A Novel Approach to Outlier-Aware K-means Clustering

Quarter 2 Project

Dhruv Chandna & Soham Jain

Dr. Yilmaz

Period 6

02/05/2025

Background

- Created our own dataset with 450 instances
- **Input:** CSV containing values for 2 quantitative continuous features
- **Output:** Target classification (0 or 1) using a novel, outlier-aware K-means clustering algorithm

q2_project_data

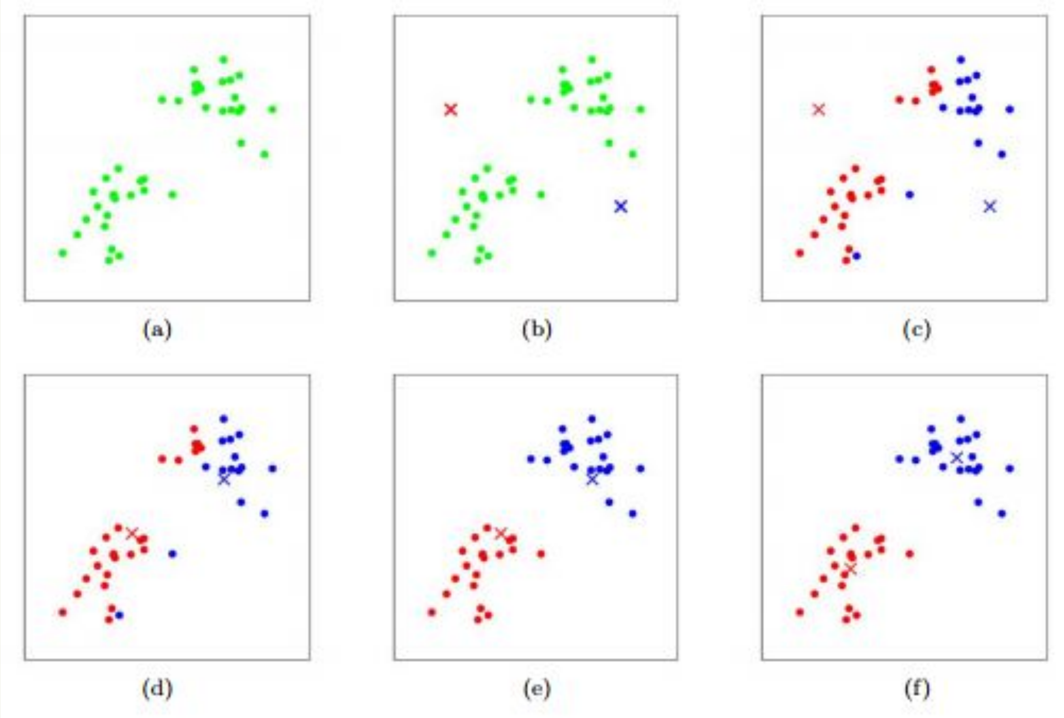
Feature1	Feature2	Target
0.05077446112416210	-0.06229318708512020	1
-0.06750982072188660	0.0934274898521248	1
-0.07840845307785590	0.15735138387553300	1
0.004247756419569180	0.004463420924361780	1
-0.14844072303878700	-0.01886803801494720	1
-0.005645205363727250	0.1654952823024780	1
0.46183212126828600	0.1679362367151870	1
-0.9063820274421770	0.09425182638964200	0
…	…	…

Input

Background: K-means

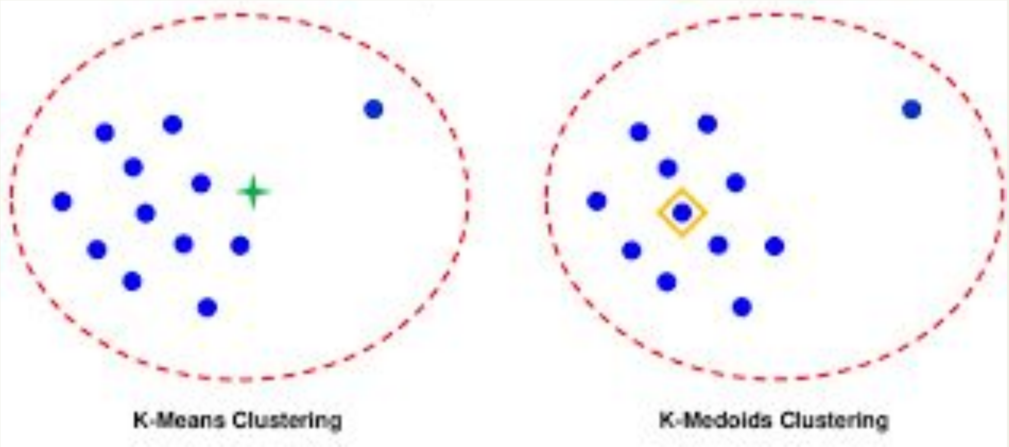
- Calculates Euclidean distances between data points
- Uses mean value as the new centroid
- Problem: sensitivity to outliers

k = 2:



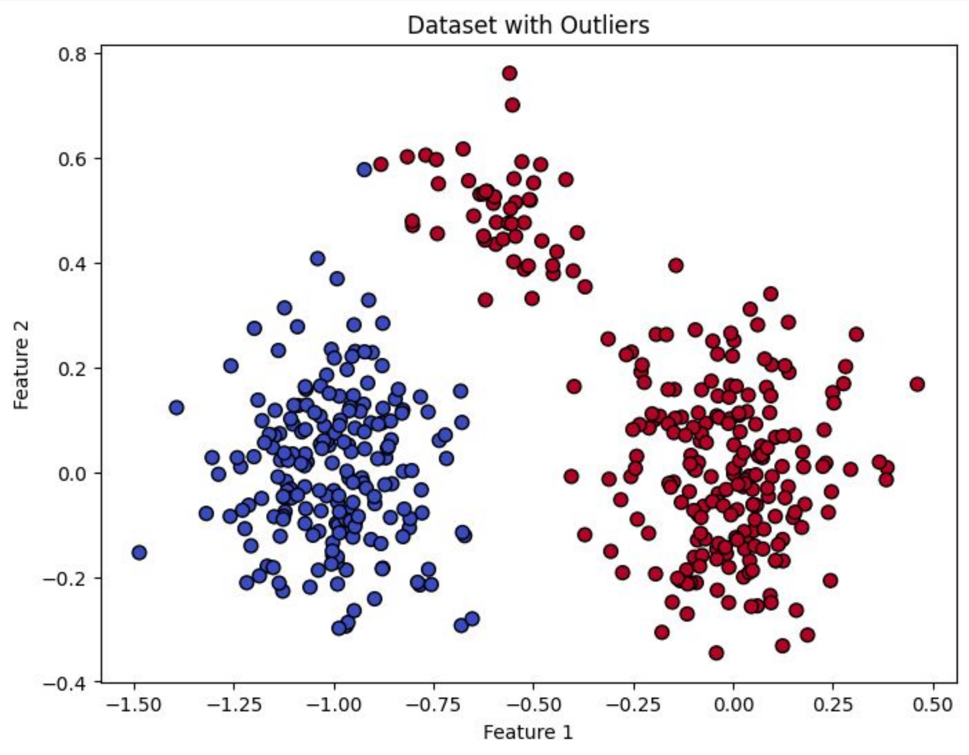
Background: K-medians

- Instead of mean, the measure of center is median
- Resistant to outliers
- Problem: Poor representation of varying/split data clusters



Dataset

- 450 total instances
- Two large clusters of 200 instances for each class
- One cluster with 50 instances in between



Related Work

- Song (2024)
  - 89% accuracy with K-means on iris dataset
  - Limitation: Sensitivity to outliers and variance
- Zhang et al. (2020); Olukanmi and Twala (2017)
  - Applied K-means after removing all outliers
  - Limitation: loss of information

Methods

- We propose a clustering algorithm that accounts for outliers without needing to completely remove them
- Weighted mean for setting new centroid locations
  - Hybrid integration of K-means with K-medians

Methods: Our Algorithm

$$(x_f, y_f, \ldots) = (1 - \lambda) \cdot (x_{mean}, y_{mean}, \ldots) + \lambda \cdot (x_{median}, y_{median}, \ldots)$$

New centroid

Centroid calculated with K-means

Weight

Centroid calculated with K-medians

Methods: Weight Calculation

$$\underbrace{\lambda}_{\text{Weight}} = \sqrt[3]{\frac{\#\text{ of outliers}}{\#\text{ of instances}}}$$

- Ratio between outliers and total number of instances
- Cube root to amplify the weight
- Weight ranges between 0 and 1

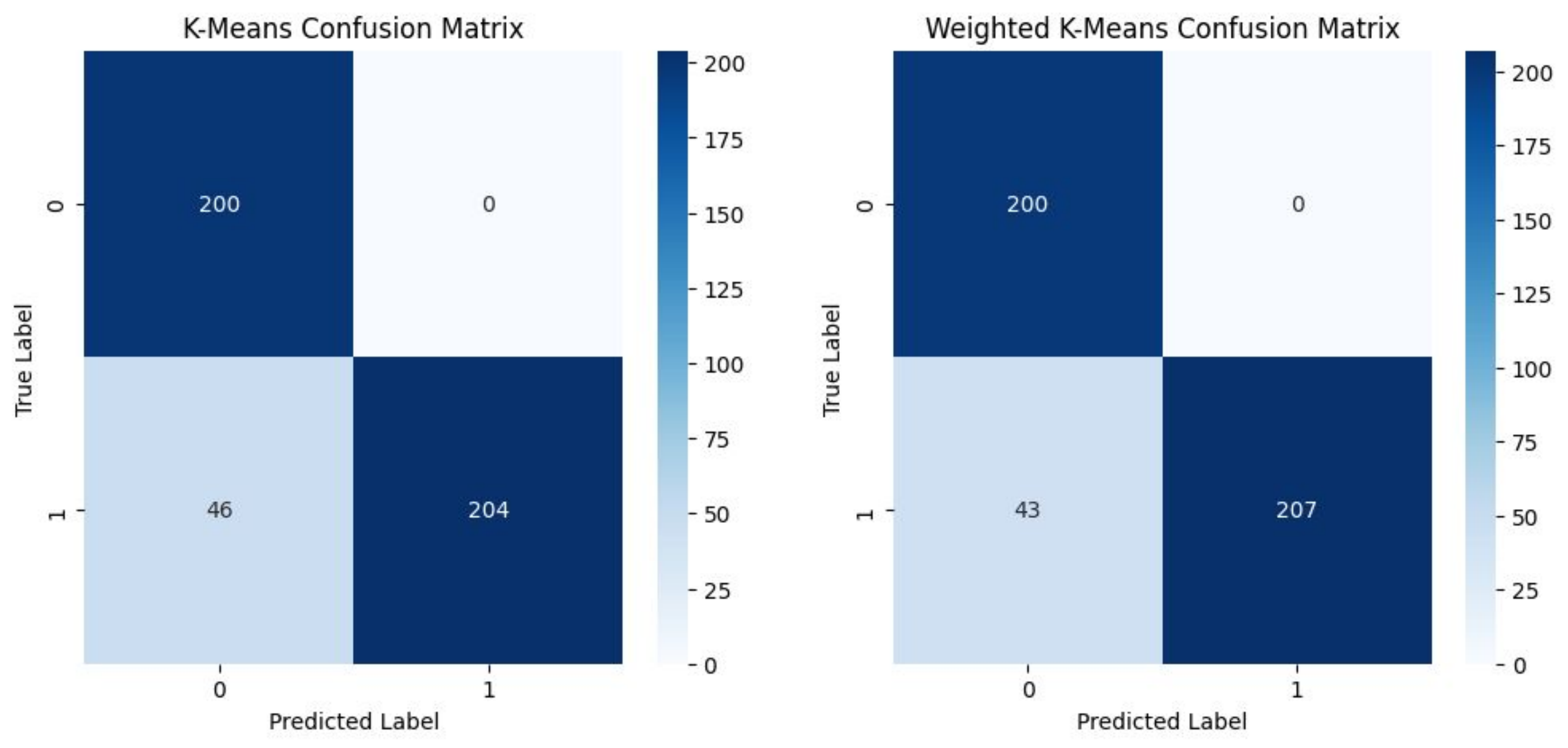
Results: Performance Metrics

- Accuracy, Precision, Recall, F1-score
- Created our own performance metric called “homogeneity” to measure how much of a cluster belongs to the same class
  - Homogeneity = (max count of a label in C) / (total points in C), where C is the cluster
- Confusion matrix

Results: Performance Metrics

- K-means:
  - Precision: 0.9065
  - Recall: 0.9080
  - F1-Score: 0.8978
  - Accuracy: 0.8978
  - Cluster 1 Homogeneity: 1.0
  - Cluster 2 Homogeneity: 0.813
- Outlier-Aware K-means:
  - Precision: 0.9115
  - Recall: 0.9140
  - F1-Score: 0.9044
  - Accuracy: 0.9044
  - Cluster 1 Homogeneity: 1.0
  - Cluster 2 Homogeneity: 0.823

Results: Confusion Matrices



Conclusion

- Our Outlier-Aware K-means algorithm improves performance compared to standard K-means in datasets with outliers.
- The weighted integration of K-means and K-medians allows for better cluster homogeneity without completely discarding outliers.
- Performance metrics, including accuracy and F1-score, show a slight improvement over traditional K-means.

Future Work

- Test our model on real-world datasets with higher dimensionality (e.g. iris, diabetes, etc.)
- Explore other adaptive weighting strategies to dynamically adjust outlier influence

References

https://stanford.edu/~cpiech/cs221/handouts/kmeans.html

https://www.researchgate.net/publication/380208588_Research_on_clustering_algorithms_based_on_the_Iris_dataset

https://onlinelibrary.wiley.com/doi/10.1155/2020/3650926

https://ieeexplore.ieee.org/document/8261116