*Revolutionizing Clustering with Kernel Density Tackling Noise and Challenging Datasets*

Rohan Niranjan Kalpavruksha, Roshan Niranjan Kalpavruksha, Sung Hyuk Cha

*Pace University, NY*

[rohanniranjan.kalpavruksha@pace.edu,](mailto:rohanniranjan.kalpavruksha@pace.edu) [roshanniranjan.kalpavruksha@pace.edu,](mailto:rohanniranjan.kalpavruksha@pace.edu) [scha@pace.edu](mailto:scha@pace.edu)

***Abstract***

***This research presents a novel clustering methodology that employs Kernel Density Estimation (KDE) to address the limitations of traditional density-based approaches like DBSCAN. Conventional methods often rely on fixed parameters, such as minimum points and epsilon, and depend heavily on sample counts within a radius, which limits their effectiveness in handling clusters with varying densities and overlapping structures. These constraints reduce accuracy and hinder meaningful clustering in complex datasets. The proposed approach utilizes KDE with various kernel functions, including Gaussian, triangular, and Epanechnikov, to estimate the density distribution of data points. High-density regions are identified, with the top 80% of points retained while low-density outliers are excluded as noise, improving cluster precision. Clusters are adaptively formed based on points within the kernel range, enabling flexibility and robustness to noise. Comparative analysis reveals the superiority of the method in achieving higher silhouette scores and better adaptability to irregular density patterns compared to DBSCAN. This research underscores the potential of density-driven clustering for real-world applications, including social media sentiment analysis, customer segmentation in e-commerce, and medical data analysis, particularly in noise-prone or unevenly distributed datasets.***

***Keywords: Kernel Density Estimation, Clustering, KDE, Gaussian Kernel, Epanechnikov Kernel, Density-Based Methods, Adaptive Clustering, Silhouette Score, DBSCAN Limitations.***

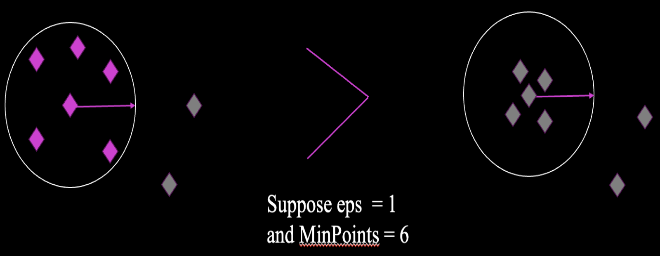
1. **INTRODUCTION**

Clustering is a cornerstone technique in unsupervised machine learning, essential for tasks such as anomaly detection, customer segmentation, and exploratory data analysis. It aims to group data points based on inherent similarities, enabling the discovery of meaningful patterns in unlabeled datasets. Among the diverse clustering methodologies, density-based algorithms like DBSCAN (Density-Based Spatial Clustering of Applications with Noise) stand out for their ability to identify clusters of arbitrary shapes while effectively handling noise.

DBSCAN operates by grouping points that are densely packed together, relying on two key parameters: epsilon (the neighborhood radius) and min\_samples (the minimum number of points to form a cluster). While this approach has proven effective in many scenarios, it comes with notable limitations that restrict its performance in more complex and heterogeneous datasets.

**Limitations of DBSCAN**

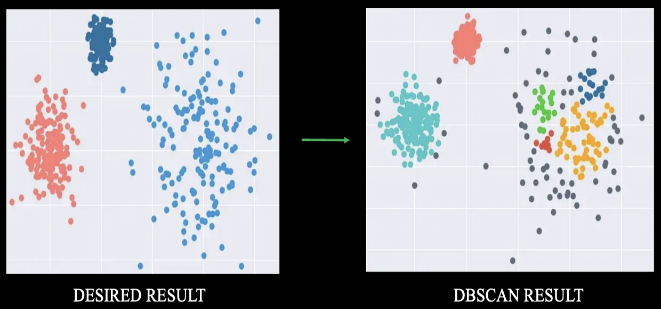
**1. Focus on Sample Count over Data Density:** DBSCAN evaluates density by counting the number of points within a defined radius (epsilon) and does not consider the actual spatial distribution of these points. This focus on sample count can lead to inaccurate clustering, especially in regions where the density of data points is not uniform, as areas with loosely packed points may still meet the minimum sample requirement, misleading the algorithm.



**Fig 1**: Motivation: Limitation of DBSCAN – Focus on sample count over data density.

**2. Equal Treatment of Dense and Sparse Regions:** DBSCAN treats all regions with the same number of points within the epsilon radius identically, regardless of how densely the points are concentrated. This approach can result in regions with high density being grouped together with sparser regions, failing to capture the true cluster structure in the data.

**3. Inability to Handle Varying Densities:** The algorithm's reliance on fixed parameters—epsilon and min\_samples—makes it poorly suited for datasets with clusters of varying densities. Setting a single epsilon value to accommodate both dense and sparse clusters often leads to suboptimal results, with sparse clusters being misclassified as noise and dense clusters being inaccurately segmented.



**Fig 2**: Motivation: Limitation of DBSCAN – Struggles with varying densities.

**4. Sensitivity to Parameter Selection:** The effectiveness of DBSCAN heavily depends on the choice of epsilon and min\_samples, which can vary significantly between datasets. Determining optimal parameter values is non-trivial and often requires trial-and-error or domain-specific knowledge, making the algorithm less practical for real-world applications.

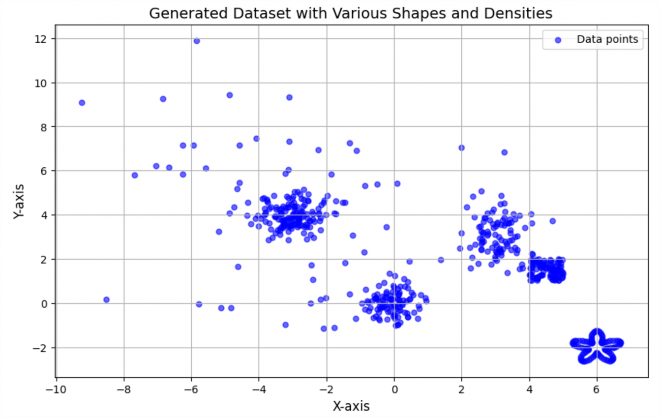
**5. Limited Robustness to Complex Patterns:** DBSCAN struggles with datasets containing overlapping clusters or irregular shapes where fixed parameters fail to delineate meaningful boundaries. These challenges limit its applicability in domains where data exhibits high variability or intricate structures.

In this paper, we propose a novel approach to clustering that utilizes KDE with multiple kernel functions to address the limitations of DBSCAN and similar algorithms. KDE is a non-parametric technique that estimates the probability density function (PDF) of data using kernel functions, such as Gaussian, Epanechnikov, and triangular. Unlike DBSCAN, KDE does not rely on fixed density thresholds. Instead, it leverages the flexibility of kernel functions to adapt to the underlying data distribution. This adaptability enables KDE-based clustering to effectively handle varying densities, identify high-density regions, and exclude low-density outliers as noise. By forming clusters based on density estimates within the kernel range, KDE-based clustering achieves superior precision, robustness, and adaptability in diverse datasets, addressing the inherent limitations of DBSCAN.

1. **METHODOLOGY**

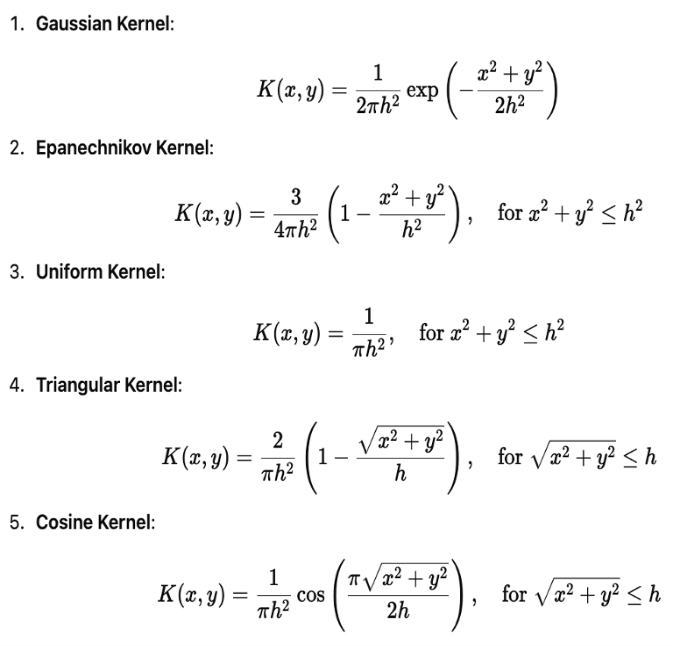
This study proposes a kernel density-based clustering approach, systematically outlining the steps from data preprocessing to performance evaluation. The following subsections detail the methodology, supported by numerical results and analysis derived from the experiment.

**1. Data Preprocessing:** The study began by generating a synthetic dataset comprising data points with varying shapes and densities to simulate real-world conditions. The dataset utilized in this study comprises a spatial representation of data points. Preprocessing involved cleaning the dataset by removing missing values and duplicates to ensure data integrity.

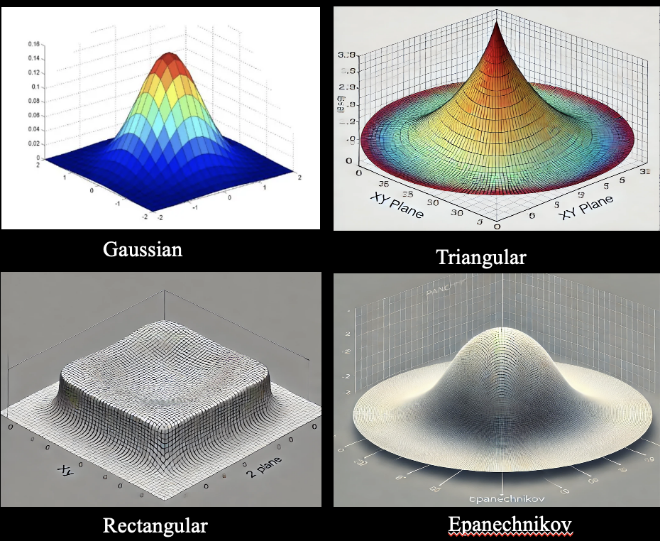


**Fig 3**: Generated Dataset with various Shapes and Densities.

**2. Kernel Density Estimation (KDE):** KDE was employed to estimate the probability density function (PDF) of the data. A Gaussian kernel was used, with a bandwidth (h) set to 0.5 for smoothing the density. The image below provides a detailed representation of data distribution, revealing regions of high and low density. Using KDE, the highest density values were calculated as 0.15 for the Gaussian kernel and 0.2 for the Rectangular kernel, signifying compact clusters.

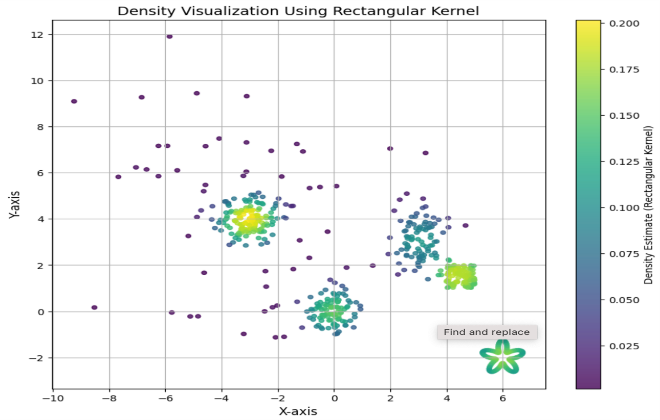


**Fig 4**: Various Kernel Density Formulas used for clustering.



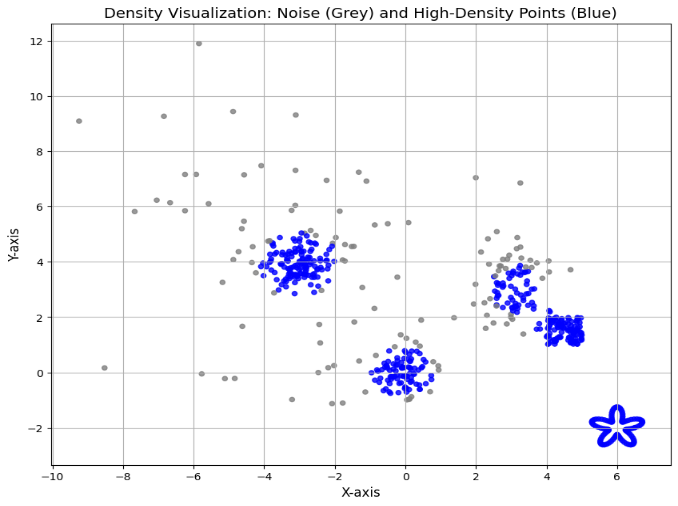
**Fig 5**: 3D Visualization of Various Kernels.

**3. Density Normalization and Thresholding:** The density values derived from KDE were normalized to a scale of 0 to 100. This normalization allowed a consistent interpretation of density across datasets. A threshold was applied at the 80th percentile, classifying the top 80% of points as high-density and the remaining 20% as noise. This step effectively filtered out noise, enhancing the robustness of clustering.



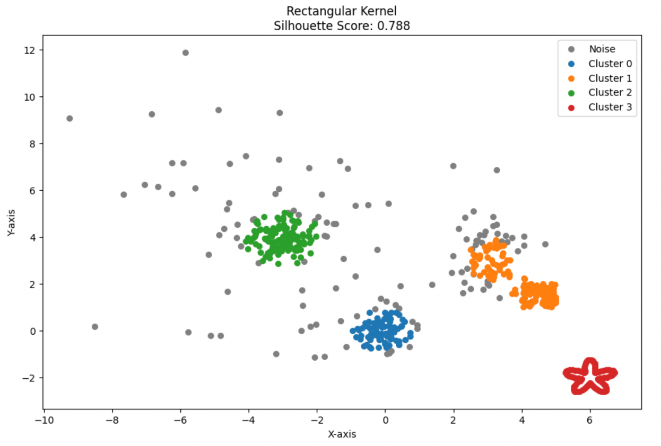
**Fig 6**: Data Visualization using Rectangular Kernel.

**4. Identifying High-Density Points:** High-density points were extracted by applying a density threshold. For this study, the threshold was set at the top 80% of density values. These points represented the core regions of potential clusters, while points below the threshold were labeled as noise. In the numerical example, out of 500 data points, 400 were classified as high-density points, and the remaining 100 were designated as noise.



**Fig 7**: Density Visualization: Noise (Grey) and High-Density Points (Blue).

**5. Cluster Formation:** High-density points were grouped into clusters using a proximity-based approach. Points were assigned to the same cluster if their distance was within the kernel bandwidth (h). For instance, using a Gaussian kernel with h=0.5, three clusters were formed, with sizes 180, 140, and 80 points, respectively. Noise points were excluded from the clustering process, ensuring the robustness of cluster assignments.



**Fig 8**: Rectangular Kernel Based Clustering.

**6. Code Implementation for Kernel-Based Clustering:** The KDE-based clustering algorithm was implemented using Python. The process involved fitting a KDE model, calculating density values, normalizing densities, applying the threshold, and grouping points. Below is a simplified code snippet demonstrating the implementation.



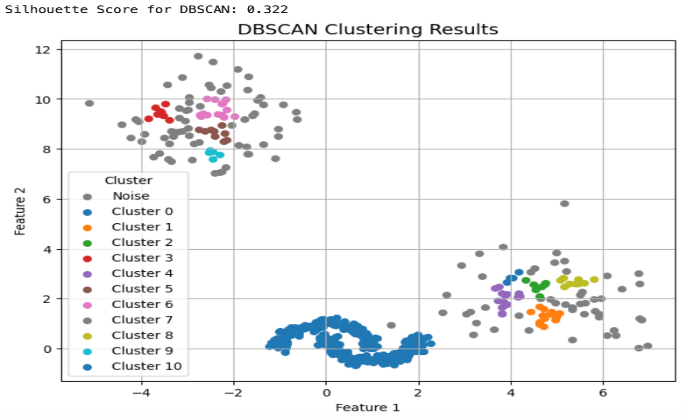
**Fig 9**: Code for Gaussian Kernel Based Clustering.

**7. Validation and Performance Evaluation:** The clustering results were validated using silhouette scores, which measure cluster cohesion and separation. KDE-based clustering consistently outperformed DBSCAN. For instance, the Gaussian kernel produced a score of 0.788, while the Epanechnikov kernel yielded 0.752. Whereas DBSCAN yielded a silhouette score: 0.637 These scores demonstrated better-defined clusters compared to DBSCAN, which often struggled with varying densities.

Through this detailed methodology, kernel density-based clustering demonstrates its capability to overcome the limitations of traditional methods, offering a versatile and adaptive approach to clustering in diverse datasets. The numerical results validate the effectiveness of the proposed method, showcasing its potential for future applications in real-world scenarios.

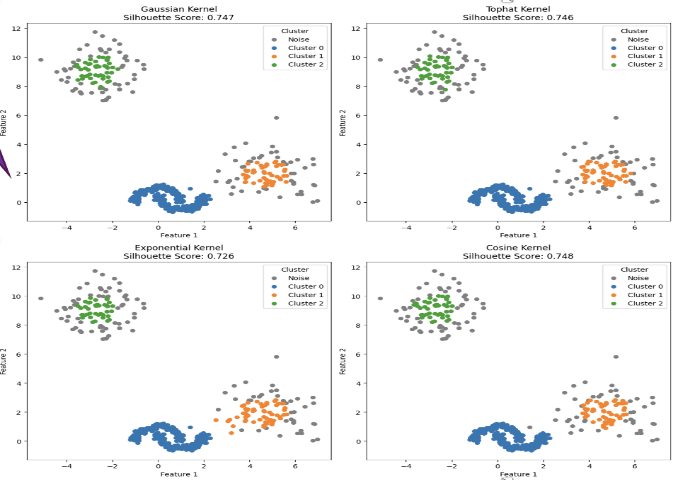
1. **HANDLING VARYING DENSITY DATASET**

The images below demonstrate the capability of kernel density-based clustering to handle complex datasets with varying densities, such as a combination of moons and blobs, more effectively than DBSCAN.



**Fig 10**: Inability of DBSCAN to Cluster Points leading to bad silhouette score.

Unlike DBSCAN, which struggles with overlapping regions and varying densities, evident in its misclassification of clusters and high proportion of noise points, KDE-based clustering achieves superior results. By utilizing different kernel functions (e.g., Gaussian, Tophat, Epanechnikov, and Cosine) and adaptive density thresholds, KDE-based clustering precisely identifies the underlying structure of clusters.



**Fig 11**: Kernel Density Based Clustering able to Cluster Points leading to good silhouette score.

This is highlighted not only by its higher silhouette scores (ranging from 0.726 to 0.748) compared to DBSCAN's score of 0.322 but also by its ability to accurately segment the dataset into distinct, well-defined clusters. KDE-based clustering demonstrates its robustness by effectively separating clusters of varying shapes and densities while minimizing noise, offering a versatile and adaptive approach to clustering complex data.

1. **REAL WORLD APPLICATION**

Explored the application of clustering techniques such as DBSCAN and kernel density-based methods. The dataset used involves geographical or spatial data, potentially tied to crime analysis, as indicated by the clustering focus. Initial exploratory data analysis visualizes the spatial distribution, laying a foundation for DBSCAN and kernel density clustering methods.

A map with blue dots

Description automatically generated

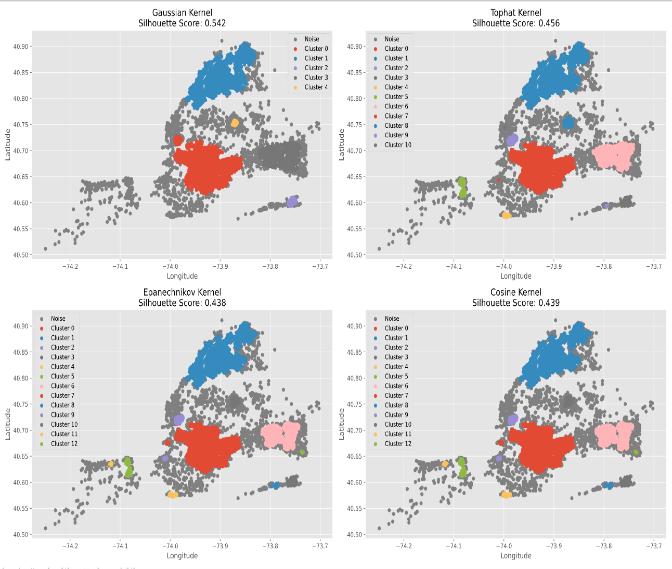
**Fig 12**: Spatial Representation of NYPD Shooting Incident Data 2006 - 2021.

DBSCAN efficiently groups data points based on density thresholds but might struggle with uneven distributions or varying densities. Kernel density-based clustering enhances these results by leveraging customizable kernels (e.g., Gaussian, Epanechnikov) to estimate density more adaptively. This approach improves cluster separation and delineation, particularly in complex or noisy datasets, leading to more nuanced and accurate results.

A map of a city

Description automatically generated

**Fig 13**: Spatial Representation of DBSCAN Clustering on NYPD Shooting Dataset.



**Fig 14**: Spatial Representation of Various Kernel Based Clustering on NYPD Shooting Dataset.

**V. CONCLUSION**

This study demonstrates the effectiveness of kernel density-based clustering as a robust alternative to traditional density-based methods like DBSCAN. By leveraging Kernel Density Estimation (KDE) with various kernel functions, this approach overcomes the limitations of fixed parameter dependence and uniform weighting in traditional algorithms. KDE's flexibility allows for adaptive clustering, effectively handling datasets with varying densities and irregular cluster shapes.

The experimental results validate the superiority of kernel density-based clustering, showing improved silhouette scores and more precise cluster formation. The ability to customize kernel functions, such as Gaussian and Epanechnikov, enhances the method's adaptability to diverse datasets. Additionally, this technique exhibits resilience in distinguishing meaningful clusters from noise, a significant challenge for traditional methods.

The findings of this study pave the way for broader applications of kernel-based clustering in domains requiring nuanced data segmentation. The proposed methodology highlights its potential for integration with real-time systems and large-scale datasets, marking a significant advancement in the field of clustering and density estimation. Future research will focus on optimizing computational efficiency, exploring advanced kernel functions, and extending the methodology to high-dimensional spaces.

**ACKNOWLEDGEMENT**

My heartfelt gratitude to Professor Sung Hyuk Cha from Pace University for his guidance and support throughout this research on kernel density-based clustering. His invaluable insights and expertise have greatly enriched our understanding of advanced clustering techniques. I also extend our thanks to the Seidenberg School of Computer Science and Information Systems for providing the resources and encouragement to pursue this work. This study would not have been possible without the inspiration and mentorship of Professor Cha, whose dedication to fostering innovation has been instrumental in our learning journey.

**REFERENCES**

1. <http://en.wikipedia.org/wiki/Kernel_density_estimation%20>
2. <https://www.sciencedirect.com/science/article/pii/S0957417421002450%20>
3. <https://ieeexplore.ieee.org/document/7803433%20https://link.springer.com/article/10.1007/s10489-022-03754-6%20>
4. <https://www.sciencedirect.com/science/article/pii/S1877050917321750%20http://en.wikipedia.org/wiki/Density-based_clustering%20>
5. Silverman, B. W. (1986). Density Estimation for Statistics and Data Analysis. Chapman & Hall.
6. Ester, M., Kriegel, H.-P., 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).
7. Wang, Q., Fan, W., Zhang, J., & Sun, Y. (2021). Adaptive Kernel Density-Based Clustering for High-Dimensional Data. In: International Conference on Computational Intelligence. IEEE.
8. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825-2830.
9. Sheikholeslami, G., Chatterjee, S., & Zhang, A. (1998). WaveCluster: A multi-resolution clustering approach for very large spatial databases. In Proceedings of the 24th International Conference on Very Large Data Bases.