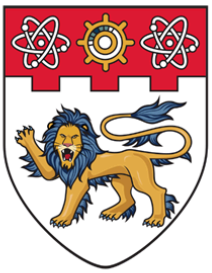


NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

AY2025/26 SEMESTER 1

SC4020 DATA ANALYTICS & MINING

Group: 21

Project 1

Topic/Task: Clustering

Links: Google Drive | Review Report | Code (Google Colab) | Contribution Report

#	Name	Email	Matriculation #
1	ARYAN NANGIA	ARYAN010	U2221575A
2	LUKE ENG PENG KEE	LUKE0019	U2321688J
3	PANG YEE LEONG	PANG0257	U2322916B
4	PUAH RONG QI	RPUAH002	U2221477H

Table of Contents

1. Abstract

In this project, we study the clustering algorithms Mean Shift, DBSCAN and K-Means thoroughly in order to understand their behavior over a variety of datasets such as student records, weather records and COVID-19 case data. The approach applies a hierarchical procedure for feature pre-processing, parameter search and internal evaluation metrics of Silhouette Score, Davies-Bouldin Index, Calinski-Harabasz. Our experiments show that each of these methods has its own advantages and limitations: Mean Shift is computationally expensive yet is able to produce relatively good separation in distance-based clustering; DBSCAN performs well on irregularly shaped clusters and outliers, especially for data with low dimensions or spatial position but does not work as well when the dimensionality increases, nor it reacts very sensitively to its parameters; K-Means runs fast and indicates more stability with spherical clusters that are clearly separated from each other, however provides worse results when the number of predefined cluster centers is incorrect or when clusters are non-uniform. The research has also demonstrated that depending on the size, dimensionality and density of static dataset as well as clustering goal of interest how much biased performance can have been witnessed by a selection of one algorithm over another.

2. Introduction

2.1. Data Processing

Datasets

weather.csv

Column Name	Description	Data Type	Example Value	Units
Region	Geographical region where the weather station is located	string	"North"	-
Station	Name of the specific weather observation station	string	"Admiralty"	-
Date	Observation date	date	"01-01-2020"	-
Daily_Rainfall_Total	Total rainfall accumulated during the day	float	7	mm
Rainfall_30min	Maximum rainfall recorded in any 30-minute period	float	6.2	mm
Rainfall_60min	Maximum rainfall recorded in any 60-minute period	float	6.8	mm
Rainfall_120min	Maximum rainfall recorded in any 120-minute period	float	7	mm
Temp_Mean	Mean daily temperature	float	27.5	°C
Temp_Max	Maximum temperature recorded during the day	float	31.6	°C
Temp_Min	Minimum temperature recorded during the day	float	25.1	°C
Wind_Mean	Mean wind speed during the day	float	22	km/h
Wind_Max	Maximum instantaneous wind speed during the day	float	57.6	km/h

student.csv

Column Name	Description	Data Type	Example Value	Units
student_id	Unique identifier for each student	string	"S1000"	-
age	Age of the student	integer	21	years
gender	Gender of the student	string	"Female"	-
study_hours_per_day	Average number of hours the student studies daily	float	4.3	hours/day
social_media_hours	Average daily time spent on social media	float	2.8	hours/day

netflix_hours	Average daily time spent watching Netflix or other streaming platforms	float	1.1	hours/day
part_time_job	Whether the student has a part-time job	string	"Yes"	-
attendance_percentage	Class attendance rate	float	85.8	%
sleep_hours	Average daily sleep duration	float	7.4	hours/day
diet_quality	Subjective rating of the student’s diet	string	"Fair"	-
exercise_frequency	Number of exercise sessions per week	integer	4	sessions/ week
parental_education_level	Highest education level of parents	string	"Master"	-
internet_quality	Self-reported quality of internet connection	string	"Average"	-
mental_health_rating	Self-reported mental health rating on a 1–10 scale	integer	8	-
extracurricular_participation	Participation in extracurricular activities	string	"Yes"	-
exam_score	Final examination score or performance metric	float	72.6	%

cars.csv

Column Name	Description	Data Type	Example Value	Units
brand	Manufacturer or make of the vehicle	string	"Ford"	-
model	Model name and variant of the vehicle	string	"Utility Police Interceptor Base"	-
model_year	Year the vehicle model was manufactured	integer	2021	-
milage	Distance the vehicle has been driven	integer	34742	miles
fuel_type	Type of fuel used by the vehicle	string	"Gasoline"	-
engine	Engine details including displacement, configuration, and power	string	"3.8L V6 24V GDI DOHC"	-
transmission	Type of transmission	string	"8-Speed Automatic"	-

ext_col	Exterior color of the vehicle	string	"Black"	-
int_col	Interior color of the vehicle	string	"Gray"	-
accident	Accident or damage history of the vehicle	string	"None reported"	-
clean_title	Indicates whether the vehicle has a clean (non-salvage) title	string	"Yes"	-
price	Listed price of the vehicle	float	38005	USD

covid.csv

Column Name	Description	Data Type	Example Value	Units
case_id	Unique identifier assigned to each COVID-19 case	integer	204	-
latitude	Geographic latitude of the reported case location	float	1.306699	-
longitude	Geographic longitude of the reported case location	float	103.847571	-
case_type	Classification of the case based on status	string	"active cases"	-
date_reported	Date when the case was officially reported	date	"17/5/2023"	-

Methodology

1. Populate a dataclass for each data set.
2. Using the builder class and dataclass in step 1, clean and process raw data (e.g. clean missing values, encode categorical columns, clean numeric columns, normalize features, select features, split features/labels).
3. Inspect data if necessary.

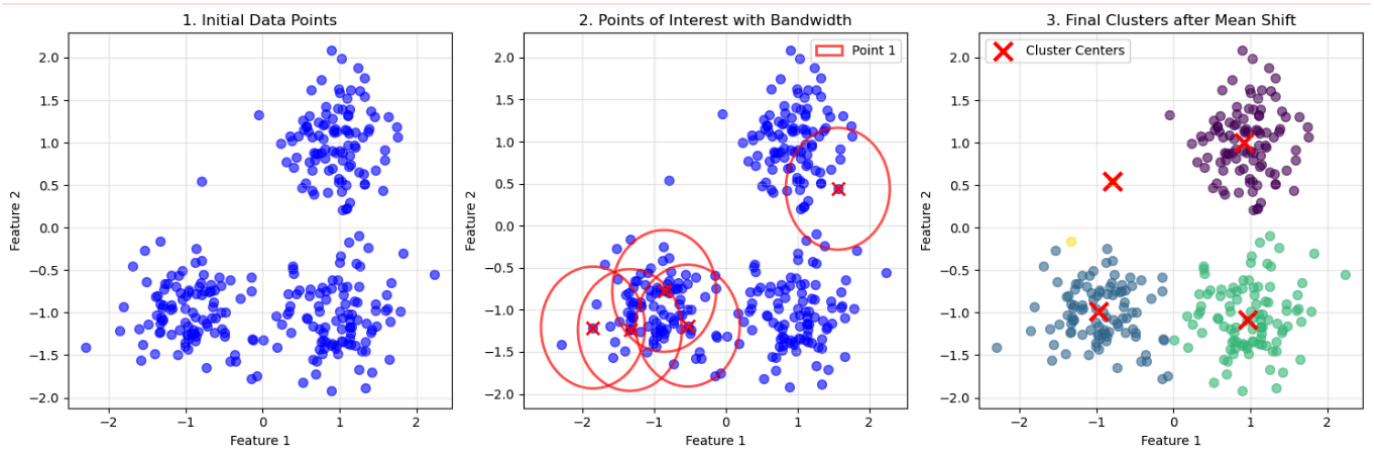
2.2. Means Shift

Means Shift is a clustering algorithm that is used to identify clusters where the number of clusters is not known. It does so by iteratively by finding the mean of the area around the point of interest and then shifting the point to the mode of the area of interest from many different points of interest. After convergence, the area the points of interests converge will then be a cluster.

Key Parameters in Mean Shift

Bandwidth: the radius which is used to calculate the mode of the various data points in the area. A bigger bandwidth suggests a bigger cluster as a wider area is being calculated.

Initial Point: also called a point of interest, it acts as the initial starting point when calculating the area around it.



Mean Shift Algorithm

1. **Define Bandwidth and Point of Interest:** Select a bandwidth and the point of interest based on the data points.
2. **Expand area visited by area of interest:** The mode from the area of interest is calculated and the point of interest is shifted to the mode. Move the bandwidth to the new point of interest.
3. **Stop when convergence is met:** When different points converge to 1 mode, stop and create a cluster from all the areas covered by the different points.

2.3. DBSCAN

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is a clustering algorithm that groups data points that are closed packed together. Dense regions in the data space are identified as clusters and are separated by areas of lower density.

Key Parameters in DBSCAN

eps: eps defines the maximum distance between two points for them to be considered neighbours. Two points are considered neighbours if the distance between two points is less than or equal to eps.

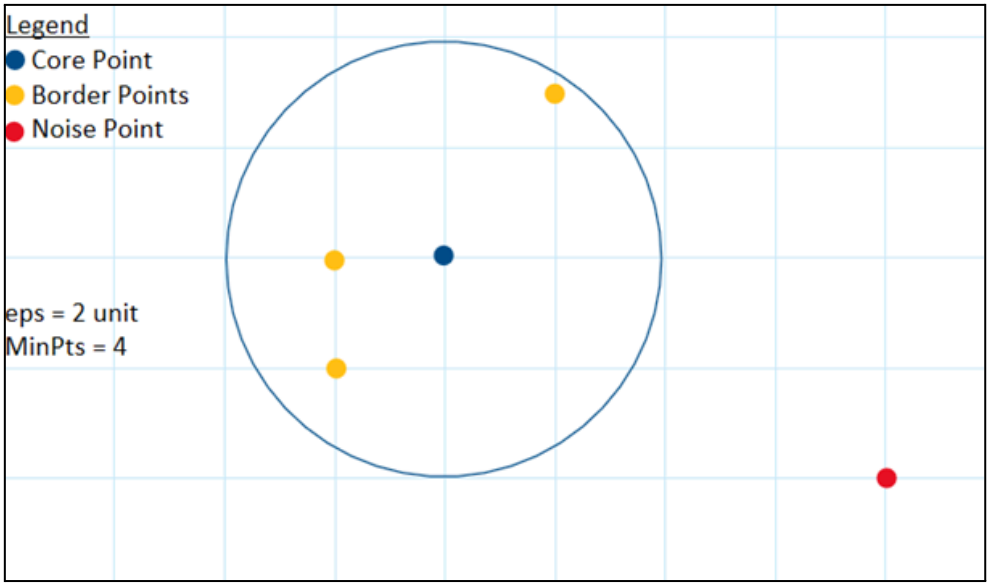
MinPts: MinPts define the minimum number of points required to form a dense region.

Data Point Types in DBSCAN

Core Points: Data points which contain at least MinPts number of neighbours points (including the point itself) within a specified radius eps.

Border Points: Data points which contain fewer than MinPts number of neighbours points but lies within a core point's neighbourhood.

Noise Points: Data points which are not within eps of any core point.



DBSCAN Algorithm

1. **Start a cluster:**
    a. Randomly select an unvisited point in the dataset and count the number of points within its distance eps (the neighbourhood).
    b. If the number of points in that neighbourhood is greater than or equal to MinPts, it is a core point, and a cluster is started (go to step 2).
    c. Otherwise, go to step 1.
2. **Cluster expansion:**
    a. All points within the neighbourhood of this core point are added to the cluster.
    b. If any of those points are also core points, their neighbours are added to the cluster too.
3. **Stop when all points are visited:** Points that do not belong to any cluster are labelled as noise points.

2.4. K-Means

K-Means is a clustering algorithm that splits data into a defined number of clusters k. The algorithm then iteratively assigns each data point to the nearest cluster center (i.e centroid) and updates the centroids based on the mean of the points assigned to that cluster. The algorithm converges once the data points are not assigned to different clusters and the centroids represent the clusters' true centres.

Key Parameters in K-Means

Number of Clusters (k): The number of clusters to initialize. It must be predefined before the algorithm is run.

Initial Centroids: The starting coordinates for the centroids of the clusters. These are either chosen randomly or using probabilistic methods based on furthest separation (K-Means++).

K-Means Algorithm

1. **Initialize the centroids:** Select k initial centroids randomly or using a heuristic algorithm like K-Means++.

2. **Assign points to clusters:** Assign each data point to the nearest centroid based on a distance metric (typically Euclidean distance).
3. **Update centroids:** Recompute the new centroid of each cluster by taking the mean of all data points assigned to that cluster.
4. **Repeat until convergence:** Repeat steps 2 and 3 until the centroids do not change significantly or a maximum number of iterations is reached.

It is worth noting that K-Means assumes the clusters are roughly spherical and of similar size. Outliers can significantly affect the centroids and cluster assignments. Furthermore, the choice of k significantly affects the quality of clustering and can be determined using methods like the Elbow Method or Silhouette Score.

3. Methods

3.1. Means Shift

Mean shift function from scikit-learn is used. Some important parameters in the function are bandwidth, bin_seeding. Bandwidth is the area for calculation of means and is an important parameter in means shift. Bandwidth estimation from scikit-learn is used for estimating bandwidth to use in the main function. Bin_seeding parameter is put as false to speed up the process of the algorithm.

When bin_seeding is turned on, a seeding algorithm is used to determine the initial points of interest to be used by using a grid based approach to reduce the number of initial points being used. Comparison of bin centers vs all data points. With fewer data points to start the clustering process, we are able to speed up the algorithm, providing a boost in time when running bigger datasets such as the weather dataset which contains more than 60000 entries.

3.2. DBSCAN

Parameter Settings

DBSCAN's performance depends heavily on a good choice of eps and MinPts. The following describes the effect of different parameter settings and approaches selecting these parameters:

eps:

When eps is too small, very few points are within the neighbourhood of any point, many points will be labelled as noise, and many small clusters may form.

When eps is too large, almost all points fall within each other's neighbourhood, and clusters merge to form only one big cluster.

A k-distance graph was plotted to select a good eps:

1. The average distance to the k-th nearest neighbour (where k = MinPts) was calculated for each point.
2. These k-distances were plotted in ascending order.
3. An "elbow" point in the graph was identified and the corresponding distance was used as eps.
4. Data points belong to dense regions as they are close to each other before the elbow point. The elbow point indicates that the distance between data points and its neighbours suddenly increases, thus these points are not able to form any cluster and become isolated noises.

MinPts:

When MinPts is too small, random groupings of points may be considered clusters, as it is easy for points to qualify as core points. Noise is reduced, but the resulting clusters may be unreliable.

When MinPts is too large, it becomes harder for points to qualify as core points. Fewer clusters will be formed, and more points will be classified as noise.

MinPts was set to 2 * number of dimensions of the dataset, following the rule of thumb proposed by Sander et al. 1998.

3.3. K-Means

K-Means is used to split the dataset into a specific number of clusters. The key parameters in the algorithm are n_clusters, init, and n_init. n_clusters states the number of clusters to form and must be specified before the algorithm is executed. init determines the method for initializing centroids and can be either randomized or use K-Means++. The K-Means++ algorithm is recommended as it uses smarter initial centroid placement and by having a larger spread this improves convergence and avoids poor local minima. n_init indicates how many times the algorithm will run with different centroid seeds where the best output in terms of inertia (i.e. SSD to the nearest centroid) is kept.

The algorithm iteratively assigns each point to the nearest centroid and then recalculates centroids based on the mean of points in each cluster. This process will repeat iteratively until the centroids stabilize or a maximum number of iterations is reached. K-Means assumes clusters are roughly spherical and of similar size but outliers can heavily influence cluster and centroid positions.

Choosing the right number of clusters is crucial. Methods like the Elbow Method or Silhouette Score are commonly used to determine the optimal value for n_clusters. Proper initialization and multiple runs with n_init can help to improve clustering performance and reduce sensitivity to initial centroid placement. If the model converges in very few iterations, then the initial parameters are appropriately chosen.

3.4. Scoring

Comparison across algorithms was conducted with the same dataset; a smaller dataset (*student.csv*) was selected for the sake of speed.

Score Calculation

After initial computation, scores are normalized to fit on the same scale for plotting on the same graph, which allows for easier viewing and comparison.

Intermediate Formulae

The formulae below are used in some scores under *Final Scoring Formulae*.

Metric	Computation	Interpretation
Mean Intra-Cluster Distance (Jain et al., 1999)	$\text{intra}(C_i) = \frac{2}{\lvert C_i\rvert(\lvert C_i\rvert - 1)} \sum_{x,y \in C_i, x \neq y} \lVert x - y\rVert$ $\text{mean intra-distance} = \frac{1}{k}\sum_{i=1}^{k} \text{intra}(C_i)$	Measures how close points are within the same cluster. Smaller values indicate tighter, more compact clusters. Used to evaluate cluster cohesion.
Mean Inter-Cluster Distance (Kaufman & Rousseeuw, 1990)	$\text{inter}(C_i, C_j) = \frac{1}{\lvert C_i\rvert\lvert C_j\rvert} \sum_{x \in C_i} \sum_{y \in C_j} \lVert x - y\rVert$ $\text{mean inter-distance} = \frac{2}{k(k-1)} \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} \text{inter}(C_i, C_j)$	Measures how far clusters are from each other. Larger values indicate better separation between clusters. Used to evaluate cluster separation.

Final Scoring Formulae

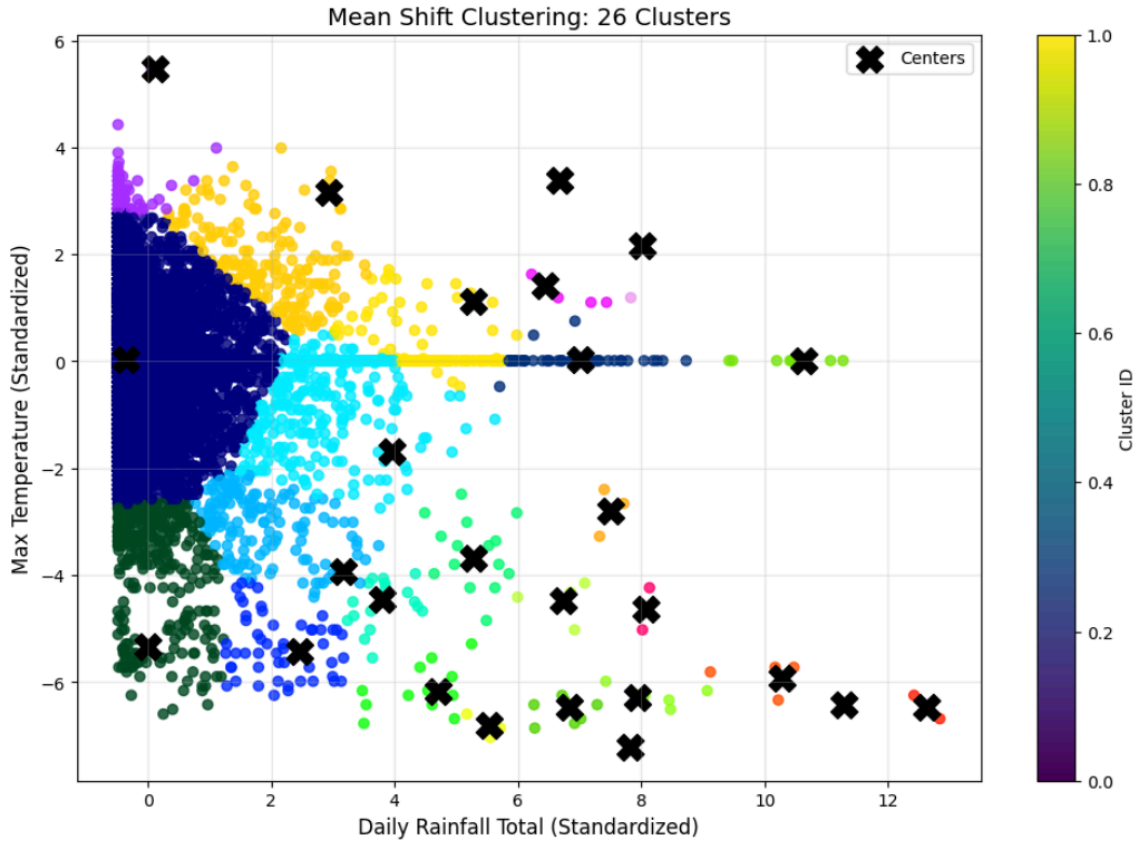
Below shows how scores are calculated before normalization.

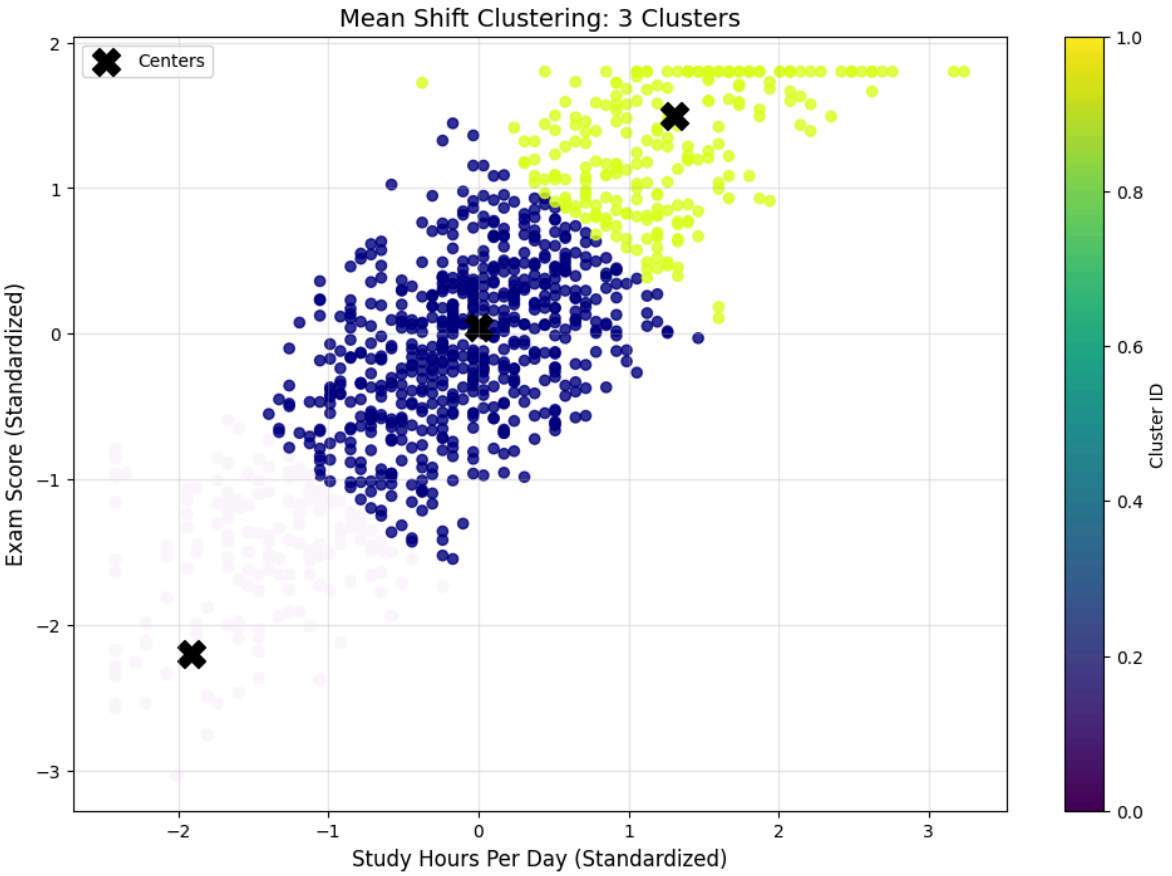
Metric	Computation	Interpretation
Silhouette (Rousseeuw, 1987)	$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$ $s = \frac{1}{n}\sum_{i=1}^{n} s(i)$	Measures how well each point fits in its cluster vs nearest cluster. Range -1 to 1. Higher = better separation.
Davies-Bouldin (Davies & Bouldin, 1979)	$DB = \frac{1}{k}\sum_{i=1}^{k} \max_{j \neq i} \frac{\sigma_i + \sigma_j}{d(c_i, c_j)}$	Ratio of cluster scatter to separation. Lower = better clustering.
Calinski-Harabasz (Calinski & Harabasz, 1974)	$CH = \frac{\mathrm{tr}(B_k)}{\mathrm{tr}(W_k)} \cdot \frac{n-k}{k-1}$ B_k=between-cluster dispersion W_k=within-cluster dispersion	Higher = clusters are dense and well-separated.
Dunn Index (Dunn, 1974)	$DI = \frac{\min_{i \neq j} d(C_i, C_j)}{\max_i \mathrm{diam}(C_i)}$ $d(C_i, C_j) = \min_{x \in C_i, y \in C_j} \|x - y\|$	Higher = well-separated clusters with small intra-cluster spread.
Cluster Size StdDev	$\mathrm{StdDev} = \sqrt{\frac{1}{k}\sum_{i=1}^{k}(\lvert C_i\rvert - \mathrm{mean}(\lvert C\rvert))^2}$	Measures balance of cluster sizes. Lower = more uniform clusters.
Smallest Cluster	$\min_i \lvert C_i\rvert$	Size of the smallest non-noise cluster.
Largest Cluster	$\max_i \lvert C_i\rvert$	Size of the largest non-noise cluster.
Separation Ratio (Halkidi et al., 2002)	$SR = \frac{\text{mean inter-cluster distance}}{\text{mean intra-cluster distance}}$	Higher = clusters are well-separated relative to spread.
Local Outlier Factor (LOF) Score (Breunig et al., 2000)	$LOF(x) = \frac{1}{\lvert N_k(x)\rvert}\sum_{y \in N_k(x)} \frac{\mathrm{lrd}(y)}{\mathrm{lrd}(x)}$	Local Outlier Factor per cluster. Lower/more negative = anomalous.
Isolation Forest Score (Liu et al., 2008)	$IF(x) = -\mathrm{decision\_function}(x)$	Higher = normal, lower = anomalous. Aggregated per cluster.
Silhouette Outlier Fraction (Rousseeuw, 1987)	$\text{fraction} = \frac{\#\{i : s(i) < 0\}}{n}$	Fraction of points poorly assigned. Lower = better.
Noise Fraction - DBSCAN (Ester et al., 1996)	$\text{fraction} = \frac{\#\{i : \mathrm{label}_i = -1\}}{n}$	Fraction of points labeled as noise (DBSCAN). Lower = fewer outliers.
Core-Border Ratio - DBSCAN (Ester et al., 1996)	$\text{ratio} = \frac{\#\text{core points}}{\#\text{clustered points}}$	Higher = more points are dense/core points.

Average Core Distance - DBSCAN (Ester et al., 1996)	$\text{avg} = \frac{1}{k}\sum_{i=1}^{k}\frac{1}{\lvert C_i\rvert^2}\sum_{x,y\in C_i}\lVert x-y\rVert$	Measures cluster compactness. Lower = tighter clusters.
Inertia - KMeans (MacQueen, 1967)	$\text{Inertia} = \sum_{i=1}^{k}\sum_{x\in C_i}\lVert x-\mu_i\rVert^2$	Within-cluster sum of squares. Lower = tighter clusters.
N-Iterations - KMeans (MacQueen, 1967)	$SR = \frac{\text{mean inter-cluster distance}}{\text{mean intra-cluster distance}}$	Fewer iterations = faster convergence.

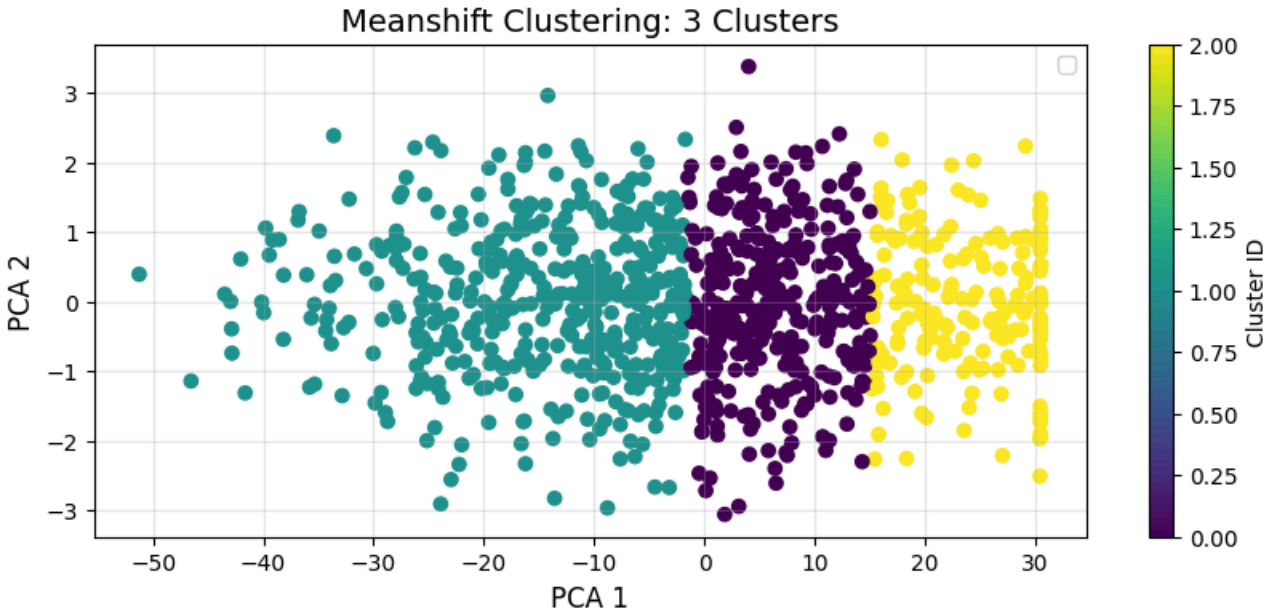
4. Experiments

4.1. Means Shift





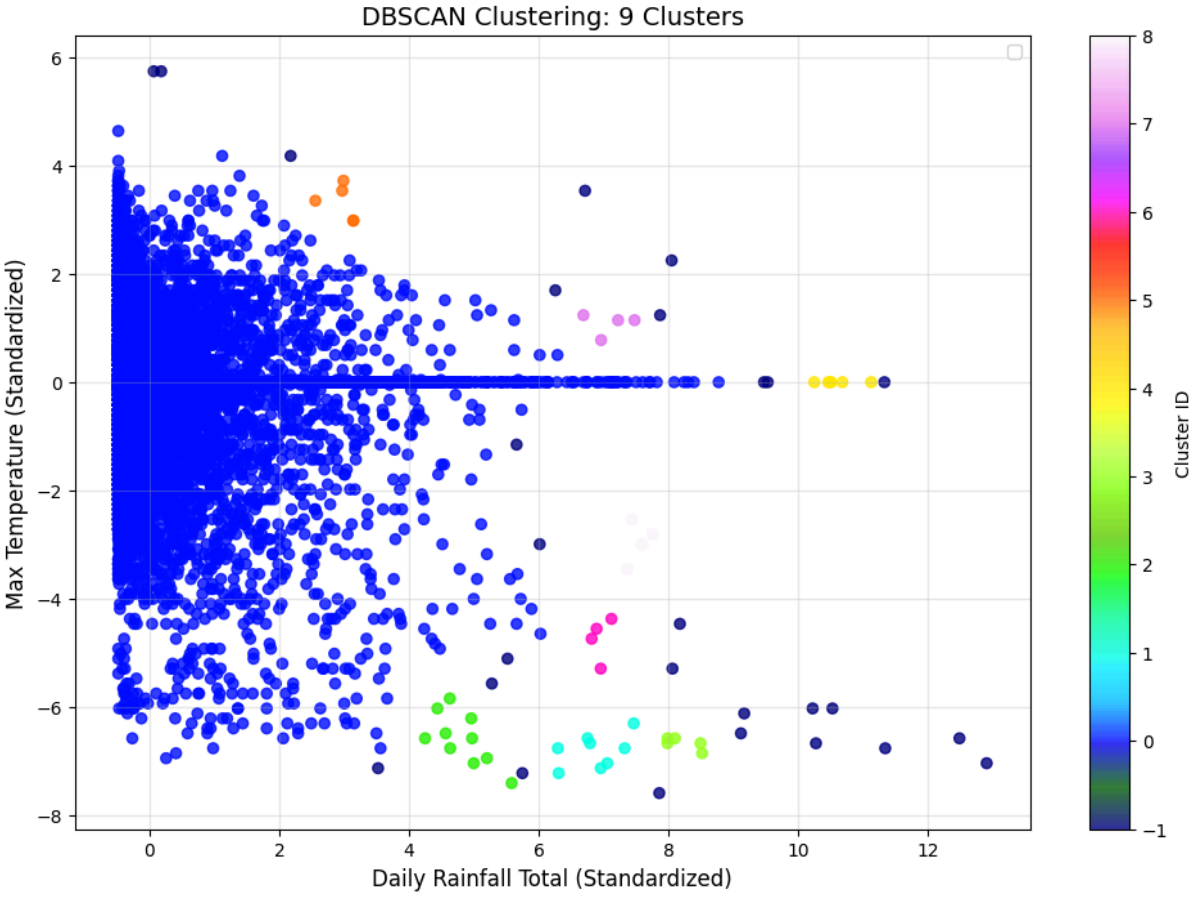
Means Shift is used in 2 datasets of varying shapes and sizes in Student dataset and Weather dataset. The student dataset is more packed and oval in nature while the weather dataset is spread out in an uneven shape, sparse and consists of many outliers. The weather dataset also contains many times more datapoints than in the student dataset, 60000 compared to 1000. Slight tweaks of the bandwidth are done to check if there are any significant changes of clusters we are able to visualize. We can see that the mean shift algorithm performed better in clustering students compared to weather. The student cluster showed a relationship between the study hours per day and the final exam score obtained by students. There are not any easy indications of a relationship from the weather dataset comparing rainfall with temperature and other methods of comparison might be needed before we can make any conclusions.



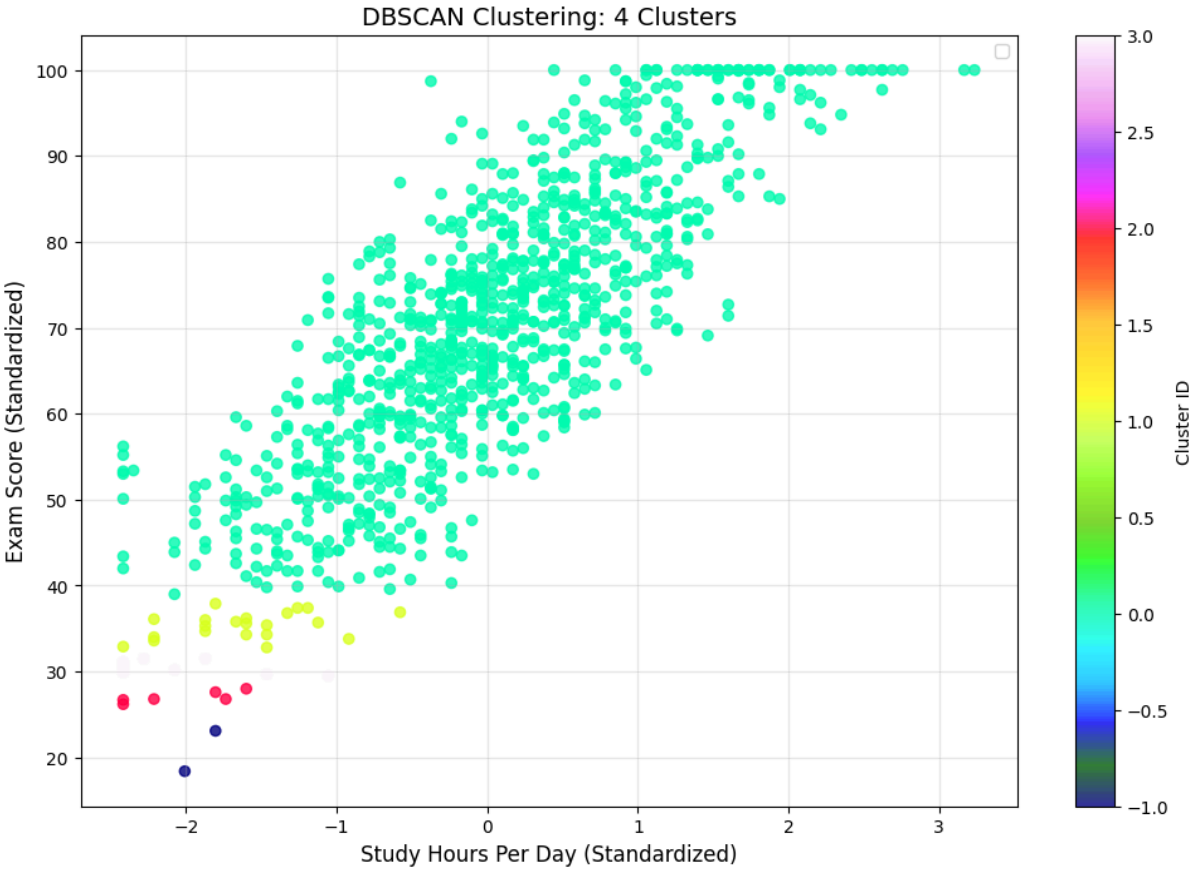
Mean Shift was used to test the curse of higher dimensionality by running it against all parameters from the Student dataset. However in this case, we are unable to determine the result. Other dataset might bring us more insight on the curse of dimensionality.

4.2. DBSCAN

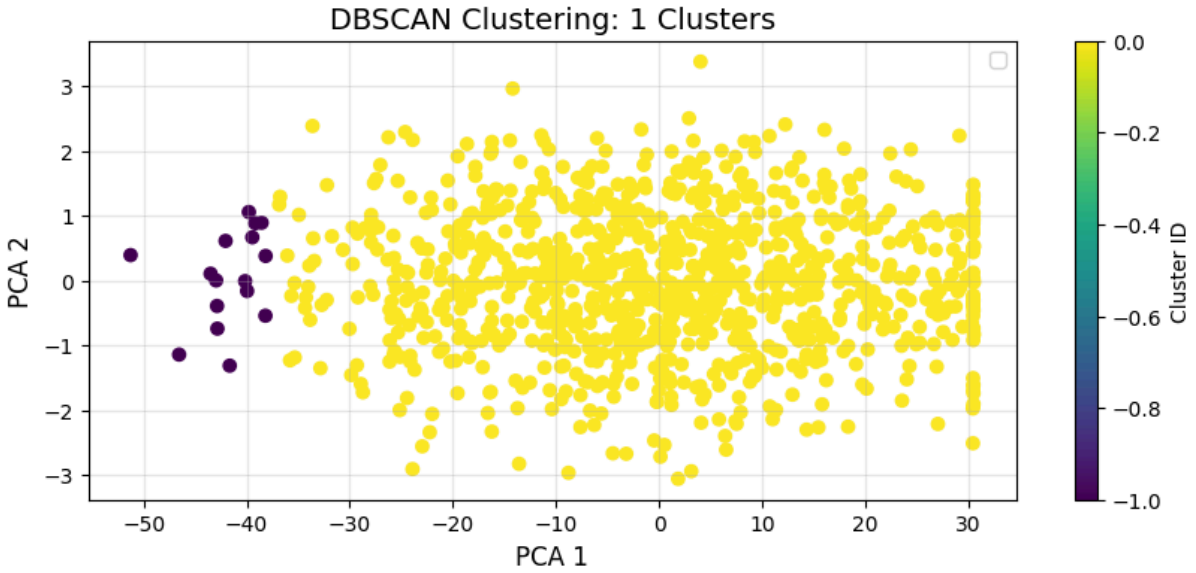
DBSCAN clustering was first performed on the weather dataset, which is a large dataset containing approximately 68,000 records. Given the uneven density of weather data, DBSCAN faced challenges in identifying clear cluster boundaries:



DBSCAN was then applied to a smaller student dataset, using 2 features namely study hour per day and exam score. With these two dimensions, DBSCAN successfully identified meaningful clusters:

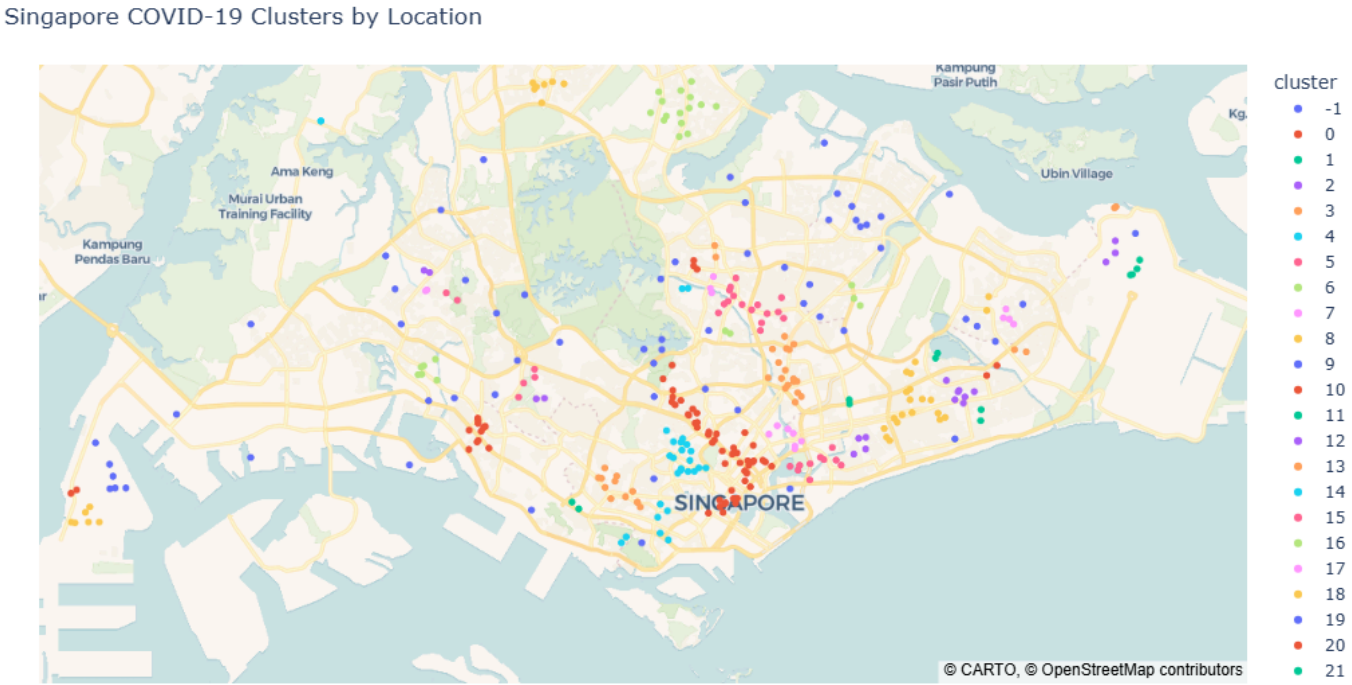


To further test DBSCAN's performance in higher dimensions, the same student dataset was expanded to include 15 attributes and was then projected into two dimensions using Principal Component Analysis (PCA) for visualization. After clustering, DBSCAN failed to produce meaningful or distinct clusters because the distances between points become less meaningful, all points tend to be approximately equally distant from each other. As a result, DBSCAN produced one huge cluster containing most of the data points:



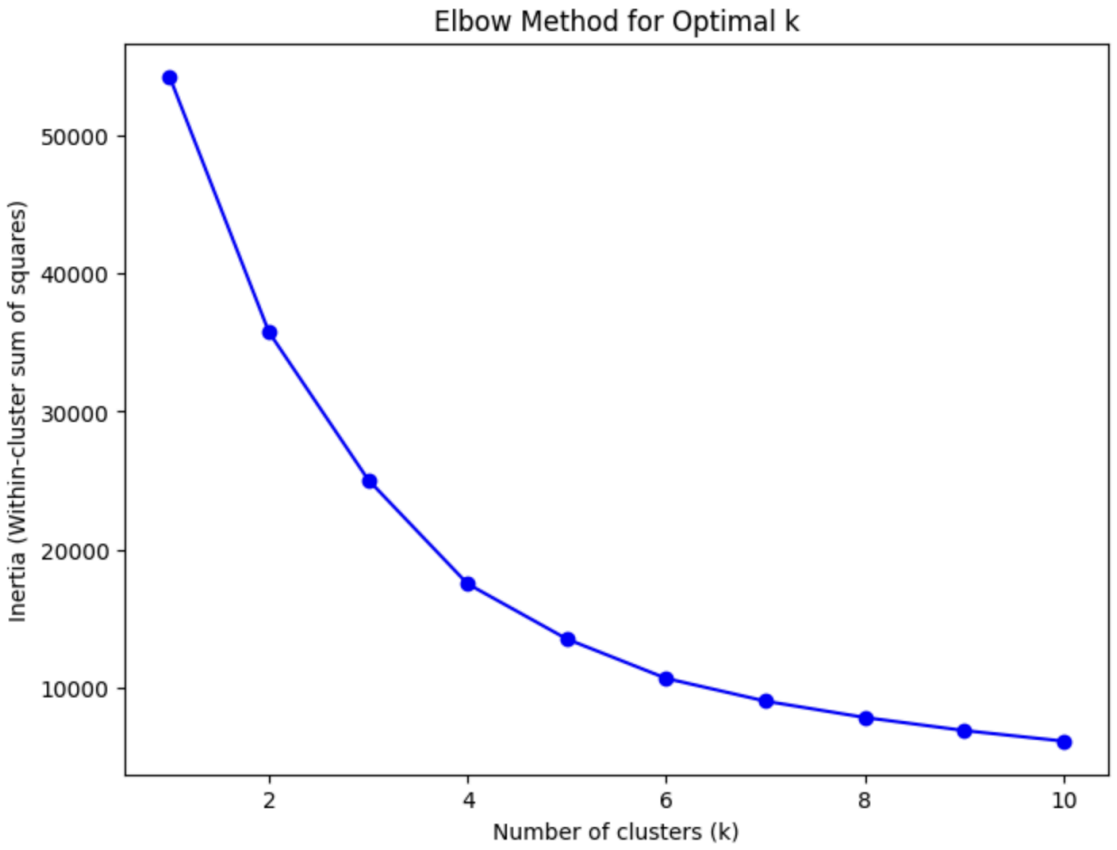
Lastly, a COVID-19 dataset was created to simulate DBSCAN's real-world application. DBSCAN proved to be particularly valuable for detecting outliers in epidemiological data. In the context of COVID-19 case clustering, such outliers often represent unlinked cases, where the source of infection remains unknown. Detecting these cases is important for public health authorities, as it may indicate hidden transmission chains or community spread.

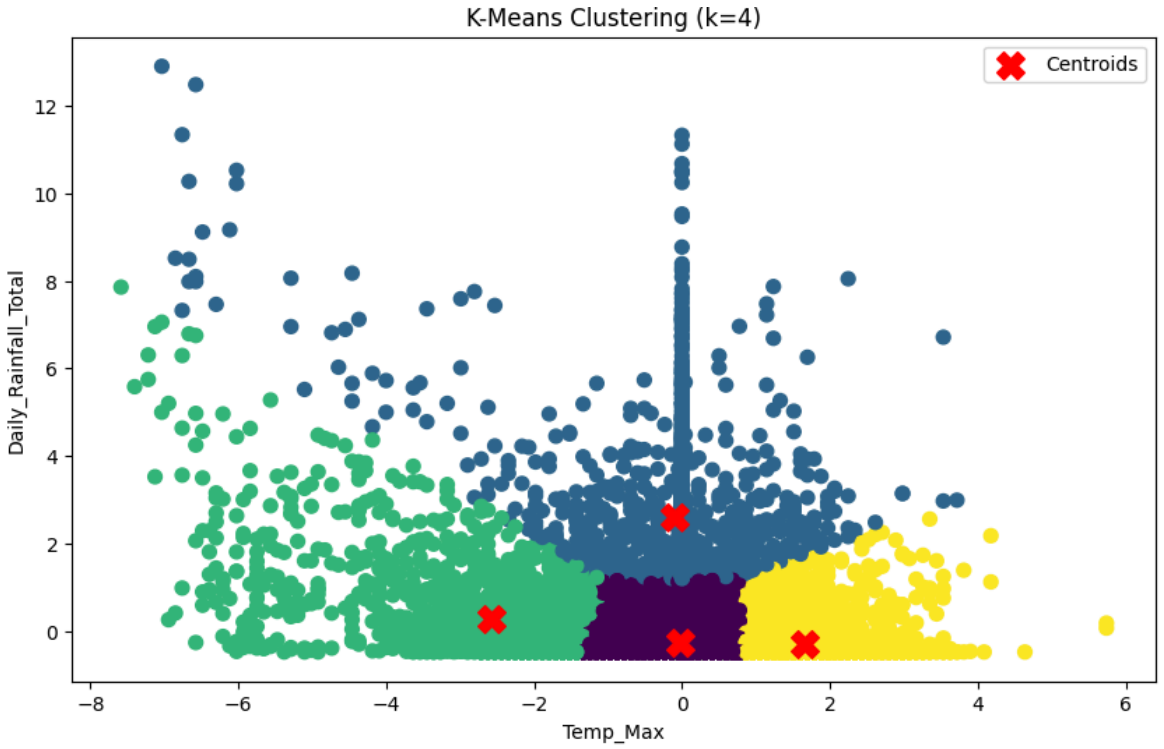
In addition, DBSCAN's ability to identify clusters of irregular and non-spherical shapes makes it well-suited for modeling disease transmission patterns. DBSCAN can effectively capture complex spatial distributions of infection cases. This is particularly relevant because infectious diseases do not spread uniformly—clusters of cases may follow population density rather than forming neat circular patterns:



4.3. K-Means

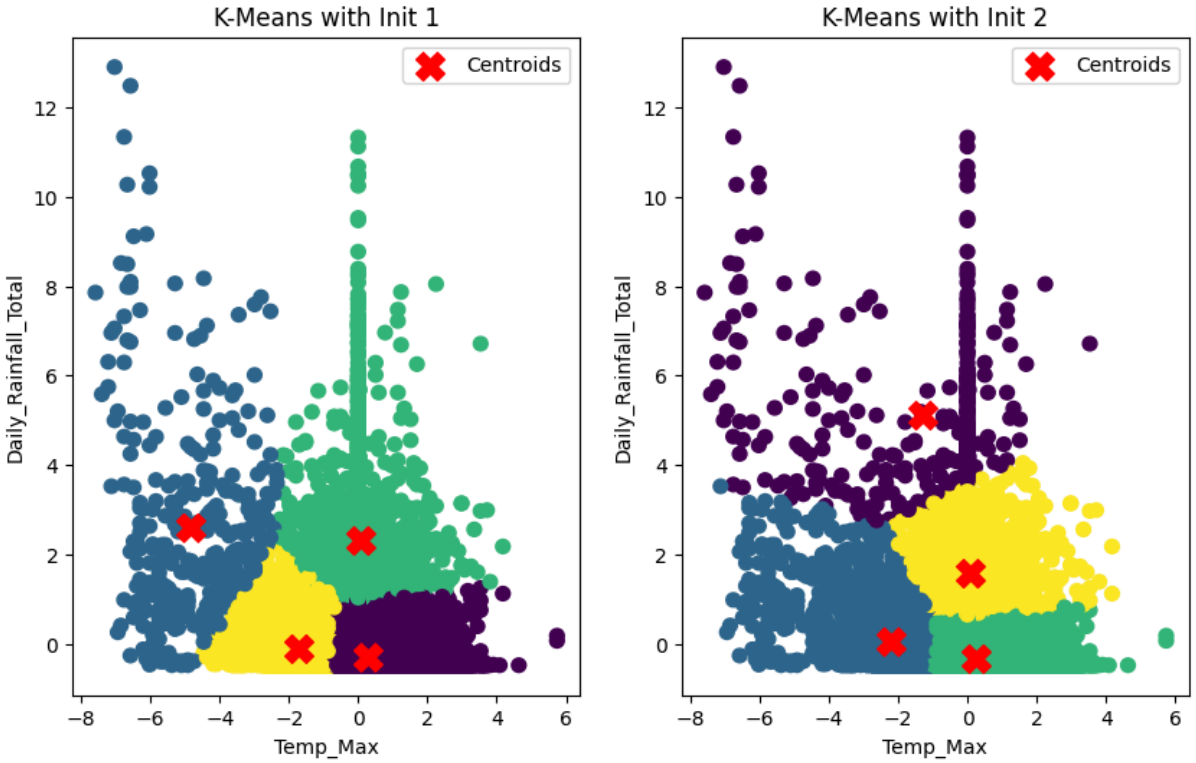
The K-Means algorithm was applied to both the Student & Weather datasets to evaluate its clustering effectiveness given different data characteristics. Using the Elbow method, the optimal number of clusters was determined to be k = 4, as the inertia values slowly started to level out and remained stable. Having too many clusters would result in more complex, unintuitive patterns and might lead to overfitting. The gain in performance would also be marginal at the cost of a model that is too complex.






```
Iterations until convergence: 8
```

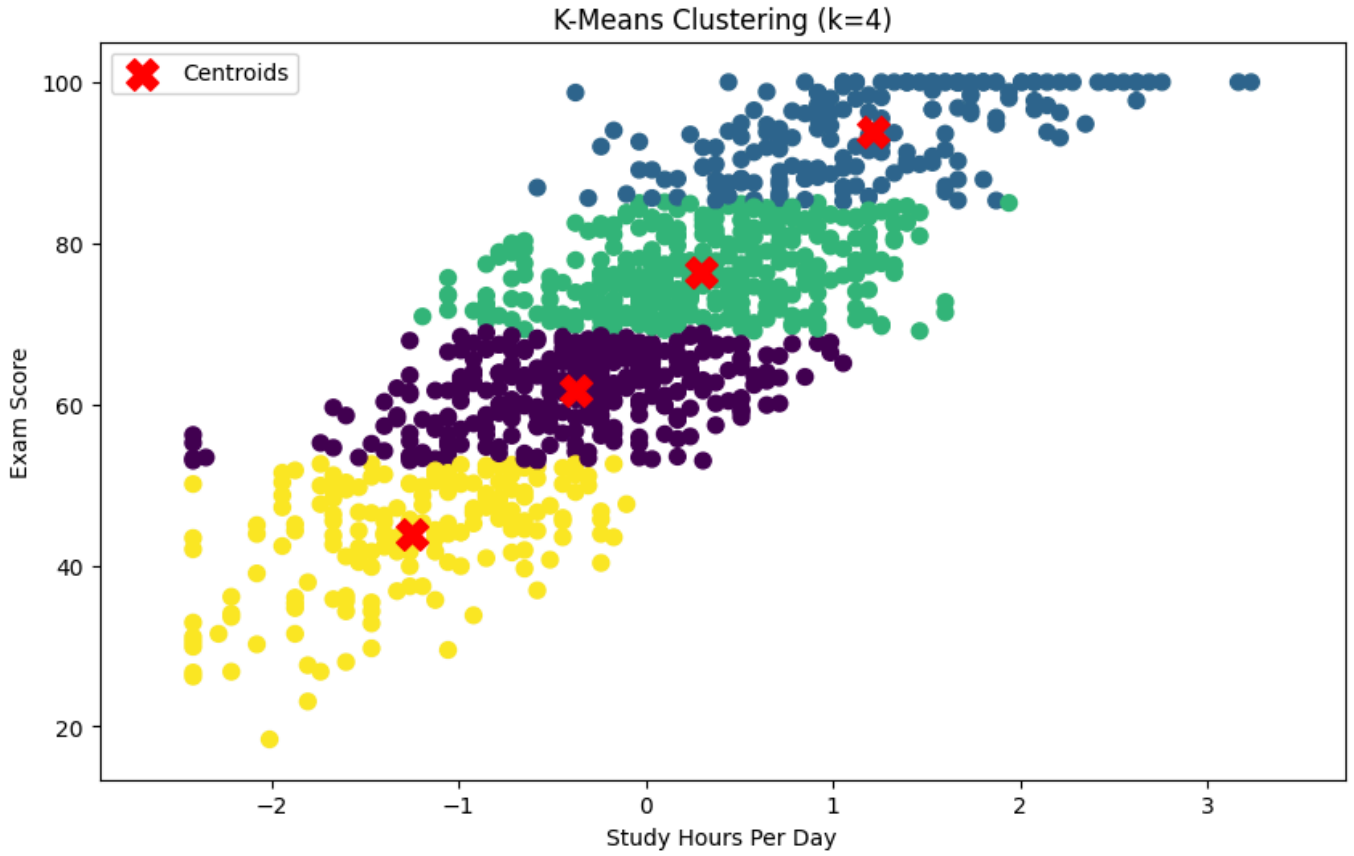
The figure above shows that the algorithm converges within 8 iterations when there are 4 clusters. The relationship between daily maximum temperature and daily total rainfall is not very clear but we can conclude that the two variables have a low correlation. However, K-Means' performance was found to be highly sensitive to centroid initialization as poor initialization occasionally led to poor final results especially in the Weather dataset example above where clusters were unevenly distributed. For the next experiment we will use random coordinates for initialization that are poorly selected to determine the effectiveness of initial centroid placement, as used by K-Means++.




```
Iterations until convergence (left fig): 17
Iterations until convergence (right fig): 26
```

The results above show differing final convergence results just from changing one variable: the coordinates of the 4 initial points.

Next, K-Means was applied on the Student dataset comparing two variables: Exam Score and Study hours per day. K-Means successfully generated four compact and distinct clusters that aligned with expected academic performance trends. The more hours that a student studies per day, the higher their exam score and for this dataset the pattern is most likely linear. The centroids clearly grouped students with varying study hours and exam scores and showed that K-Means performs well on linearly separable, low-dimensional, and dense datasets. However, the Weather dataset shows that K-Means struggles in handling non-spherical or overlapping clusters. The algorithm was unable to assign points meaningfully when data density varied and outliers further affected the final centroid positions.




```
  Iterations until convergence: 4
Centroids:
 [[-3.74232278e-01  6.16906355e+01  9.10382880e-15]
 [ 1.21707596e+00  9.39141304e+01  1.00000000e+00]
 [ 2.96962424e-01  7.63455072e+01  2.00000000e+00]
 [-1.24708466e+00  4.38174419e+01  3.00000000e+00]]
```

K-Means can be applied to work in n-dimensional space however distance metrics tend to become less meaningful due to the curse of dimensionality, ultimately leading to invalid results with no general patterns. This is because data points tend to become more sparse and uniformly distant from one another which makes it challenging for the algorithm to distinguish meaningful clusters based on Euclidean distance for instance.

4.4. Evaluation

See *Appendix 7.1. Cluster Plots* for all plots of clustering for each modified parameter.

This section aims to fine-tune parameters to find the optimized model. In summary, these are the optimal parameters:

Mean Shift	bin_seeding=TRUE, quantile=0.1
DBSCAN	eps≈0.15, min_samples≈2
K-Means	n_clusters≈3, n_init≈any

Each sub-section will go through the trends in each score given the change in as single parameter, holding all other parameters constant:

Means Shift

See *Appendix 7.2. Scores - Means Shift* for score data and graphs.

quantile

Increasing the quantile generally reduces runtime and memory usage but merges clusters, which initially improves but eventually degrades cluster quality metrics (Silhouette, Calinski-Harabasz, Dunn), indicating a trade-off between computational efficiency and meaningful cluster separation.

Metric	Trend with increasing quantile	Interpretation
Runtime (s)	Decreases gradually	Smaller bandwidths at higher quantiles reduce computation time.
Memory usage (MB)	Decreases steadily	High quantiles result in fewer clusters, lowering memory footprint.
Silhouette	Increases slightly then drops sharply	Optimal cluster separation at low quantiles; too high merges clusters, reducing cohesion.
Davies-Bouldin	Decreases initially then collapses	Compact clusters at low quantiles; high quantiles produce poorly defined clusters.
Calinski-Harabasz	Rises initially then drops sharply	Cluster separation improves at low quantiles, then declines as clusters merge.
Dunn Index	Decreases steadily	Inter-cluster distances shrink relative to intra-cluster distances at higher quantiles.
Cluster Size StdDev	Increases then drops to zero	Cluster sizes vary at low quantiles; high quantiles merge clusters into uniform sizes.
Smallest Cluster	Increases rapidly	Smaller clusters are absorbed into larger ones as quantile increases.
Largest Cluster	Increases then plateaus	High quantiles merge clusters, enlarging the largest cluster.

Separation Ratio	Decreases sharply	Cluster distinction diminishes as quantile rises.
LOF Mean Score	Becomes less negative slightly	Outlier impact diminishes as clusters merge.
Isolation Forest Mean Score	Fluctuates mildly	Anomaly detection is slightly stronger or weaker depending on cluster merge effects.
Silhouette Outlier Fraction	Decreases steadily	Fewer poorly clustered points remain as clusters merge into larger groups.

bin_seeding

Enabling bin seeding significantly reduces runtime and memory usage with minimal impact on clustering quality metrics, indicating it improves efficiency without degrading cluster structure or separation.

Metric	**Trend with bin_seeding TRUE → FALSE**	**Interpretation**
Runtime (s)	Decreases slightly	Using bin seeding drastically reduces runtime, improving computational efficiency
Memory usage (MB)	Large increase	Memory footprint is significantly reduced by bin seeding, allowing larger datasets to be handled
Silhouette	Nearly constant	Cluster cohesion and separation are stable regardless of bin seeding
Davies-Bouldin	Slight decrease	Cluster compactness and separation remain almost unchanged
Calinski-Harabasz	Slight increase	Overall cluster separation improves slightly with bin seeding
Dunn	Slight increase	Minimal improvement in worst-case cluster separation
Cluster Size Standard Deviation	Nearly constant	Cluster size balance is unaffected by bin seeding
Smallest Cluster	Constant	Smallest cluster size is stable
Largest Cluster	Slight decrease	Largest cluster size slightly reduces with bin seeding
Separation Ratio	Nearly constant	Relative inter- vs intra-cluster distance does not change meaningfully
LOF Mean Score	Slight decrease	Outlier detection is slightly more sensitive with bin seeding
Isolation Forest Mean Score	Slight decrease	Cluster-weighted anomaly scores slightly improve with bin seeding
Silhouette Outlier Fraction	Slight decrease	Fraction of poorly clustered points reduces marginally

DBSCAN

See *Appendix 7.3. Scores - DBSCAN* for score data and graphs.

eps

Increasing eps generally reduces noise and allows larger, looser clusters, improving some cohesion metrics but also causing cluster size imbalance and occasional degradation in separation at high values.

Metric	Trend with increasing eps	Interpretation
Runtime (s)	Fluctuates slightly, no clear trend	Runtime varies moderately with eps; clustering complexity depends on neighborhood density
Memory usage (MB)	Decreases then slightly increases	Memory footprint is non-monotonic, reflecting different cluster sizes and number of core points
Silhouette	Increases overall with fluctuations	Cluster cohesion improves as eps allows more points to be clustered together
Davies-Bouldin	Decreases then spikes at high eps	Cluster compactness improves initially but deteriorates at very high eps due to merging of clusters
Calinski-Harabasz	Increases slightly then decreases	Overall cluster separation peaks at moderate eps
Dunn	Increases	Worst-case cluster separation improves with larger eps
Cluster Size Standard Deviation	Increases	Cluster sizes become more imbalanced as eps grows
Smallest Cluster	Slightly increases	Smallest clusters grow slowly as more points are included
Largest Cluster	Increases significantly	Largest clusters absorb more points with increasing eps
Separation Ratio	Decreases overall	Relative separation reduces as clusters merge and inter-cluster distances shrink
LOF Mean Score	Decreases (more negative)	Outlier scores indicate more points identified as anomalous with larger eps
Isolation Forest Mean Score	Increases slightly towards zero	Cluster-weighted anomaly detection shows fewer extreme anomalies at high eps
Silhouette Outlier Fraction	Fluctuates	Proportion of poorly clustered points is variable, reflecting cluster merging effects
Noise Fraction	Decreases sharply	More points are assigned to clusters rather than labeled as noise with increasing eps
Core Border Ratio	Increases	Higher fraction of points become core points as eps grows
Average Core Distance	Increases	Average distance between core points increases with larger eps, reflecting looser clusters

min_sample

Increasing min_samples generally increases noise and reduces cluster density, leading to worse cohesion and separation metrics, while shrinking the largest clusters and making core points sparser.

Metric	Trend with increasing min_samples	Interpretation
Runtime (s)	Fluctuates with no clear trend	Runtime is variable; higher min_samples can increase computation initially but may reduce later as fewer clusters form
Memory usage (MB)	Peaks at mid values then decreases	Memory usage is highest when multiple mid-sized clusters exist; fewer clusters at higher min_samples reduce memory
Silhouette	Fluctuates around low negative values	Overall cluster cohesion is poor and sensitive to min_samples, indicating noisy or overlapping clusters
Davies-Bouldin	Increases	Clusters become less compact and more dispersed as min_samples rises
Calinski-Harabasz	Increases then slightly decreases	Cluster separation improves at moderate min_samples but may drop at higher values due to smaller clusters
Dunn	Decreases	Minimum cluster separation worsens with larger min_samples
Cluster Size Standard Deviation	Increases then slightly decreases	Cluster size imbalance grows initially, then stabilizes as clusters shrink at higher min_samples
Smallest Cluster	Increases initially then stabilizes	Smallest clusters grow until min_samples limits point inclusion
Largest Cluster	Decreases	Largest clusters shrink as more points are filtered out with increasing min_samples
Separation Ratio	Decreases	Clusters merge or shrink, reducing relative inter-cluster separation
LOF Mean Score	Increases slightly (less negative)	Fewer extreme outliers detected with higher min_samples
Isolation Forest Mean Score	Decreases	Points are more likely to be considered anomalous with increasing min_samples
Silhouette Outlier Fraction	Fluctuates	Proportion of poorly clustered points varies due to changing cluster definitions
Noise Fraction	Increases	More points are labeled as noise with higher min_samples thresholds
Core Border Ratio	Decreases	Fewer points qualify as core points as min_samples rises
Average Core Distance	Increases	Core points are more spread out as clusters shrink and fewer points meet the core threshold

K-Means

See *Appendix 7.4. Scores - K-Means* for score data and graphs.

n_clusters

K-Means performance stabilizes beyond 6–7 clusters: silhouette and Calinski-Harabasz scores suggest that moderate K yields best structure, while separation ratio and Dunn index modestly improve with higher K.

Metric	**Trend with increasing n_clusters**	**Interpretation**
Runtime (s)	Gradual increase with minor fluctuations	More clusters slightly increase computational cost due to repeated centroid assignment and updates.
Memory usage (MB)	Initial rise then stabilizes	Memory usage grows early as centroid count increases, but plateaus once internal structures are reused efficiently.
Silhouette	Steady decline then stabilizes around low value	Cluster cohesion and separation worsen as clusters become smaller and less distinct, then level off beyond ~7 clusters.
Davies-Bouldin	Increases until ~6 clusters, then improves slightly	Higher values early indicate poorer separation; after moderate K, clusters stabilize and overlap reduces slightly.
Calinski-Harabasz	Rises to a peak around 3–4 clusters, then gradually declines	Optimal cluster separation likely occurs at moderate K, after which over-partitioning reduces global distinctness.
Dunn Index	Fluctuates but generally improves beyond K≈6	Some increase in inter-cluster separation at higher K, though gains are inconsistent and small.
Cluster Size StdDev	Sharp decrease as K increases, then stabilizes	Clusters become more balanced in size as points are distributed across more groups.
Smallest Cluster	Monotonic decrease	Smallest clusters get progressively smaller as total cluster count increases.
Largest Cluster	Consistent decrease	Largest clusters lose members steadily, confirming more even data distribution among clusters.
Separation Ratio	Strong monotonic increase	Average inter-cluster distance grows relative to intra-cluster spread, indicating better geometric separation.
LOF Mean Score	Stable around −1.05 with negligible variation	Local outlier structure remains constant; clustering granularity does not affect local density anomalies.
Isolation Forest Mean Score	Slightly improves (less negative) up to K≈9, then minor noise	Outlier intensity decreases as finer clustering captures local variations better.
Silhouette Outlier Fraction	Stable with minor oscillations (0–0.017)	Boundary points vary slightly but the overall model remains consistent in handling ambiguous samples.
Inertia	Monotonic decline	Within-cluster variance decreases predictably as more centroids fit data more closely.

Iterations Until Convergence	Fluctuates randomly between 5–24	Convergence variability is driven by initialization randomness, not systematic with cluster count.

n_init

Increasing n_init mostly affects runtime and memory, while clustering quality metrics remain unchanged, indicating the chosen number of clusters is robust and K-Means consistently finds similar solutions across different initializations.

Metric	**Trend with increasing n_init**	**Interpretation**
Runtime (s)	Increases initially up to n_init=50, then drops slightly	Runs take longer as more initializations are computed; slight drop at 100 may reflect early convergence in some runs.
Memory usage (MB)	Rises with higher n_init, peaks at 50, then decreases	More initializations consume memory for storing multiple centroids; at very high n_init, memory usage may optimize internally.
Silhouette	Constant at 0.382756	Cluster cohesion and separation remain stable; multiple initializations converge to similar quality.
Davies-Bouldin	Constant at 0.825894	Inter-cluster overlap and compactness unaffected by number of initializations.
Calinski-Harabasz	Constant at 1.491904	Global separation remains unchanged with more initializations.
Dunn Index	Constant at 0.010607	Relative inter- vs intra-cluster distances are stable.
Cluster Size StdDev	Constant at 77.4177	Cluster size distribution unaffected by n_init.
Smallest Cluster	Constant at 159	No change in smallest cluster size.
Largest Cluster	Constant at 335	No change in largest cluster size.
Separation Ratio	Constant at 0.360613	Inter-cluster separation relative to intra-cluster spread remains constant.
LOF Mean Score	Constant at −1.05403	Local density anomaly detection unaffected.
Isolation Forest Mean Score	Minor fluctuations	Outlier detection varies slightly, but no clear trend.
Silhouette Outlier Fraction	Constant at 0	Boundary points remain unchanged.
Inertia	Constant at 364.0563	Within-cluster variance unaffected by additional initializations.
Iterations Until Convergence	Constant at 14	Convergence speed is stable across n_init values.

5. Conclusion

5.1. Means Shift

Meanshift is shown to be better when the dataset is in a certain shape that is more circular in nature. One good point is that we do not need to specify the number of clusters. The disadvantages are, firstly it is very slow when the number of data points gets significantly larger if bin seeding is not turned on as all data points will be used. Secondly, it performs very badly in uneven shapes and data with many outliers.

5.2. DBSCAN

DBSCAN is an effective clustering algorithm that excels at detecting clusters of arbitrary shapes and identifying outliers without needing a predefined number of clusters. It performs well on low- to medium-dimensional datasets, as shown in the student and COVID-19 examples. However, its performance heavily depends on the proper selection of parameters ε (eps) and MinPts, which determine how clusters are formed. Using techniques like the elbow method for eps and the rule of thumb (MinPts = 2 × dimensions) helps improve results. Despite its strengths, DBSCAN struggles with datasets of varying densities and high dimensions.

5.3. K-Means

K-Means performs well on separable, structured datasets and promises quick convergence and comprehensible results. It relies heavily on the chosen hyperparameters like the number of clusters (k) which allows for flexibility but can lead to inaccurate groupings if k is poorly chosen. Additionally, the algorithm assumes spherical clusters of similar density thus making it less effective for irregular, overlapping, or noisy data. K-Means remains a powerful and practical clustering algorithm for discovering general patterns in organized datasets which consist of relatively uniform distributions.

5.4. Overall

Mean Shift achieves the best separation and handles outliers well but at moderate computational cost.
K-Means is fast, stable, and produces cohesive clusters but assumes spherical shapes. DBSCAN identifies arbitrary-shaped clusters but produces more noise and less compact clusters, sensitive to eps and min_samples.

Metric	Mean Shift	DBSCAN	K-Means	Explanation
Runtime	Moderate	Moderate	Low	Mean Shift performs kernel density estimation, DBSCAN checks neighbors, K-Means is a simple iterative centroid update.
Memory usage	Moderate	Moderate	Low	Mean Shift and DBSCAN store pairwise distances, K-Means only stores centroids and assignments.
Silhouette	Good	Low to moderate	Moderate	Mean Shift produces well-separated clusters; DBSCAN has noise and uneven clusters; K-Means forms roughly spherical clusters.

Davies-Bouldin	Low	Moderate to high	Low	Better separation and cohesion in Mean Shift and K-Means; DBSCAN clusters less compact.
Calinski-Harabasz	High	Low	High	Mean Shift and K-Means optimize variance-based separation; DBSCAN sensitive to density variations.
Dunn	Moderate	Moderate	Low	Mean Shift balances cluster separation; DBSCAN varies with eps; K-Means splits some natural clusters.
Cluster Size StdDev	High	Moderate to high	Moderate	Mean Shift produces uneven clusters; DBSCAN clusters vary due to density; K-Means balances sizes more.
Smallest Cluster	Moderate	Small	Moderate	DBSCAN can produce very small clusters; K-Means and Mean Shift avoid extremely tiny clusters.
Largest Cluster	Moderate	Large	Moderate	DBSCAN may group dense points into very large clusters.
Separation Ratio	High	Low	Moderate	Mean Shift maximizes distance between clusters; DBSCAN clusters can be close; K-Means moderately spaced.
LOF Mean Score	Low	Moderate	Low	K-Means and Mean Shift handle noise better; DBSCAN labels more points as noise.
Isolation Forest Score	Low	Moderate	Low	DBSCAN identifies more anomalies; K-Means/Mean Shift produce denser, cohesive clusters.
Silhouette Outlier Fraction	Low	Moderate	Low	DBSCAN has more negative silhouette points due to noise; others are more cohesive.

6. References

Breunig, M. M., Kriegel, H., Ng, R. T., & Sander, J. (2000). LOF. ACM SIGMOD Record, 29(2), 93–104. https://doi.org/10.1145/335191.335388

Calinski, T., & Harabasz, J. (1974). A dendrite method for cluster analysis. Communication in Statistics-Theory and Methods, 3(1), 1–27. https://doi.org/10.1080/03610927408827101

Davies, D. L., & Bouldin, D. W. (1979). A cluster separation measure. IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-1(2), 224–227. https://doi.org/10.1109/tpami.1979.4766909

Dunn, J. C. (1974). Well-Separated clusters and optimal fuzzy partitions. Journal of Cybernetics, 4(1), 95–104. https://doi.org/10.1080/01969727408546059

Ester, M., Kriegel, H., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD '96) (pp. 226–231). AAAI Press. https://dl.acm.org/doi/10.5555/3001460.3001507

GeeksforGeeks. (2025, September 12). DBSCAN Clustering in ML Density based clustering. GeeksforGeeks. https://www.geeksforgeeks.org/machine-learning/dbscan-clustering-in-ml-density-based-clustering/

Halkidi, M., Batistakis, Y., & Vazirgiannis, M. (2002). Cluster validity methods. ACM SIGMOD Record, 31(2), 40–45. https://doi.org/10.1145/565117.565124

Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering. ACM Computing Surveys, 31(3), 264–323. https://doi.org/10.1145/331499.331504

Kaufman, L., & Rousseeuw, P. J. (1990). Finding groups in data. In Wiley series in probability and statistics. https://doi.org/10.1002/9780470316801

Liu, F. T., Ting, K. M., & Zhou, Z. (2008). Isolation Forest. 2008 Eighth IEEE International Conference on Data Mining, Pisa, Italy, 413–422. https://doi.org/10.1109/icdm.2008.17

MacQueen, J. (1967). Multivariate observations. In Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability (Vol. 1, pp. 281-297). https://www.cs.cmu.edu/~bhiksha/courses/mlsp.fall2010/class14/macqueen.pdf

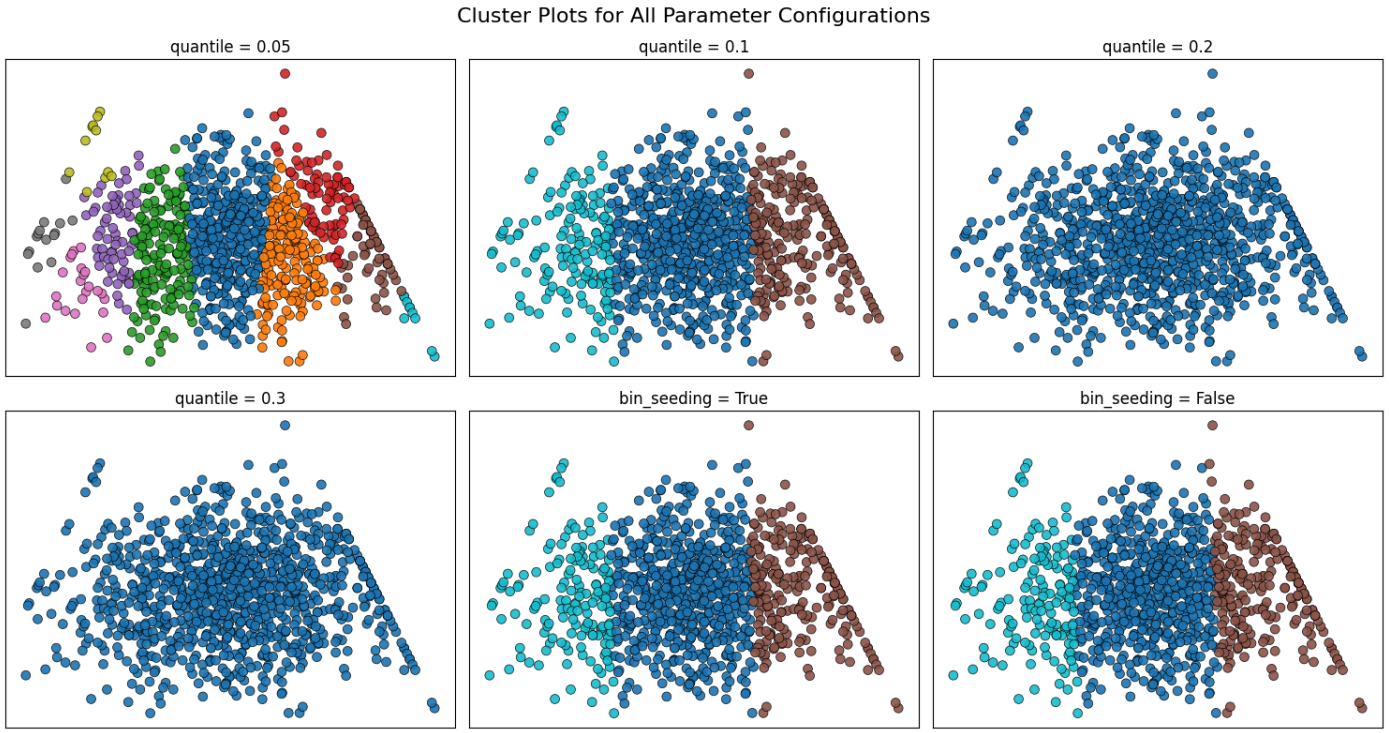
Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics, 20, 53–65. https://doi.org/10.1016/0377-0427(87)90125-7

Sander, J., Ester, M., Kriegel, H., & Xu, X. (1998). Density-Based Clustering in Spatial Databases: The Algorithm GDBSCAN and Its Applications. Data Mining and Knowledge Discovery, 2(2), 169–194. https://doi.org/10.1023/a:1009745219419

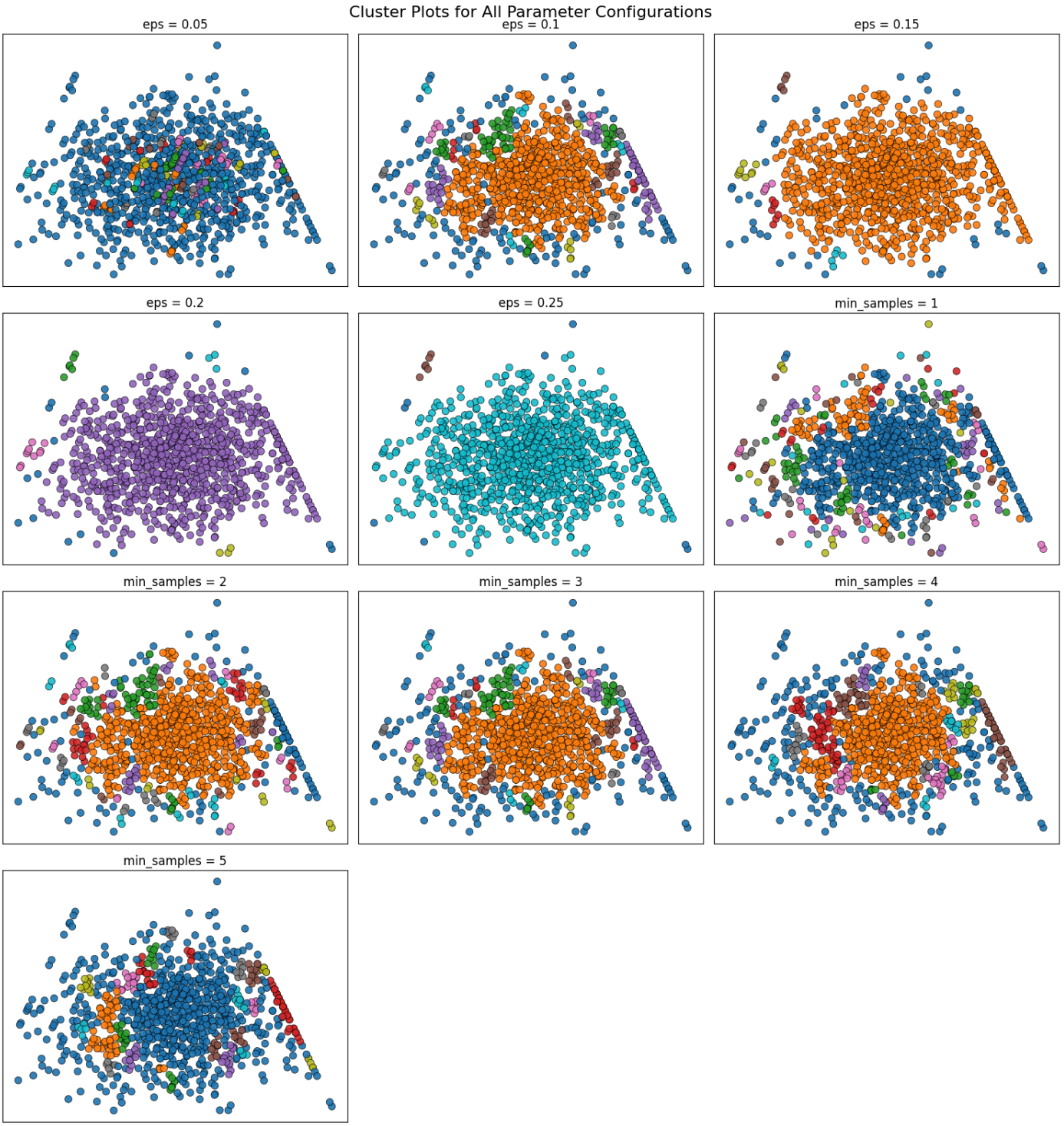
7. Appendix

7.1. Cluster Plots

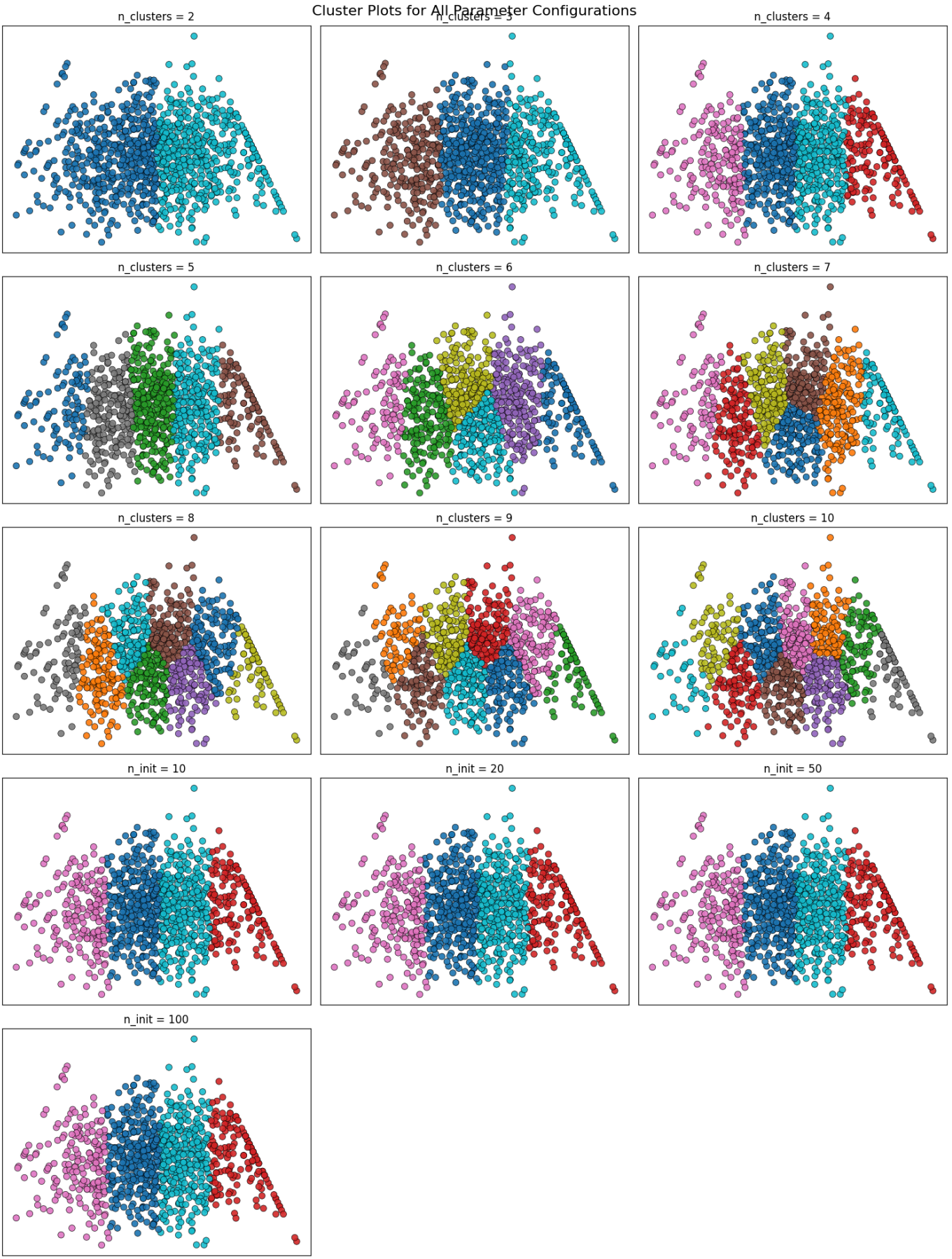
Means Shift



DBSCAN


Cluster Plots for All Parameter Configurations
eps = 0.05
eps = 0.1
eps = 0.15
eps = 0.2
eps = 0.25
min_samples = 1
min_samples = 2
min_samples = 3
min_samples = 4
min_samples = 5

K-Means

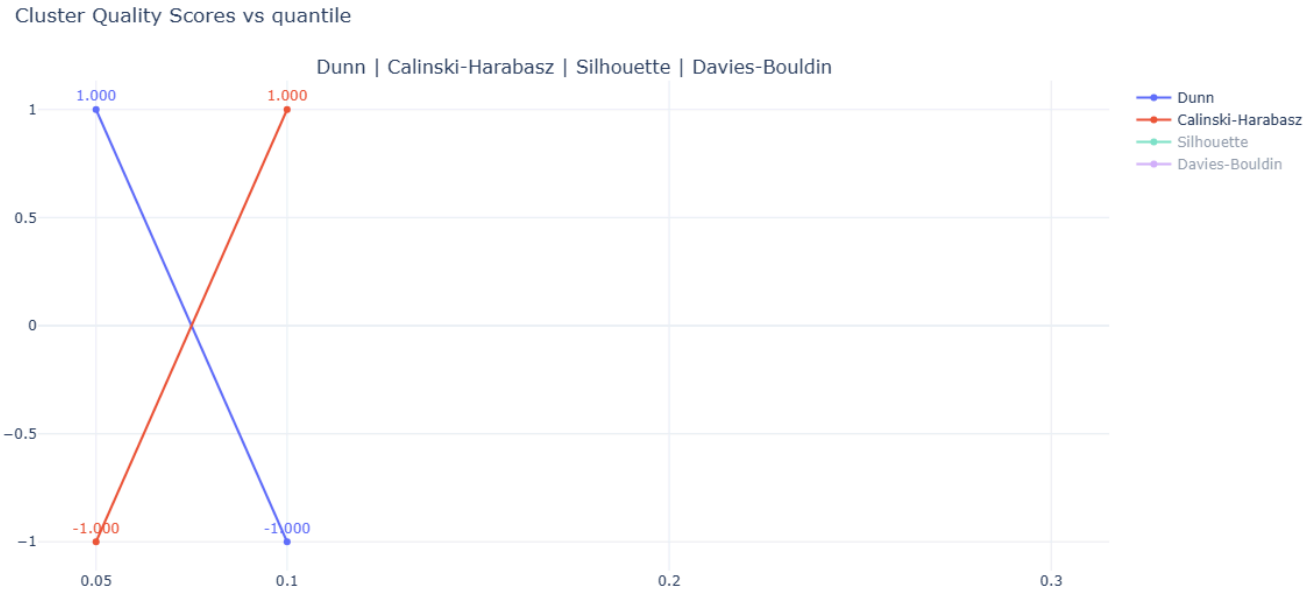

Cluster Plots for All Parameter Configurations
n_clusters = 2
n_clusters = 3
n_clusters = 4
n_clusters = 5
n_clusters = 6
n_clusters = 7
n_clusters = 8
n_clusters = 9
n_clusters = 10
n_init = 10
n_init = 20
n_init = 50
n_init = 100

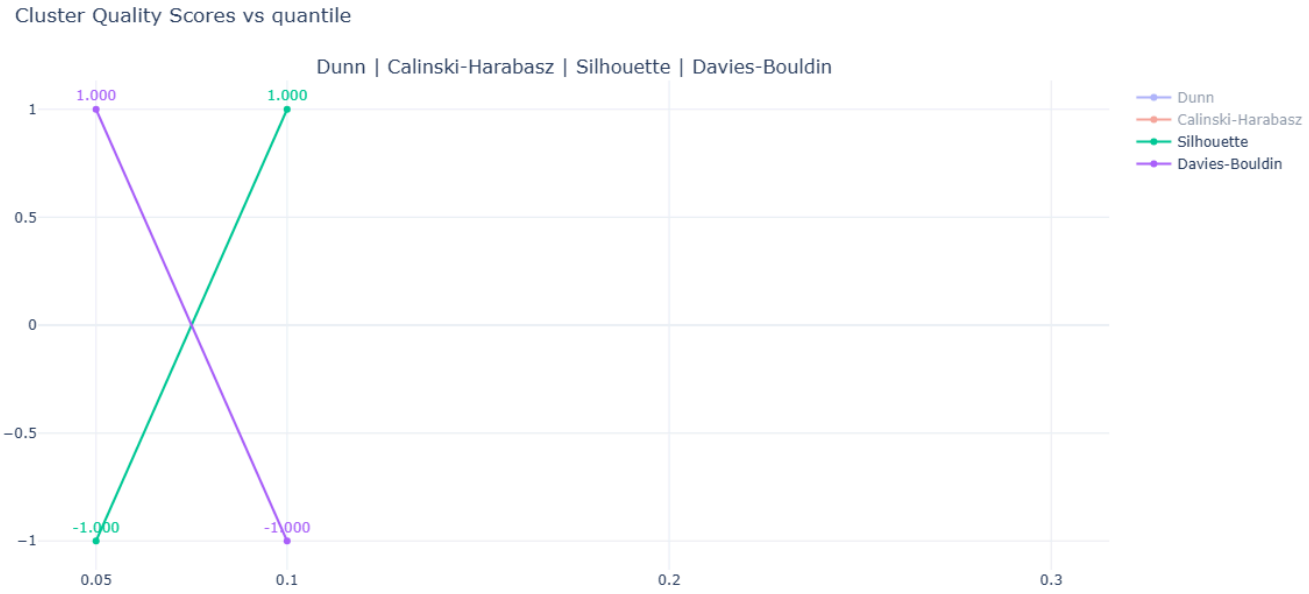
7.2. Scores - Means Shift

Means Shift Raw Data (quantile = variable, bin_seeding = true)

Score	Parameter: *quantile*			
	0.05	**0.1**	**0.2**	**0.3**
Runtime (s)	1.9083	1.5381	1.3598	1.2201
Memory usage (MB)	2888.523	2334.3	2007.847	1779.344
Silhouette	0.299181	0.414999	0	0
Davies-Bouldin	0.85543	0.687987	0	0
Calinski-Harabasz	0.818598	1.299265	0	0
Dunn	0.013496	0.009844	0	0
Cluster Size Standard Deviation	111.1216	198.3504	0	0
Smallest Cluster	10	160	1000	1000
Largest Cluster	383	611	1000	1000
Separation Ratio	1.312526	0.284287	0	0
LOF Mean Score	-1.13905	-1.05979	-1.04966	-1.04966
Isolation Forest Mean Score	-0.04731	-0.0932	-0.11167	-0.11291
Silhouette Outlier Fraction	0.112	0.1	0	0

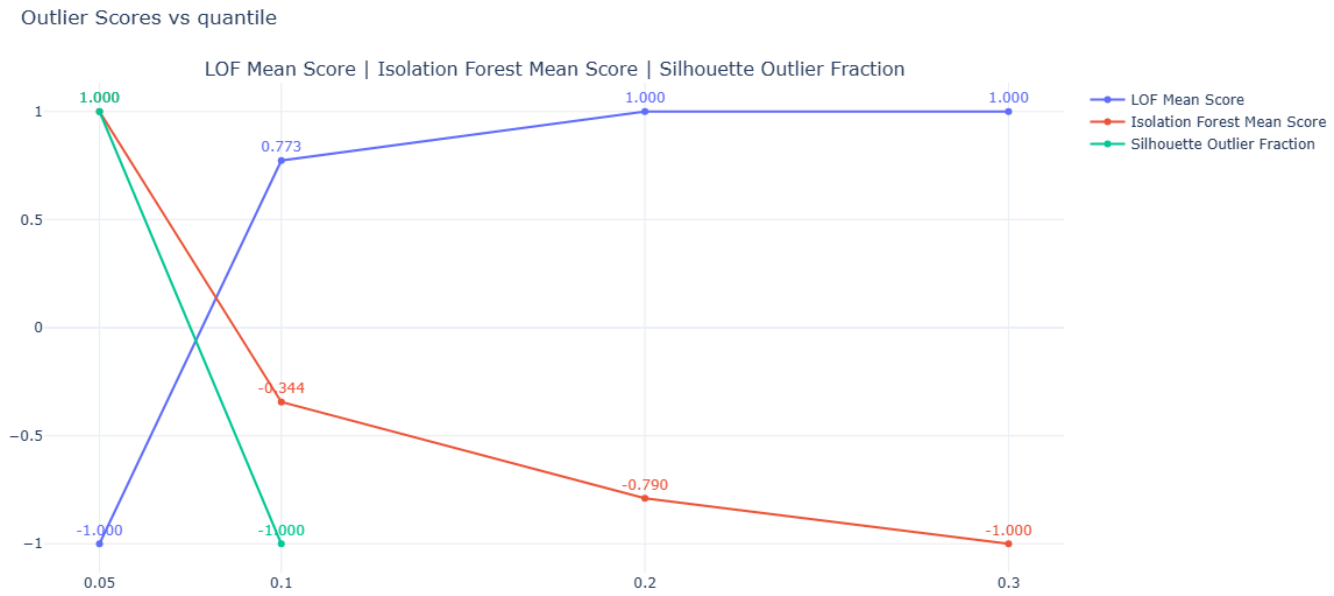
Means Shift Graphs (quantile = variable, bin_seeding = true)


Cluster Quality Scores vs quantile
Dunn | Calinski-Harabasz | Silhouette | Davies-Bouldin
1.000
1.000
-1.000
-1.000
1
0.5
0
−0.5
−1
0.05
0.1
0.2
0.3
Dunn
Calinski-Harabasz
Silhouette
Davies-Bouldin


Cluster Quality Scores vs quantile
Dunn | Calinski-Harabasz | Silhouette | Davies-Bouldin
1.000
1.000
-1.000
-1.000
1
0.5
0
−0.5
−1
0.05
0.1
0.2
0.3
Dunn
Calinski-Harabasz
Silhouette
Davies-Bouldin

Performance vs quantile

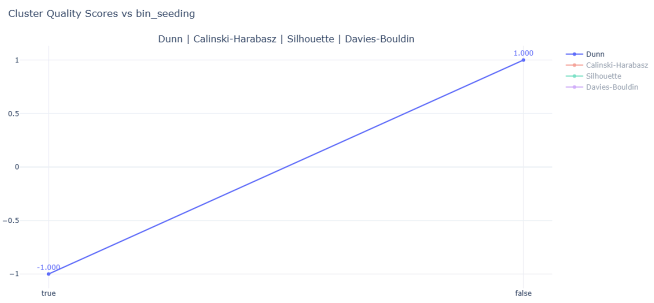
Cluster Sizes vs quantile



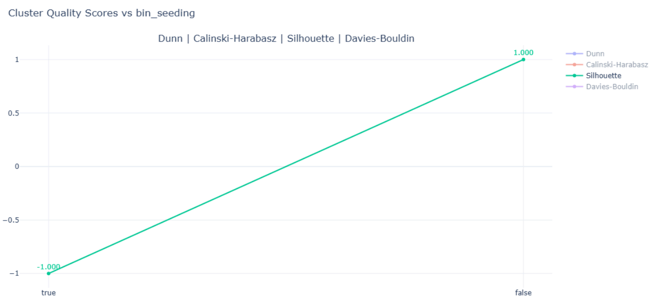
Means Shift Raw Data (quantile = 0.1, bin_seeding = variable)

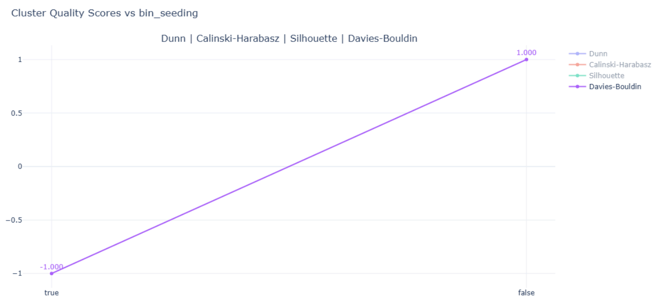
Score	**Parameter: *bin_seeding***	
	TRUE	**FALSE**
Runtime (s)	1.5381	17.8225
Memory usage (MB)	2334.3	27555.53
Silhouette	0.414999	0.416356
Davies-Bouldin	0.687987	0.691472
Calinski-Harabasz	1.299265	1.309705
Dunn	0.009844	0.011485
Cluster Size Standard Deviation	198.3504	194.5222
Smallest Cluster	160	160
Largest Cluster	611	605
Separation Ratio	0.284287	0.284471
LOF Mean Score	-1.05979	-1.06044
Isolation Forest Mean Score	-0.0932	-0.09985
Silhouette Outlier Fraction	0.1	0.097

Means Shift Graphs (quantile = 0.1, bin_seeding = variable)


Cluster Quality Scores vs bin_seeding
Dunn | Calinski-Harabasz | Silhouette | Davies-Bouldin
Dunn
Calinski-Harabasz
Silhouette
Davies-Bouldin
1
0.5
0
−0.5
−1
1.000
-1.000
true
false

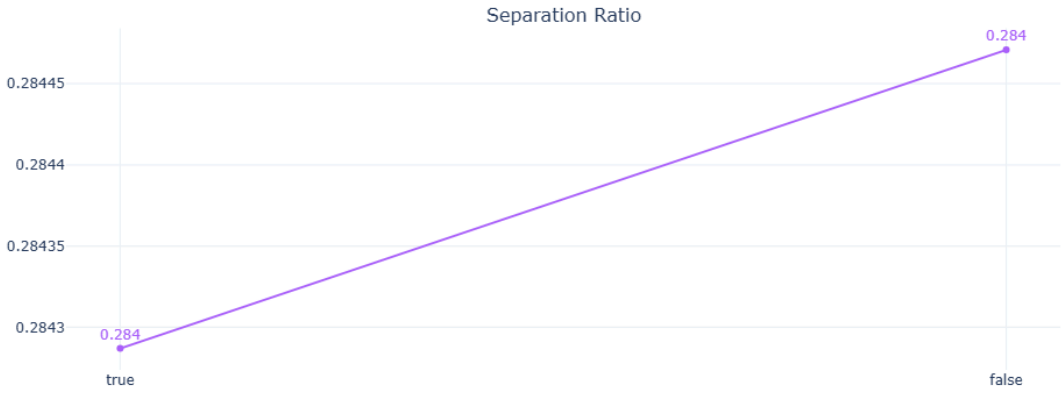

Cluster Quality Scores vs bin_seeding
Dunn | Calinski-Harabasz | Silhouette | Davies-Bouldin
Dunn
Calinski-Harabasz
Silhouette
Davies-Bouldin
1
0.5
0
−0.5
−1
1.000
-1.000
true
false


Cluster Quality Scores vs bin_seeding
Dunn | Calinski-Harabasz | Silhouette | Davies-Bouldin
Dunn
Calinski-Harabasz
Silhouette
Davies-Bouldin
1
0.5
0
−0.5
−1
1.000
-1.000
true
false


Cluster Quality Scores vs bin_seeding
Dunn | Calinski-Harabasz | Silhouette | Davies-Bouldin
Dunn
Calinski-Harabasz
Silhouette
Davies-Bouldin
1
0.5
0
−0.5
−1
1.000
-1.000
true
false

Performance vs bin_seeding

Cluster Sizes vs bin_seeding



Outlier Scores vs bin_seeding

Outlier Scores vs bin_seeding

Outlier Scores vs bin_seeding

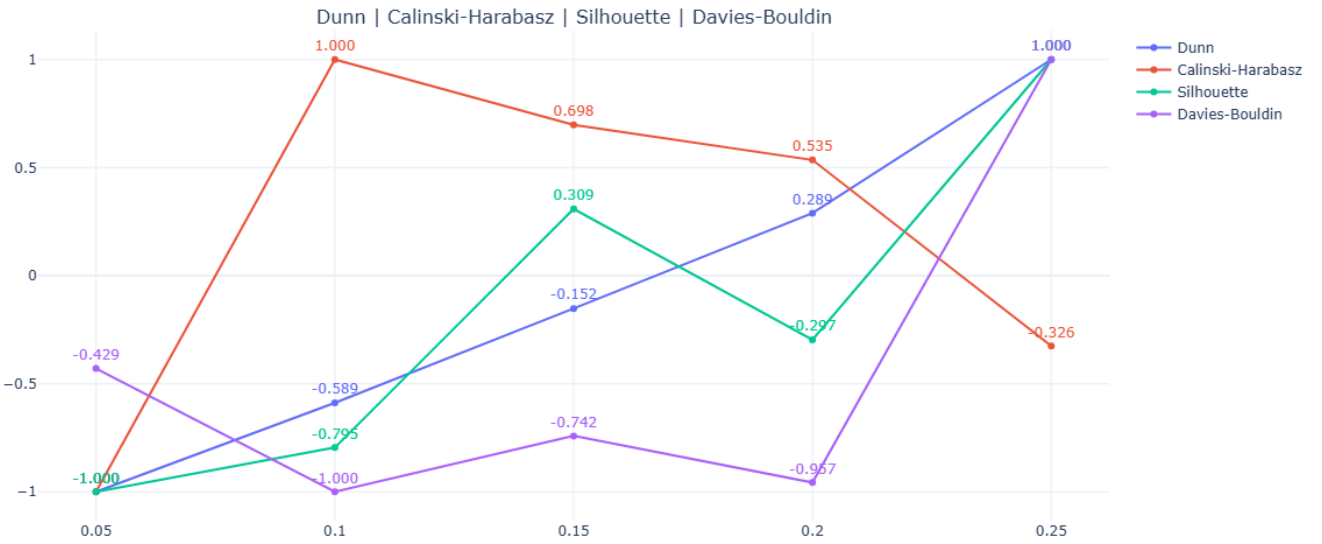
7.3. Scores - DBSCAN

DBSCAN Raw Data (eps = variable, min_sample = 3)

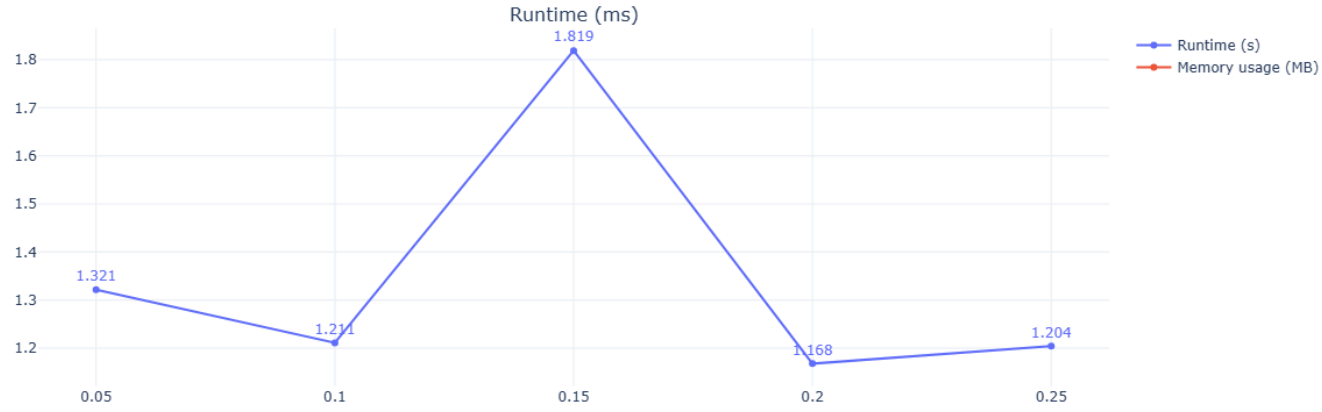
Score	Parameter: *eps*				
	0.05	**0.1**	**0.15**	**0.2**	**0.25**
Runtime (s)	1.3286	1.8709	1.2156	1.2111	1.2399
Memory usage (MB)	2027.491	2731.929	1848.804	1848.808	1881.886
Silhouette	-0.40499	-0.33509	0.04211	-0.16475	0.278249
Davies-Bouldin	2.881898	1.462747	2.105335	1.569392	6.436707
Calinski-Harabasz	0.002732	0.026283	0.022727	0.020811	0.010674
Dunn	0.00746	0.014092	0.02113	0.028241	0.039694
Cluster Size Standard Deviation	1.693204	97.10167	343.6863	382.2086	490
Smallest Cluster	3	3	4	3	6
Largest Cluster	12	604	928	961	986
Separation Ratio	21.63466	6.765378	0.729439	1.235288	0.118785
LOF Mean Score	-1.01004	-1.08785	-1.21409	-1.39923	-1.43317
Isolation Forest Mean Score	-0.13862	-0.06639	-0.01138	-0.00504	-0.00307
Silhouette Outlier Fraction	0.701	0.771	0.363	0.65	0.188
Noise Fraction	0.688	0.129	0.043	0.017	0.008
Core Border Ratio	0.798077	0.926521	0.980146	0.987792	0.995968
Average Core Distance	0.025259	0.118917	0.355	0.437757	0.891609

Graphs (eps = variable, min_sample = 3)

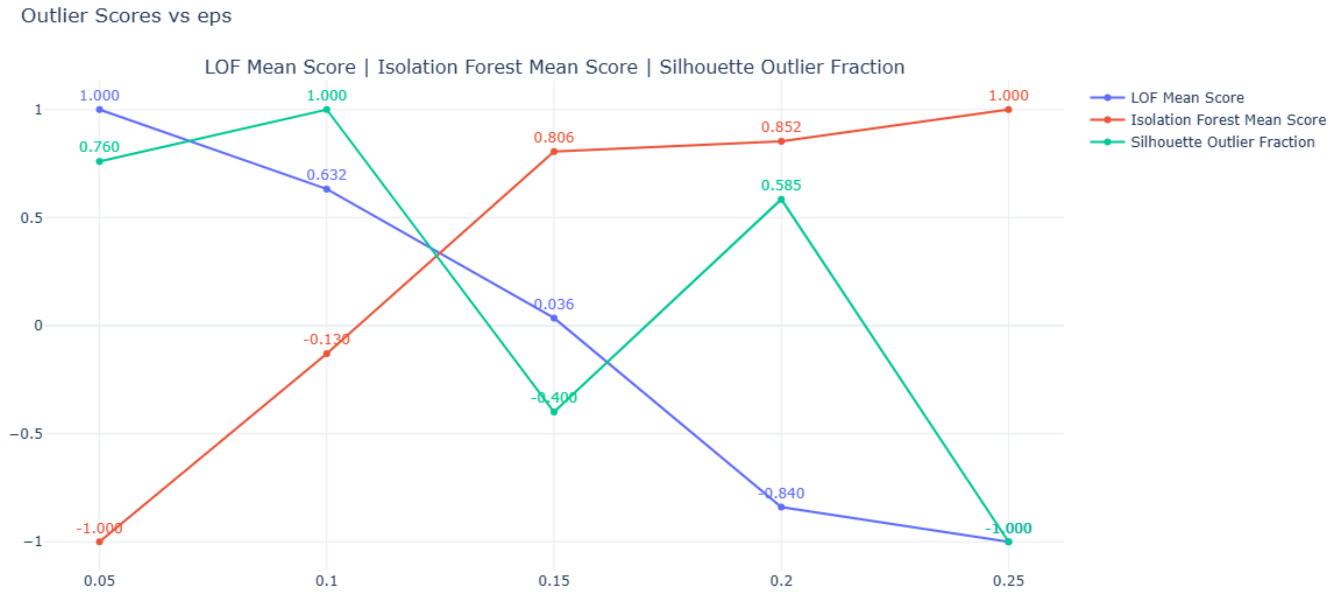
Cluster Quality Scores vs eps

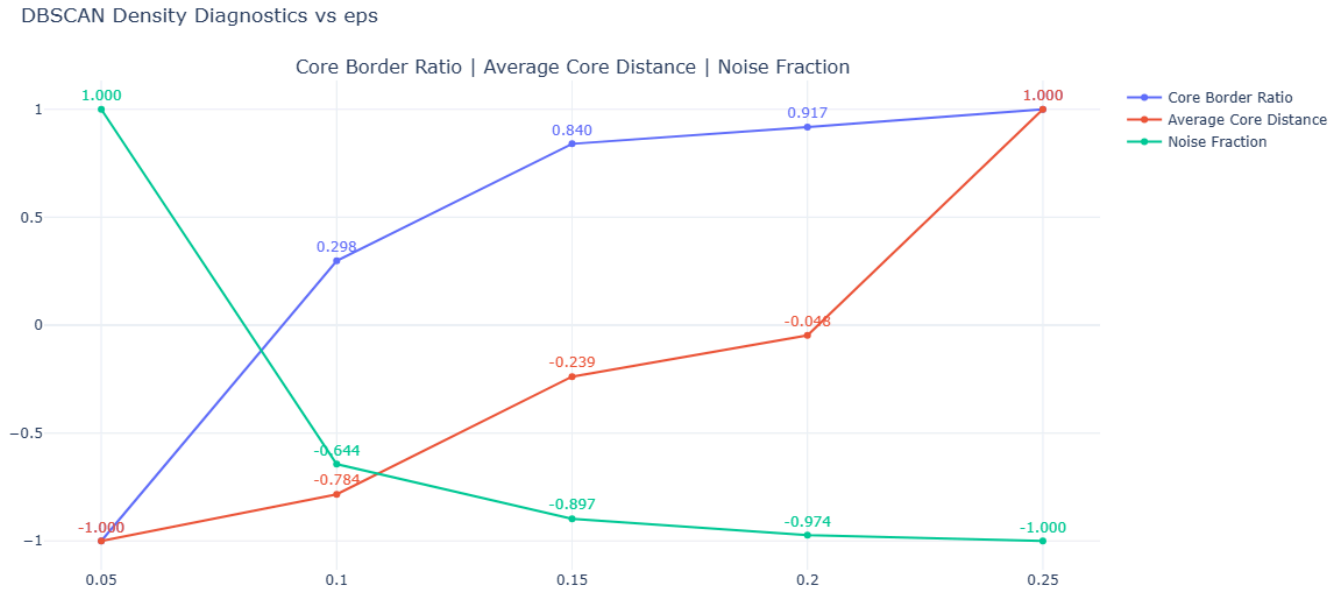


Performance vs eps



Cluster Sizes vs eps





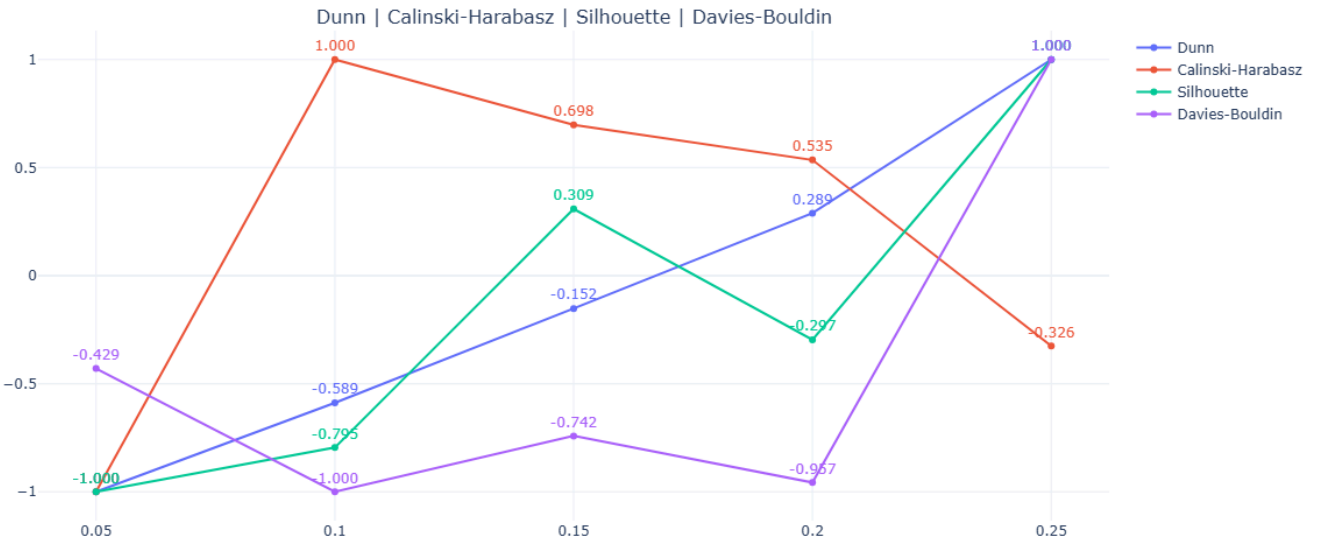
DBSCAN Raw Data (eps =0.1, min_sample = variable)

Score	**Parameter: *min_sample***				
	1	**2**	**3**	**4**	**5**
Runtime (s)	1.5193	1.9743	1.8709	1.2316	1.2284
Memory usage (MB)	2323.903	2913.137	2731.929	1865.756	1865.768
Silhouette	-0.30094	-0.29107	-0.33509	-0.22994	-0.26099
Davies-Bouldin	0.60164	1.414297	1.462747	1.626388	1.796127

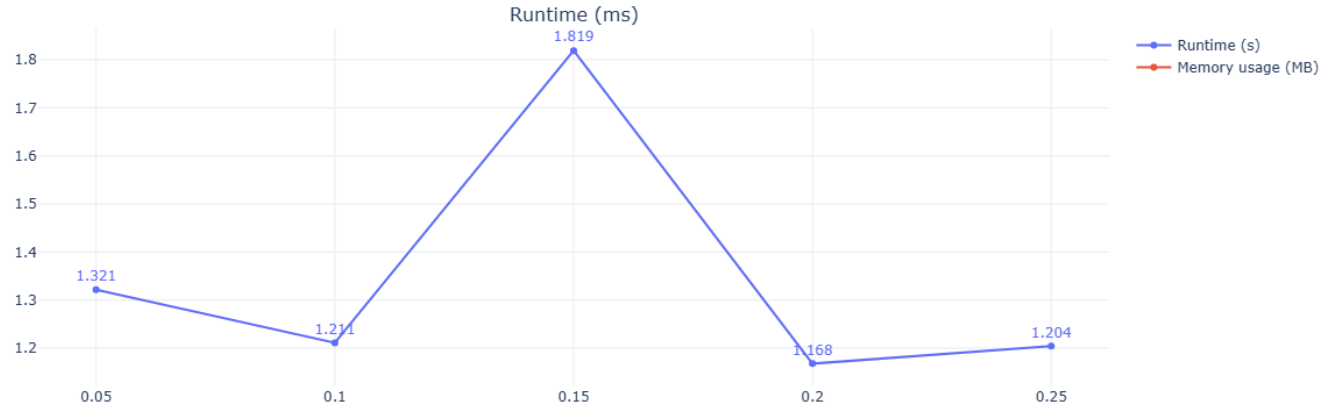
Calinski-Harabasz	0.019454	0.02253	0.026283	0.036176	0.025331
Dunn	0.02778	0.015266	0.014092	0.000829	0.000829
Cluster Size Standard Deviation	52.17403	71.40994	97.10167	85.70228	81.65272
Smallest Cluster	1	2	3	4	4
Largest Cluster	604	604	604	471	449
Separation Ratio	12.07861	11.71742	6.765378	4.283398	4.6149
LOF Mean Score	-1.13464	-1.13587	-1.08785	-1.04673	-1.04304
Isolation Forest Mean Score	-0.04262	-0.06207	-0.06639	-0.09138	-0.10045
Silhouette Outlier Fraction	0.699	0.735	0.771	0.655	0.695
Noise Fraction	0	0.063	0.129	0.205	0.255
Core Border Ratio	1	1	0.926521	0.861635	0.78255
Average Core Distance	0.076734	0.076734	0.118917	0.149296	0.136862

DBSCAN Graphs (eps =0.1, min_sample = variable)

Cluster Quality Scores vs eps

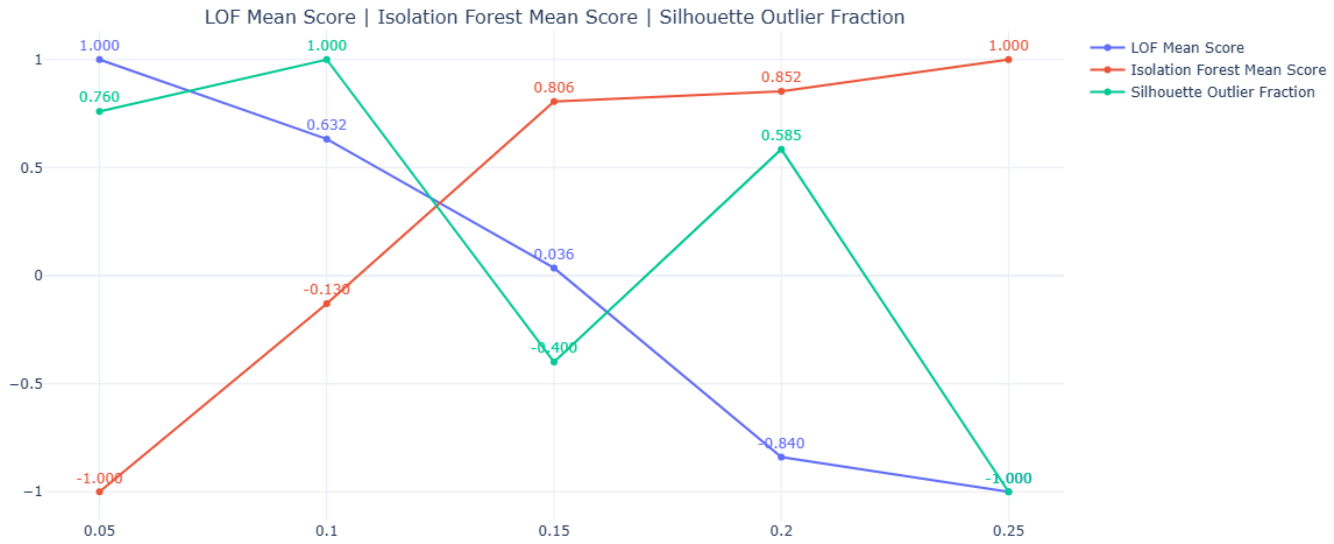


Performance vs eps

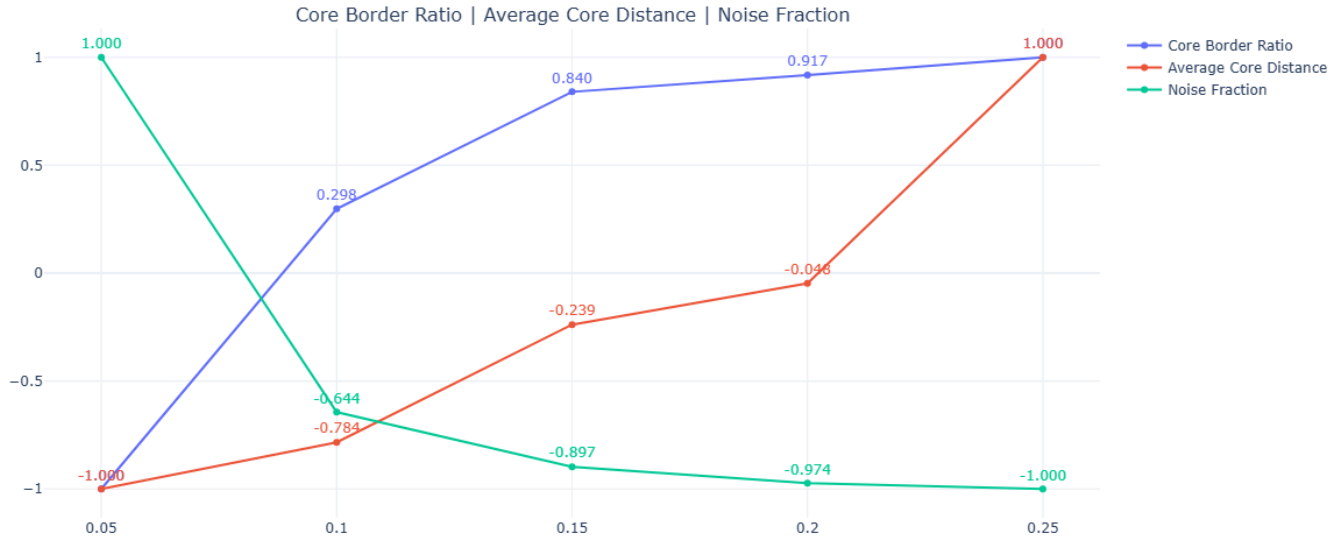


Cluster Sizes vs eps

Outlier Scores vs eps



DBSCAN Density Diagnostics vs eps



7.4. Scores - K-Means

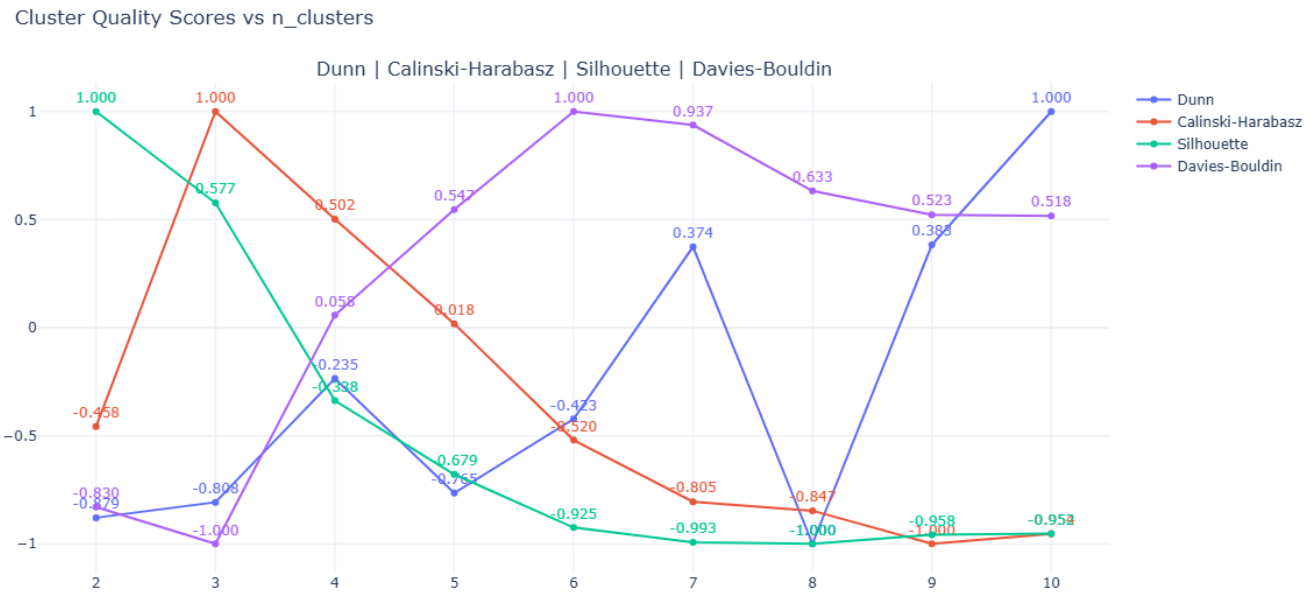
K-Means Raw Data (n_clusters = variable, n_init = 60)

Score	Parameter: *n_clusters*				
	2	**3**	**4**	**5**	**6**
Runtime (s)	0.7972	0.7968	1.2752	1.0322	0.9611
Memory usage (MB)	1163.349	1147.028	1786.512	1507.932	1409.589
Silhouette	0.465268	0.439194	0.382756	0.361736	0.346566
Davies-Bouldin	0.752614	0.738607	0.825894	0.866225	0.903626
Calinski-Harabasz	1.377763	1.551084	1.491904	1.434355	1.370364
Dunn	0.006058	0.006563	0.010607	0.006864	0.009282
Cluster Size Standard Deviation	6	98.90175	77.4177	80.60521	45.06908
Smallest Cluster	494	257	159	103	100
Largest Cluster	506	473	335	318	216
Separation Ratio	0.006073	0.19619	0.360613	0.483422	0.451493
LOF Mean Score	-1.0496	-1.05431	-1.05403	-1.0585	-1.05608
Isolation Forest Mean Score	-0.12322	-0.10548	-0.10425	-0.10073	-0.09952
Silhouette Outlier Fraction	0	0.004	0	0.005	0.017
Inertia	840.151	486.4402	364.0563	295.5878	253.3833
Iterations Until Convergence	15	5	14	11	14

Score	Parameter: *n_clusters*			
	7	**8**	**9**	**10**

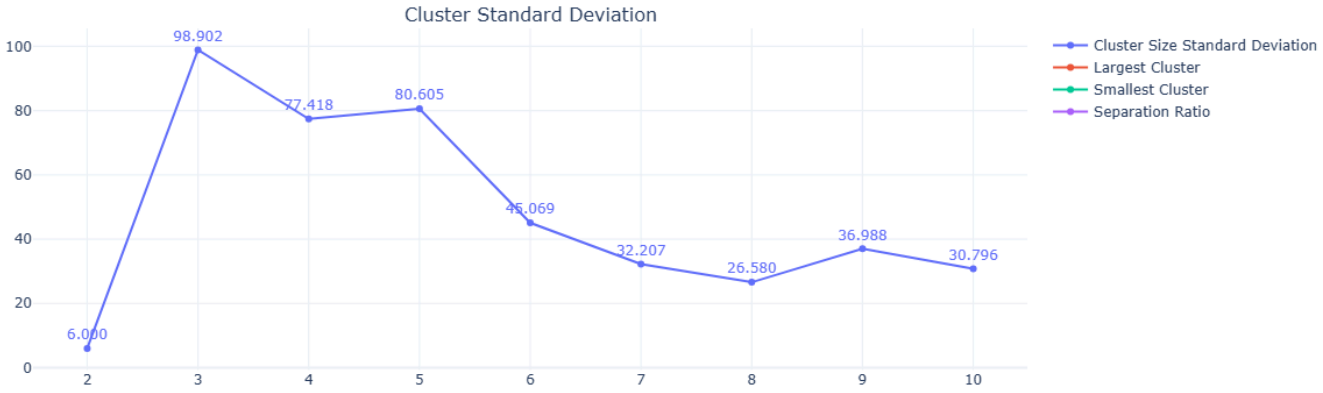
Runtime (s)	0.9102	0.9324	0.9361	0.962
Memory usage (MB)	1344.027	1376.808	1376.808	1409.589
Silhouette	0.342344	0.341928	0.344523	0.344868
Davies-Bouldin	0.898467	0.873328	0.864254	0.863818
Calinski-Harabasz	1.33645	1.331486	1.313298	1.318766
Dunn	0.01491	0.005205	0.014971	0.019328
Cluster Size Standard Deviation	32.2066	26.58007	36.98782	30.7961
Smallest Cluster	88	77	44	39
Largest Cluster	178	158	162	148
Separation Ratio	0.511652	0.604105	0.740107	0.807506
LOF Mean Score	-1.05581	-1.05632	-1.06023	-1.0599
Isolation Forest Mean Score	-0.10826	-0.10332	-0.09852	-0.1038
Silhouette Outlier Fraction	0.013	0.013	0.009	0.012
Inertia	220.3835	192.3896	172.39	153.979
Iterations Until Convergence	13	24	20	19

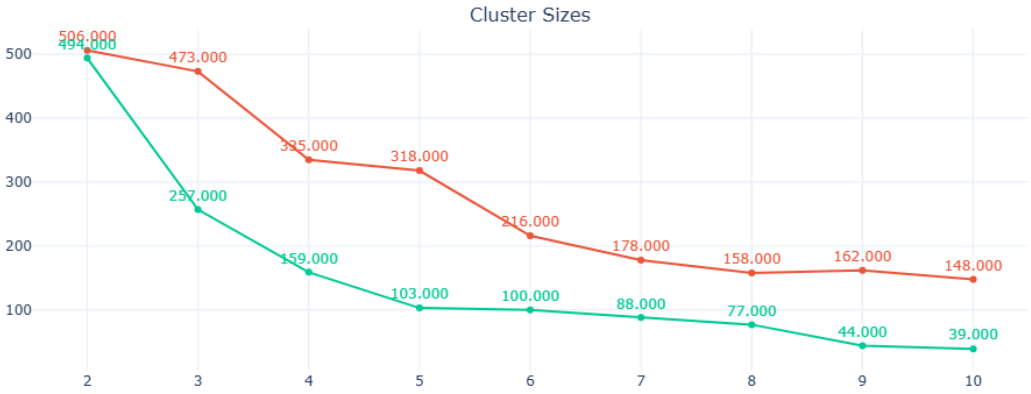
K-Means Graphs (n_clusters = variable, n_init = 60)

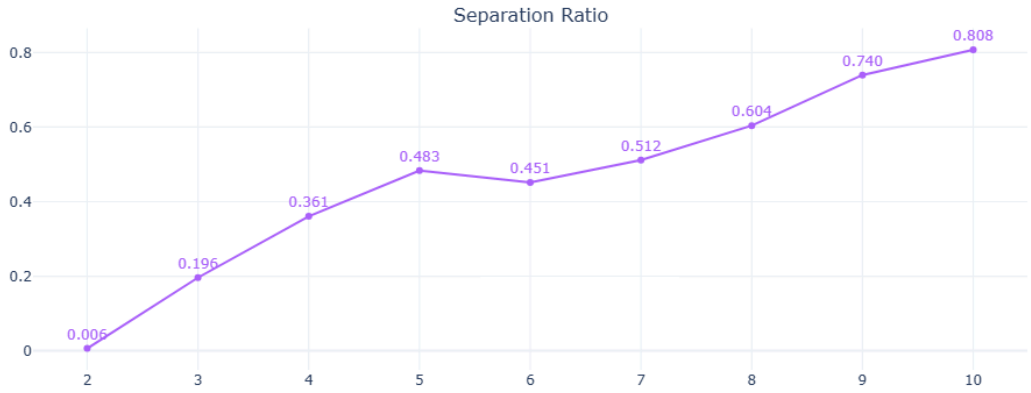

Cluster Quality Scores vs n_clusters
Dunn | Calinski-Harabasz | Silhouette | Davies-Bouldin
Dunn
Calinski-Harabasz
Silhouette
Davies-Bouldin
1
0.5
0
−0.5
−1
2
3
4
5
6
7
8
9
10
1.000
1.000
0.577
0.502
0.547
1.000
0.937
0.633
0.523
0.518
0.374
0.383
0.058
0.018
-0.235
-0.458
-0.423
-0.520
-0.805
-0.847
-0.830
-0.879
-0.808
-1.000
0.679
-0.925
-0.993
-1.000
-0.958
-1.000

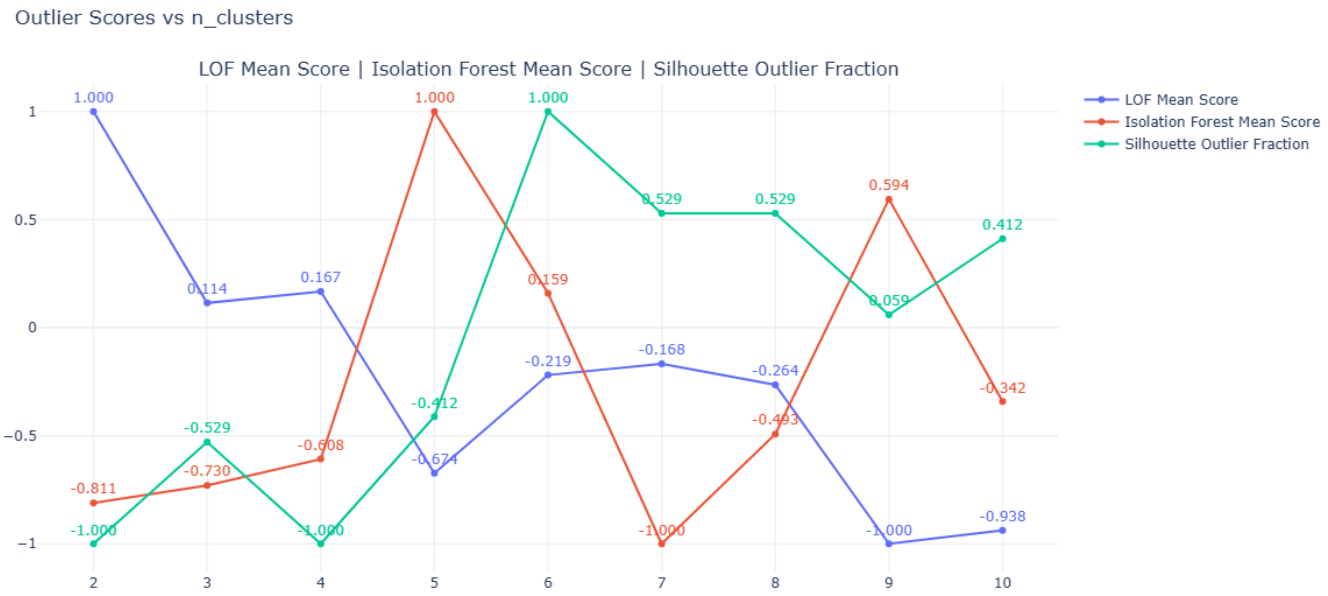
Performance vs n_clusters

Cluster Sizes vs n_clusters









K-Means Raw Data (n_clusters = 4, n_init = variable)

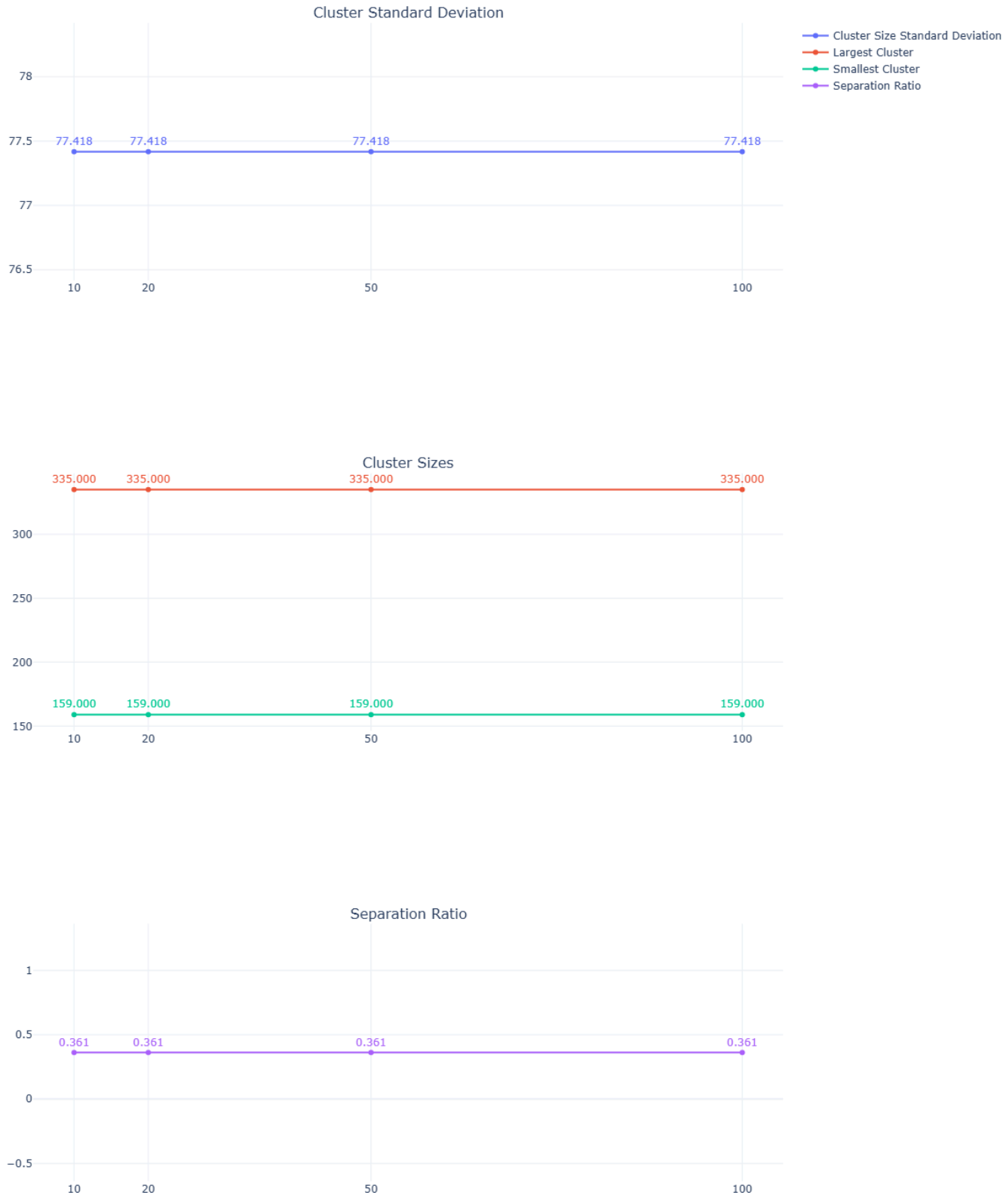
Score	**Parameter: *n_init***			
	10	**20**	**50**	**100**
Runtime (s)	0.7139	1.0094	1.2422	0.9253
Memory usage (MB)	1048.997	1458.762	1770.182	1360.418
Silhouette	0.382756	0.382756	0.382756	0.382756
Davies-Bouldin	0.825894	0.825894	0.825894	0.825894
Calinski-Harabasz	1.491904	1.491904	1.491904	1.491904
Dunn	0.010607	0.010607	0.010607	0.010607
Cluster Size Standard Deviation	77.4177	77.4177	77.4177	77.4177
Smallest Cluster	159	159	159	159
Largest Cluster	335	335	335	335
Separation Ratio	0.360613	0.360613	0.360613	0.360613
LOF Mean Score	-1.05403	-1.05403	-1.05403	-1.05403
Isolation Forest Mean Score	-0.10349	-0.10128	-0.11474	-0.10456

Silhouette Outlier Fraction	0	0	0	0
Inertia	364.0563	364.0563	364.0563	364.0563
Iterations Until Convergence	14	14	14	14

K-Means Graphs (n_clusters = 4, n_init = variable)

Cluster Quality Scores vs n_init

Cluster Sizes vs n_init



Performance vs n_init

Outlier Scores vs n_init