Optimal Clustering: Developing and Evaluating Mixed-Integer Optimization Models to Perform K-means and K-medoids Clustering

Zack Horton and Tanner Street

15.095 - Machine Learning under a Modern Optimization Lens
December 14, 2023

PROBLEM AND REASONING

Clustering is an unsupervised learning method that groups similar observations in distinct subgroups. K-means and k-medoids are frequently used clustering methods. Both algorithms begin by randomly selecting points to be centers of each cluster. The remaining observations are then assigned to clusters based on their proximity to that cluster's center. K-means uses Euclidean distance to determine an observation's proximity to the center, whereas k-medoids may use Manhattan or Euclidean distance. The algorithms diverge in the remaining iterations: k-means calculates cluster centers as the average of points in that cluster (centroids) whereas k-medoids require cluster centers to be actual observations (medoids).

Both algorithms are heuristic, so there is no guarantee that the global optimum is found. Additionally, results vary significantly based on the random initialization of cluster centers. Mixed-integer optimization (MIO) models for k-means and k-medoids using Manhattan or Euclidean distance were developed to address these shortcomings. The optimization models were evaluated against the heuristic algorithms in terms of solution quality, scalability, and convergence time.

DATA

The optimization models were evaluated using two datasets from the UC Irvine Machine Learning Repository, "Abalone" and "Similarity Prediction". The abalone dataset contains information regarding the physical attributes of abalones, a group of marine gastropod molluscs. It consists of 4,177 observations and 9 features- most of which are continuous. The similarity prediction dataset contains information about the similarity between different molecules, such as descriptors and SMILES representations. It consists of 100 observations and 17 features- most of which are categorical. The datasets differ significantly in terms of size and types of features.

Data must be normalized or scaled before clustering. Indeed, the scale of features is important since large values introduce an intrinsic 'weighting' to the clusters, but it is more accurate for all features to be considered equally important. Both datasets were preprocessed through two different methods (Equations 1 and 2).

$$x' = \frac{x - \bar{x}}{\sigma}$$

Equation 1: Formula used for normalizing data (z-score)

$$x' = \frac{x - min(x)}{max(x) - min(x)}$$

Equation 2: Formula used for scaling data (min-max)

METHODS

Mixed-Integer Formulations

The full formulation for k-means clustering is provided in Figure B1 in Appendix B. The underlying idea behind the formulation is to allow the binary matrix, *z*, to assign observations to clusters, while the continuous matrix, *c*, takes on the appropriate centroid values. Since *c* is unconstrained, it should take values such that the distance measured within each cluster is minimized. The MIO formulation allows for optimization over both Manhattan and Euclidean distance, unlike the common Julia package, “Clustering.jl”.

The full formulation for k-medoids clustering is provided in Figure C1 in Appendix C. Similar to k-means, the underlying idea is to allow the binary matrix, *z*, to assign observations to clusters, while the continuous matrix, *c*, takes on the appropriate medoid values. Contrary to k-means, the continuous matrix *c* is exactly determined through big-M constraints and the *m* binary matrix. The use of the *m* binary matrix enforces that exactly *k* points are used as medoids in the solution, selecting points such that the distance measured within each cluster is minimized. Again, this MIO formulation allows for optimization over both Manhattan and Euclidean distances.

An interesting contrast between the “Cluster.jl” implementation of k-medoids and k-means is that users may provide the distance matrix for k-medoids, allowing Manhattan distance to be used instead of just Euclidean distance. However, to stay consistent with the k-means function from “Cluster.jl”, k-medoids heuristic models were implemented only using Euclidean distance.

Preliminary Experiments to Determine Reasonable Parameter Ranges

Before conducting experiments to evaluate the models, appropriate ranges for parameter values were determined, specifically for the number of clusters *k* and model termination time.

The range of *k* values was determined by plotting the within-cluster sum of squares (WCSS) from k-means and finding the region of the curve resembling an elbow (Figures 1 and 2). For both datasets, the reasonable range of *k* values was determined to be 3-5.

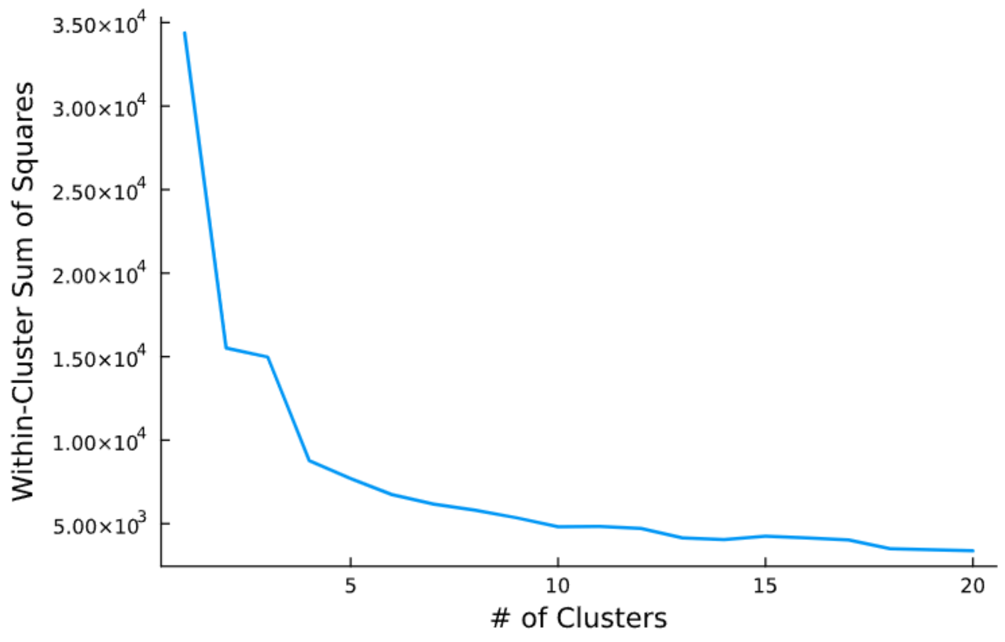

Figure 1: K-means Elbow Plot for Abalone Dataset

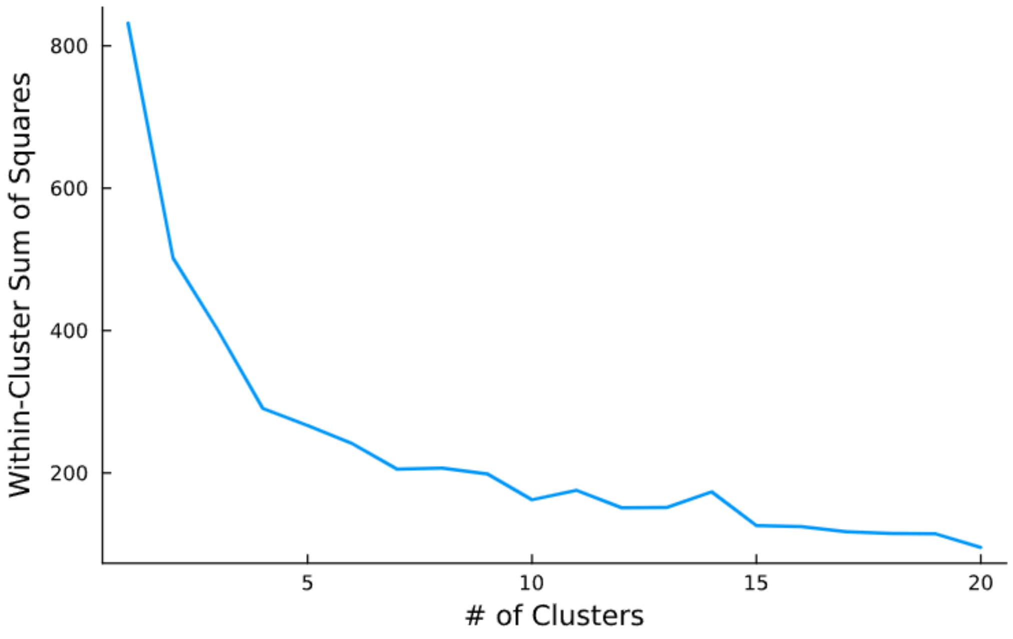

Figure 2: K-means Elbow Plot for Similarity Prediction Dataset

A common issue regarding MIO models is the time taken to converge to an optimal solution. To prevent overly long computation times, each model was given a time limit. A reasonable range of time limits was determined, ensuring the models still had enough time to make significant progress. Five k-means models with varying time limits were applied to each dataset, and the WCSS for each model was plotted to approximate the time required to reach an optimal solution. For both datasets, the models appear to find an optimal solution within 90 seconds (Figures 3 and 4). Thus, the reasonable range of time limits was determined to be 30, 90, and 180 seconds.

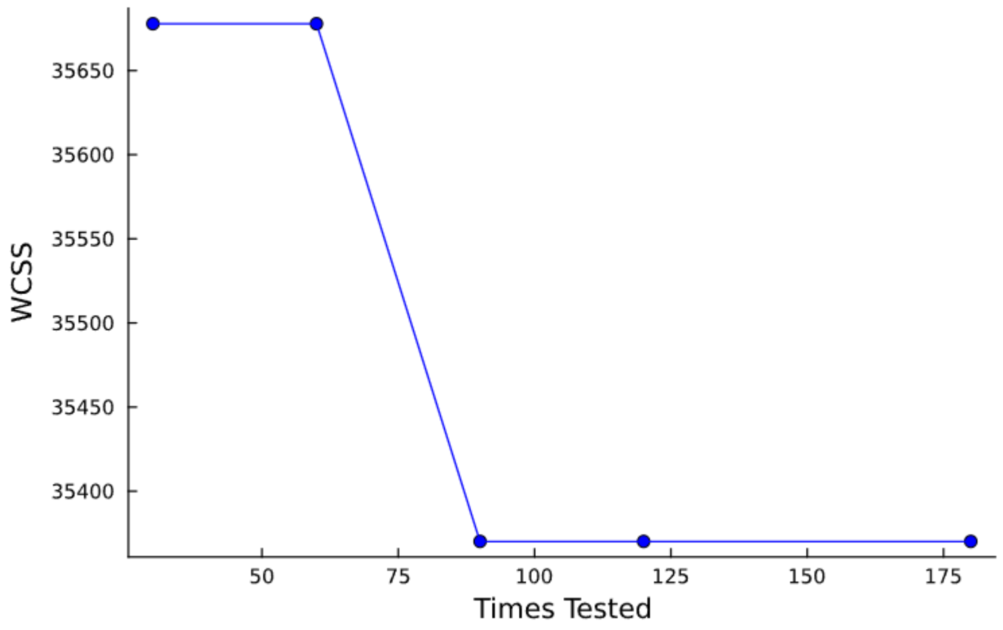

Figure 3: Optimal K-means WCSS as a Function of Time Limit; Abalone Dataset

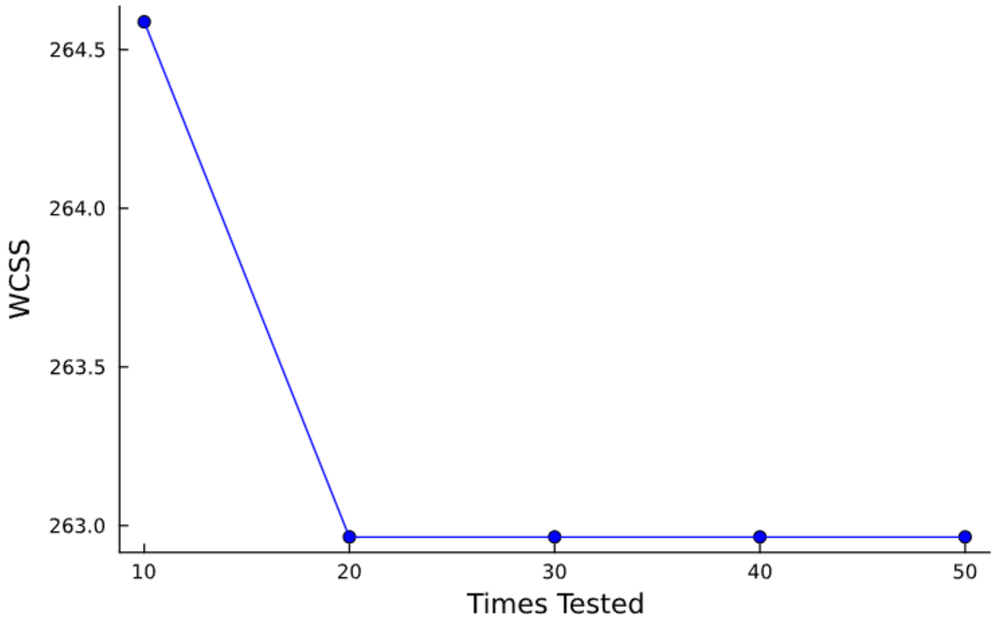

Figure 4: Optimal K-means WCSS as a Function of Time Limit; Similarity Prediction Dataset

Evaluation Methods and Metrics

As summarized by Table 1, various combinations of parameters were tested to evaluate the MIO models and compare them to the heuristic algorithms.

Table 1: MIO model parameters tested for each dataset (abalone and similarity prediction) and clustering algorithm (k-means and k-medoids).

Preprocessing	Percentage of Observations	Warm Start	Distance Metric	Number of Clusters	Time Limit [secs]
Normalized	10%	Yes	Euclidean	3	30
Scaled	25%	No	Manhattan	4	90
	75%			5	180

Both preprocessing techniques were tested to observe if the models perform better with normalized or min-max scaled data. Various-sized subsets were tested to observe the scalability of the models. Finally, the heuristic solutions were used as warm starts, when applicable, to test if using warm starts significantly improves solutions. Evaluating all possible parameter combinations resulted in 216 MIO models for each dataset and clustering algorithm.

The warm start, distance metric, and time limit parameters do not apply to the heuristic methods. They were implemented with Euclidean distance only, and the time limit parameter is not applicable since they always terminate within 15 seconds. Therefore, only 18 heuristic models were generated for each dataset and clustering algorithm.

WCSS and average silhouette score were the two main metrics used for evaluation. For models using Euclidean distance, WCSS is equivalent to the objective function being minimized. For models using Manhattan distance, WCSS differs slightly due to the distances being squared. The silhouette score accounts for a cluster's tightness and separation from disparate clusters. Each cluster's silhouette score was averaged to obtain one value per model.

KEY FINDINGS

Attached is a collection of figures that summarize the results for each set of parameters (Abalone and Similarity Prediction). These figures illustrate that there is no ubiquitous result that is consistent across every test. Indeed, a large variety of behaviors are observed across all experiments. However, there still exist some overarching trends in the results.

Table A1 in Appendix A provides several summary statistics comparing the models. In most scenarios, the optimization models achieved better results than the heuristic models in both WCSS and average silhouette scores, on average. Most notably, the optimal k-medoids models on the similarity prediction dataset achieved an average 14% improvement in WCSS over the heuristic model. However, the optimization models did not always perform better, such as the k-medoids models on the abalone dataset performing worse than the heuristic models, on average. This is likely due to the heuristic k-means algorithm only utilizing Euclidean distance whereas some of the optimization models utilize Manhattan distance, which does not punish large errors as severely.

Unsurprisingly, the optimization models with warm starts outperformed the optimization models with cold starts, on average. Considering these results and the ease of generating warm starts, most practical implementations of these models should utilize warm starts.

A major issue highlighted by Table A1 is the number of times the optimization models using Euclidean distance either did not converge (for cold starts) or did not improve the heuristic solution (for warm starts). Most notably, 80% of the k-means optimization models using Euclidean distance on the abalone dataset failed. This is a testament to the difficulty in dealing with quadratic constraints, exacerbated when scaling up to larger datasets.

The impact of preprocessing was analyzed, and the results are displayed by Tables A2 and A3 in Appendix A. For each preprocessing technique, the average percent improvement in average silhouette score over the heuristic solution was determined. For both datasets, k-medoids models performed better with normalized data than min-max scaled data, on average. The results for k-means models are less conclusive, as both preprocessing techniques yielded similar results. The type of preprocessing is highly dependent on the nature of the data, so more datasets should be tested to determine if k-medoids models truly perform better with normalized data.

The impact of dataset size was analyzed, and the results are displayed in Tables A4 and A5 in Appendix A. For each dataset size proportion, the average percent improvement in average silhouette score over the heuristic solution was determined. For both datasets, the subsets with 75% of observations performed the worst, on average. This is especially pronounced in the abalone dataset, which has many more observations than the similarity prediction dataset. This highlights the challenge of scaling with MIO models.

CONCLUSION

Optimal, consistent solutions are not guaranteed with heuristic problem-solving methods. However, heuristic methods can be developed into tractable optimization models to provide better solutions. The models presented here were created in an attempt to improve k-means and k-medoids clustering using MIO formulation strategies.

The results from experimenting across a variety of parameter combinations do not provide conclusive evidence that the MIO models consistently perform better than heuristic models. This inconsistency is most evident with models using Euclidean distance instead of Manhattan distance. Indeed, Gurobi struggled to make progress on finding optimal solutions for these models, speaking to the complexity that quadratic constraints introduce to optimization models.

The size and type of data also play a significant role in the tractability of the problem. The abalone dataset proved to be very difficult for Gurobi to make progress on, especially as the proportion of data used increased. Compared to the similarity prediction dataset, which provided more consistent MIO results, the results from the abalone dataset demonstrate the compounding effect of generating an exponential number of constraints and variables on tractability.

K-means and k-medoids are NP-hard optimization problems, making them notoriously hard to solve to optimality. Some aspects of the formulations were thus simplified to convert these algorithms into linear programs that Gurobi can handle. For example, Gurobi was left to decide the optimal values for centroids in the k-means models, since the direct calculation is nonlinear.

In future iterations of this work, it is recommended to find creative ways to transform these fractional, binary equations into linear terms. As seen in lectures for identifying exceptional responders in clinical trials, these types of terms can be linearized with the appropriate strategies. Still, the analyses performed in this work can be applied to other optimal clustering methods.

WORKS CITED

Nash,Warwick, Sellers,Tracy, Talbot,Simon, Cawthorn,Andrew, and Ford,Wes. (1995). Abalone. UCI Machine Learning Repository. https://doi.org/10.24432/C55C7W.

Gandini,Enrico. (2022). Similarity Prediction. UCI Machine Learning Repository. https://doi.org/10.24432/C5WW4F.

Abalone Results Analysis: https://github.com/zack-horton/ML-Project/blob/main/Abalone_Exploration.html

Similarity Prediction Results Analysis: https://github.com/zack-horton/ML-Project/blob/main/Similarity_Exploration.html

PARTNER CONTRIBUTIONS

We both split up the work very evenly, with Tanner focusing on the formulation and testing of the k-means clustering algorithm, and Zack taking on the same responsibilities for the k-medoids clustering algorithm. Before developing these algorithms, both Zack and Tanner met to discuss the potential difficulties we may face, what experiments and parameters we wanted to test, and to co-develop our final function which tests the algorithm across all parameter combinations to enable a comprehensive review of the techniques compared to the heuristic methods.

Creating the experimental data from each dataset took an hour to complete, on average around 6-8 hours per dataset, per algorithm. Proper planning was done between Zack and Tanner to allocate time effectively and allow the analysis and experimental testing to be completed before the deliverables were completed.

After the results of the experiments were finalized, Zack began to visualize the results using R (as seen in the markdown files submitted with this assignment) and Tanner began to create performance metrics we should present in our report and final presentation (as seen through the different metrics reported in the various tables in the report, presentations, and appendices). After collecting all materials, visualizing results, and calculating performance metrics, we both took time during the final weeks of school to develop this comprehensive report and finalize the details of the presentation.

Appendix A

Table A1: Aggregated results of parameter combinations across both abalone and similarity prediction datasets to compare optimization model performance to heuristic methods and cold-start solutions to warm-start solutions.

Analyzing Results	Similarity		Abalone	
	K-Means	K-Medoids	K-Means	K-Medoids
Avg % Improvement in WCSS: Heuristics to Opt. Warm Start	14%	15%	0.4%	-5.2%
Avg % Improvement in WCSS: Opt. Cold Start to Opt. Warm Start	4%	1%	50%	50%
Avg % Improvement in Silhouette: Heuristics to Opt. Warm Start	-1%	7%	-1%	14%
Avg % Improvement in Silhouette: Opt. Cold Start to Opt. Warm Start	4%	1%	80%	54%
Total % Euclidean Models w/o solution or no change to Heuristic	21%	11%	80%	14%

Table A2: Aggregated results displaying the effect of preprocessing on model performance for the similarity prediction dataset.

Similarity Dataset		Normalized	Scaled
Avg % Improvement in Silhouette: Heuristics to Opt. Warm Start	K-Means	-5%	2%
	K-Medoids	14%	-1%
Avg % Improvement in Silhouette: Heuristics to Opt. Cold Start	K-Means	-7%	-1%
	K-Medoids	14%	1%

Table A3: Aggregated results displaying the effect of preprocessing on model performance for the abalone dataset.

Abalone Dataset		Normalized	Scaled
Avg % Improvement in Silhouette: Heuristics to Opt. Warm Start	K-Means	-1%	-1%
	K-Medoids	21%	6%
Avg % Improvement in Silhouette: Heuristics to Opt. Cold Start	K-Means	-29%	-35%
	K-Medoids	-20%	-32%

Table A4: Aggregated results displaying the effect of dataset size on model performance for the similarity prediction dataset.

Similarity Dataset		10% Obs.	25% Obs.	75% Obs.
Avg % Improvement in Silhouette: Heuristics to Opt. Warm Start	K-Means	-6%	1%	1%
	K-Medoids	5%	11%	5%
Avg % Improvement in Silhouette: Heuristics to Opt. Cold Start	K-Means	-5%	-1%	-6%
	K-Medoids	6%	15%	2%

Table A5: Aggregated results displaying the effect of dataset size on model performance for the abalone dataset.

Abalone Dataset		10% Obs.	25% Obs.	75% Obs.
Avg % Improvement in Silhouette: Heuristics to Opt. Warm Start	K-Means	-1%	-1%	0%
	K-Medoids	15%	27%	-1%
Avg % Improvement in Silhouette: Heuristics to Opt. Cold Start	K-Means	-16%	-40%	-50%
	K-Medoids	-9%	-25%	-50%

Appendix B: K-means Clustering

Optimal K-Means Clustering:

Sets and Ranges:

Observations: $i \in [1, n]$

Variables/Factors: $j \in [1, p]$

Clusters: $k \in [1, K]$

Observed Data (normalized/scaled): x'_{ij}

Variables for MIO:

Observation i assigned to cluster k: $z_{ik} \in \{0, 1\}$

Values for cluster $k's$ centroids: $c_{kj} \in \mathbb{R}$

Distance between i and cluster $k's$ centroid: $d_{ij} \geq 0$

Constraints for MIO:

Ensure Cluster Assignment:

$$\sum_{k=1}^{K} z_{ik} = 1 \ , \quad \forall i \in [1, n]$$

Distance (if Manhattan): $d_{ij} = |x_{ij} - c_{kj}|$

$$d_{ij} \geq x_{ij} - c_{kj} - \mathbf{M} \times (1 - z_{ik}) \ , \quad \forall i \in [1, n], \quad \forall j \in [1, p], \quad \forall k \in [1, K]$$

$$d_{ij} \geq c_{kj} - x_{ij} - \mathbf{M} \times (1 - z_{ik}) \ , \quad \forall i \in [1, n], \quad \forall j \in [1, p], \quad \forall k \in [1, K]$$

Distance (if Euclidean): $d_{ij} = ||x_{ij} - c_{kj}||_2^2$

$$d_{ij} \geq (x_{ij} - c_{kj})^2 - \mathbf{M} \times (1 - z_{ik}) \ , \quad \forall i \in [1, n], \quad \forall j \in [1, p], \quad \forall k \in [1, K]$$

Objective Function:

$$\min \sum_{i=1}^{n} \sum_{j=1}^{p} d_{ij}$$

Figure B1: MIO Formulation for K-means Clustering

Appendix C: K-medoids Clustering

Optimal K-Medoids Clustering:

Sets and Ranges:

Observations: $i \in [1, n]$

Variables/Factors: $j \in [1, p]$

Clusters: $k \in [1, K]$

Observed Data (normalized/scaled): x'_{ij}

Variables for MIO:

Observation i assigned to cluster k: $z_{ik} \in \{0, 1\}$

Observation i is medoid of cluster k: $m_{ik} \in \{0, 1\}$

Values for cluster $k's$ medoid: $c_{kj} \in \mathbb{R}$

Distance between i and cluster $k's$ medoid: $d_{ij} \geq 0$

Constraints for MIO:

Ensure Cluster Assignment:

$$\sum_{k=1}^{K} z_{ik} = 1 \ , \quad \forall i \in [1, n]$$

Ensure Medoid Assignment:

$$\sum_{i=1}^{n} m_{ik} = 1 \ , \quad \forall k \in [1, K]$$

$$\sum_{k=1}^{K} m_{ik} \leq 1 \ , \quad \forall i \in [1, n]$$

$$m_{ik} \leq z_{ik} \ , \quad \forall i \in [1, n], \quad \forall k \in [1, K]$$

Determine Medoid Values:

$$x'_{ij} - \mathbf{M} \times (1 - m_{ik}) \leq c_{kj} \ , \quad \forall i \in [1, n], \quad \forall j \in [1, p], \quad \forall k \in [1, K]$$

$$c_{kj} \leq x'_{ij} + \mathbf{M} \times (1 - m_{ik}) \ , \quad \forall i \in [1, n], \quad \forall j \in [1, p], \quad \forall k \in [1, K]$$

Distance (if Manhattan): $d_{ij} = |x_{ij} - c_{kj}|$

$$d_{ij} \geq x_{ij} - c_{kj} - \mathbf{M} \times (1 - z_{ik}) \ , \quad \forall i \in [1, n], \quad \forall j \in [1, p], \quad \forall k \in [1, K]$$

$$d_{ij} \geq c_{kj} - x_{ij} - \mathbf{M} \times (1 - z_{ik}) \ , \quad \forall i \in [1, n], \quad \forall j \in [1, p], \quad \forall k \in [1, K]$$

Distance (if Euclidean): $d_{ij} = ||x_{ij} - c_{kj}||_2^2$

$$d_{ij} \geq (x_{ij} - c_{kj})^2 - \mathbf{M} \times (1 - z_{ik}) \ , \quad \forall i \in [1, n], \quad \forall j \in [1, p], \quad \forall k \in [1, K]$$

Objective Function:

$$\min \sum_{i=1}^{n} \sum_{j=1}^{p} d_{ij}$$

Figure C1: MIO Formulation for K-medoids Clustering