-learn. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.SpectralClustering.html (Accessed: April 24, 2023).

[5] *Sklearn.cluster.AgglomerativeClustering* (no date) *scikit-learn*. scikit-learn. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html (Accessed: April 24, 2023).

[6] *Parameter Selection for HDBSCAN\** (no date) *hdbscan*. hdbscan. Available at: https://hdbscan.readthedocs.io/en/latest/parameter\_selection.html (Accessed: April 24, 2023).

[7] Onnen, H. (2021) *Elbows and Silhouettes: Hands-on Customer Segmentation in Python*, *Medium*. Towards Data Science. Available at: https://towardsdatascience.com/elbows-and-silhouettes-hands-on-customer-segmentation-in-python-66c2e794c552 (Accessed: April 24, 2023).

[8] Belik, V. (2022) *silhouette score vs Distortion score*, *Stack Exchange*. Stack Exchange. Available at: https://stats.stackexchange.com/questions/576902/silhouette-score-vs-distortion-score (Accessed: April 24, 2023).

[9] Maklin, C. (2019) *Affinity Propagation Algorithm Explained*, *Medium*. Towards Data Science. Available at: https://towardsdatascience.com/unsupervised-machine-learning-affinity-propagation-algorithm-explained-d1fef85f22c8 (Accessed: April 24, 2023).

[10] Sun, S. (2020) *Implementing Self-Tuning Spectral Clustering*, *Huiwenn*. Github. Available at: https://huiwenn.github.io/spectral-clustering (Accessed: April 24, 2023).

[11] Snow, M. (2018) *Unsupervised Document Clustering with Cluster Topic Identification* [Preprint]. Available at: https://www.researchgate.net/figure/Example-of-HDBSCAN-clustering\_fig2\_328530481 (Accessed: April 24, 2023).

[12] Bhardwaj, A. (2020) *Silhouette Coefficient*, *Medium*. Towards Data Science. Available at: https://towardsdatascience.com/silhouette-coefficient-validating-clustering-techniques-e976bb81d10c (Accessed: April 24, 2023).

[13] Zhou, S., Xu, Z. and Liu, F. (2016) “Method for Determining the Optimal Number of Clusters Based on Agglomerative Hierarchical Clustering,” *IEEE Transactions on Neural Networks and Learning Systems*, pp. 99. Available at: https://doi.org/10.1109/TNNLS.2016.2608001.