Erasmus University Rotterdam
Erasmus School of Economics
Master Thesis Business Analytics and Quantative Marketing

Sparse Mixed Convex Clustering

Yuan Zhang (581010yz)

Supervisor:	Cavicchia, C.
Second assessor:	Name of your second assessor
Date final version:	1st July 2024

Abstract

Clustering analysis is fundamental in unsupervised learning to to identify natural groupings within a dataset. Traditional clustering algorithms often encounter instability due to the non-convex nature of their objective functions. In contrast, convex clustering methods offer stability by optimizing a global objective, ensuring consistent clustering outcomes. This thesis introduces a novel approach designed to handle datasets with both numerical and categorical features. Building upon the AMA algorithm by Chi and Lange (2015) and incorporating methodologies from M. Wang and Allen (2021) and Witten and Tibshirani (2010), our convex clustering method integrates feature selection within a convex clustering framework. This method additionally addresses the challenge of high-dimensional data, where feature reduction is critical for effective clustering. Through extensive simulation experiments and application to gene expression datasets, our method demonstrates superior performance over traditional clustering methods. It effectively identifies both convex and non-convex cluster shapes while selecting features essential for accurate clustering. This capability proves crucial in scenarios with substantial feature noise. Overall, we present our method as a solution for clustering (high dimensional) mixed data, ensuring stable and accurate cluster identification in complex real-world datasets.

1 Introduction

Clustering analysis is a popular method in data mining and pattern recognition used to find natural groupings or clusters in a dataset. It helps uncover patterns and relationships in the data, making it easier to explore and make decisions. Clustering is used in many areas, such as customer segmentation, image analysis, and detecting anomalies. There are various algorithms that have unique ways of defining clusters and finding them efficiently. For example, hierarchical clustering algorithms (Patel, Sihmar & Jatain, 2015) look at how data points connect, k-means clustering (MacQueen et al., 1967) groups data around central points, and distribution based models like the Gaussian mixture model (Dempster, Laird & Rubin, 1977) assume a certain data distribution and calculate the probability that each observations belongs to a certain cluster.

However, most popular clustering methods suffer from instability due to the non-convex nature of the optimization problem, and this instability mainly comes from how they start the process. This general clustering approach selects certain observations as initial centroids, leading to varying cluster allocations with different starting centroids. To address this instability, one approach is to run the algorithm multiple times (Arthur & Vassilvitskii, 2006) with different starting centroids and select the cluster allocation based on criteria such as the Silhouette coefficient, Gap statistic, or Elbow method. However, these methods may yield inconclusive results as they may not agree on the optimal number of clusters. Alternatively, focusing on a centroid initialization scheme (Xu, Xu, Zhang, Zhang & Hou, 2009; Lemaire, Ismaili & Cornuéjols, 2015) rather than relying on chance selection may mitigate this issue, although it does not guarantee finding the optimal clustering.

Moreover, datasets often contain both numeric and categorical features, although standard clustering implementations typically assume datasets with solely numerical features. Both types of data can significantly influence object clustering, making a method capable of handling both types preferable. Fortunately, most clustering methods can be accommodated to handle categorical data by incorporating different loss functions for each data type. For example, K-prototypes (Huang, 1997) incorporates different losses for numerical and categorical data type, while hierarchical clustering employs predefined dissimilarity measures such as the Gower distance (Gower, 1971), which is a popular distance measure for mixed data types. However, these methods still face the same instability issues as their counterparts that handle only numerical data since they are based on the same non-convex algorithm. Additionally, not all features hold equal importance for clustering, with noisy features potentially leading to a less accurate clustering. An algorithm equipped to filter out unimportant features could enhance both cluster accuracy and interpretability.

To address the above problems, this thesis proposes a novel convex clustering algorithm capable of handling mixed data, including both numerical and categorical features. The algorithm also incorporates the functionality of feature selection, in order to yield interpretable clusters. Our algorithm extends the convex clustering algorithm based on AMA by Chi and Lange (2015)to incorporate categorical features. Moreover, by incorporating the feature selection framework proposed by Witten and Tibshirani (2010) and M. Wang and Allen (2021), the algorithm implements a more rigorous approach to select features. Summarizing, our method investigates the following research question: "How can we extend the convex clustering framework to incorporate both numerical and categorical data, while also performing feature selection in high-dimensional datasets?"

Our results demonstrate that our mixed convex clustering algorithm can nearly perfectly retrieve both spherical and non-spherical clusters. This is shown through simulations using data generated from Gaussian distributions as well as the Half-Moon dataset. By varying the amount of noise in the data, we found that convex clustering still effectively recovered the true cluster shapes. Furthermore, when applied to a high-dimensional mixed gene expression dataset, our method successfully selected the relevant features to accurately identify the true clusters. Therefore, our contribution to the literature lies in introducing a convex clustering algorithm adept at processing both numerical and categorical data in high-dimensional datasets.

In Section 2, we provide a general overview of convex clustering, examining different algorithms and applications. Following this, Section 3 briefly summarizes the dataset that is used in this study and how it is obtained. Section 4 provides a detailed explanation of our novel convex clustering approach. Then, we begin by evaluating its performance on simulated data, detailed in Section 5, where we analyze its ability to accurately detect clusters under controlled conditions. Moving to Section 6, we present the outcomes of applying our method to a real-world dataset, showcasing its performance in uncovering groups compared to other methods. Finally, in Section 7, we briefly summarize our findings, and discuss their implications, and highlight avenues for future research in the field of convex clustering.

2 Literature

Cluster analysis is the task of grouping a set of objects in such a way that objects within the same group are more similar to each other than to those in separate groups. A popular algorithm is the k-means algorithm. The k-means algorithm minimizes the following objective function:

$$\sum_{k=1}^{K} \sum_{i \in C_k} ||\mathbf{x}_i - \bar{\mathbf{x}}_k||^2 \tag{1}$$

The basic concept to solve this objective is the following: Beginning with an initial clustering that may not be optimal, move each point to its closest new center, adjust the clustering centers by computing the mean of the member points, and continue this relocation and updating process until certain convergence criteria. While this approach is conceptually simple and computationally scalable, it generally converges to spurious local solutions both in theory and in practice (Qian, Zhang & Chen, 2021). One way to deal with the initialization of the clusters is by using multiple random starts or choose the initial cluster centers carefully. For example, Su and Dy (2004) propose a deterministic method based on Principal component analysis, while Jia and Song (2020) propose an initialization scheme based on density. However, despite the potential improvements in clustering and convergence offered by these methods, there is still no guarantee of achieving an optimal global solution. In addition, this method proves to be less effective in high-dimensional datasets due to the presence of possible noise and outliers (X.-D. Wang, Chen & Yan, 2019). Moreover, this algorithm is confined to spherical cluster structures which limits its ability to capture more complex structures (Drineas, Frieze, Kannan, Vempala & Vinay, 2004). Challenges also arise with different variable types, namely categorical and numerical data.

Model-based clustering approaches represent another methodology for clustering objects. These approaches operate under the assumption that observations are generated from specific distributions. For instance, in Gaussian Mixture Models (GMMs), it is assumed that the data arise from a mixture of K components, each characterized by its own Gaussian density parametrized by mean μ and covariance matrix Σ. The objective is to minimize the log likelihood function given by:

$$\prod_{i=1}^{n} \sum_{k=1}^{K} \pi_k \phi(x_i|\mu_k, \Sigma_k) \tag{2}$$

Dempster et al. (1977) introduced the Expectation-Maximization (EM) algorithm to optimize this objective. Despite the popularity of this method, one drawback of model-based approaches lies in the assumption of a predefined underlying distribution. Moreover, similar to the K-means algorithm, the EM algorithm is susceptible to convergence to local optima, which can lead to instability in the clustering results.

Another group of clustering methods diverge from optimizing explicit objective functions and instead adopt heuristic approaches. These include connectivity-based methods like hierarchical clustering and density-based methods such as Density-based spatial clustering of applications with noise (DBSCAN) (Ester, Kriegel, Sander, Xu et al., 1996). However, these methods often require more careful parameter tuning than models which optimize an objective. For example, hierarchical clustering includes various linkages such as single linkage (which connects clusters based on their closest points) and average linkage (which connects clusters based on the average distance). Each method may yield different clustering outcomes. Thus, the selection of parameters plays a crucial role.

Furthermore, addressing high-dimensional datasets may entail applying feature selection procedures, such as decomposing the original data matrix into a smaller, sparse matrix and conducting clustering on it (Tamayo et al., 2007; Bernardo et al., 2003). However, Witten and Tibshirani (2010) argue that there is no assurance that this decomposed matrix contains the desired signal for clustering detection. In fact, Chang (1983) explore the impact of performing Principal Component Analysis (PCA) to reduce data dimensionality before clustering and discover that the principal components with the largest eigenvalues may not always yield optimal subgroup separation. Therefore, an alternative approach involves applying a penalty to induce sparsity in the features (S. Wang & Zhu, 2008; Xie, Pan & Shen, 2008), as model-based clustering naturally lends itself to this approach. This is the approach we will adopt as well.

To deal with the problem of local optimum, we utilize the convex clustering method. The earliest mentions of convex clustering are by Pelckmans, De Brabanter, Suykens and De Moor (2005); Lindsten, Ohlsson and Ljung (2011); Hocking, Vert, Bach and Joulin (2011). Lindsten et al. (2011) formulates convex clustering as a convex relaxation of k-means clustering, while Hocking et al. (2011) explain it as a convex relaxation of hierarchical clustering. This approach formulates the clustering task as a convex optimization problem, such that we do end up in a global solution. Lindsten et al. (2011) also show that convex clustering is able to capture non-convex cluster shapes. The objective that we seek to minimize is the following:

$$\min_{\mathbf{A}\in\mathbb{R}^{n\times p}} \quad \frac{1}{2}\sum_{i=1}^{n}||\mathbf{x}_{i\cdot}-\mathbf{a}_{i\cdot}||_q^2+\gamma\sum_{l\in\mathcal{E}}w_l||\mathbf{a}_{l_1}-\mathbf{a}_{l_2}||_q, \tag{3}$$

where $\mathbf{x}_i$ denotes the data for observation i and $\mathbf{a}_i$ denotes the cluster centroids assigned to observation i. The algorithm groups observations into the same cluster by determining whether the cluster centroids are close enough, a process controlled by γ in the second term, which encourages shrinkage of cluster centroids.

Several studies have delved into the properties of convex clustering. Tan and Witten (2015) establish that convex clustering is closely related to single linkage hierarchical clustering and k-means clustering, determining the tuning parameter range for non-trivial solutions and offering a finite sample bound for prediction error. Panahi, Dubhashi, Johansson and Bhattacharyya (2017) and Zhu, Xu, Leng and Yan (2014) prove perfect recovery of the convex clustering model with uniformly weighted all-pairwise-differences regularization, Additionally, Sun, Toh and Yuan (2021) establish sufficient conditions for the perfect recovery guarantee of the general weighted convex clustering model.

Despite the advantages, convex clustering's adoption remains limited due to its computationally intensive nature (Weylandt, Nagorski & Allen, 2020). This motivates our focus on achieving fast computation time, a key advantage of methods like K-Means and hierarchical clustering. Fortunately, several efficient algorithms have been proposed for convex clustering. Chi and Lange (2015) present two splitting splitting methods for convex clustering based on Alternating Direction Method of Multipliers (ADMM) and Alternation Minimization Algorithm (AMA). Weylandt et al. (2020) extends these methods by applying Algorithmic Regularization Paths, which showed increases in computation time. Additionally, Sun et al. (2021) propose an efficient algorithm based on a semismooth Newton-based augmented Lagrangian method, while Touw, Groenen and Terada (2023) utilize majorization minimization for convex clustering.

Thus, multiple efficient algorithms for convex clustering have been proposed. However, addressing datasets with high dimensions and categorical data poses additional challenges. B. Wang, Zhang, Sun and Fang (2018) tackle the high-dimensional clustering scenario by employing an adaptive group-lasso penalty. They formulate sparse convex clustering for purely numerical data as follows:

$$\min_{\mathbf{A}\in\mathbb{R}^{n\times p}} \quad \frac{1}{2}\sum_{i=1}^{n}||\mathbf{x}_{i\cdot}-\mathbf{a}_{i\cdot}||_q^2+\gamma\sum_{l\in\mathcal{E}}w_l||\mathbf{a}_{l_1}-\mathbf{a}_{l_2}||_q+\beta\sum_{j=1}^{p}\mu_j||\mathbf{a}_j||_2^2, \tag{4}$$

where we add an additional parameter β that penalizes large values of feature values, and μ_j is an adaptive weight that adjusts the weights of the features based on their importance. In particular, the data is centered such that we expect feature j to be informative if $||\mathbf{a}_j||_2^2 > 0$.

As an extension, M. Wang and Allen (2021) introduce the Integrative Generalized Convex Clustering Optimization (iGecco) method, which integrates various data views, including numerical and categorical data. iGecco, based on ADMM, performs feature selection by incorporating penalties and weights on the features using a similar adaptive group-lasso penalty as B. Wang et al. (2018). Each feature is shrunken towards their respective loss center. The relevant features are then features that are deviate from their specific loss center. Their generalized formulation is described as follows:

$$\min_{\mathbf{A}\in\mathbb{R}^{n\times p}} \quad \sum_{i=1}^{n}\sum_{m=1}^{M}\mathcal{L}^m(\mathbf{x}_i^m,\mathbf{a}_i^m)+\gamma\sum_{l\in\mathcal{E}}w_l\sqrt{||\mathbf{a}_{l_1}-\mathbf{a}_{l_2}||_2^2}+\beta\sum_{j=1}^{p_m}\mu_j^{(m)}||\mathbf{a}_j-\tilde{\mathbf{a}}_j||_2, \tag{5}$$

where $\mathcal{L}^m$ denotes the loss function for data type m and $\tilde{\mathbf{a}}_j$ denotes the loss-specific centers for each data view. M. Wang and Allen (2021) note that their method can be seen as a generalization of the sparse convex clustering method by B. Wang et al. (2018) as it incorporates different losses for various data views, while also employing an identical Lasso-type penalty. Specifically, iGecco represents the sparse convex clustering method by B. Wang et al. (2018) when the data type is solely numerical, utilizing a squared Euclidean loss function and centered data.

Our approach shares similar objectives with M. Wang and Allen (2021) but takes a different route. Chi and Lange (2015) highlight the efficiency of AMA compared to the ADMM. Therefore, our approach incorporates the AMA algorithm. Furthermore, rather then only employing an adaptive group lasso penalty, we include a Ridge penalty which aims to shrink features towards their specific loss function. Our objective with this penalty is to solely shrink the variables without selecting them.

Given the critical role of feature selection in clustering, our objective is to implement a more rigorous approach to ensure the selection of optimal features. For feature selection, we combine the feature penalty approach with a different framework for feature selection in clustering, specifically a lasso-type penalty proposed by Witten and Tibshirani (2010), which also considers inter- and intra-cluster similarity. Their method proved to be very effective in high-dimensional dataset.

However, we note that Witten and Tibshirani (2010) employed the standard K-means method, which is known for its sensitivity to local optima. Consequently, our method can be viewed as an extension of their framework, as we incorporate a convex clustering method that achieves global solutions and while also modifying the feature selection approach. Additionally, our convex clustering method accommodates both numerical and categorical features.

3 Data

We assess the performance of our mixed convex clustering algorithm on datasets that contain both numerical and categorical data. To achieve this, we utilize a combination of simulated and various real datasets. In the simulated datasets, we control over the number of features and noise levels to evaluate the algorithm's robustness. The simulation study is further described in Section 5. Moreover, for our real datasets, we focus on a high-dimensional mixed datasets.

In particular, we use a Gene expression dataset that originates from a proof-of-concept study published by Golub et al. (1999). The dataset is available from `https://www.kaggle.com/datasets/crawford/gene-expression?select=data_set_ALL_AML_independent.csv`. Golub et al. (1999) aim to develop a generic approach to cancer classification based on gene expression monitoring by DNA microarrays. They used human acute leukemias as a test case. A class discovery procedure correctly discovered the difference between acute myeloid leukemia (AML) and acute lymphoblastic leukemia (ALL) without prior knowledge of the classes.

However, in our study, we approach the dataset with the goal of clustering rather than classification. Our objective is to evaluate the quality of clusters obtained by clustering algorithms when the true class labels (AML and ALL) are treated as potential clusters. This approach allows us to assess how well clustering methods can group similar gene expression profiles without prior knowledge of class labels.

The study utilizes two datasets: an initial training dataset comprising 38 samples and an independent test dataset consisting of 34 samples. These datasets include numerical measurements from bone marrow and peripheral blood samples for both acute lymphoblastic leukemia (ALL) and acute myeloid leukemia (AML). The intensity values have been rescaled to ensure that the overall intensities for each chip are equivalent. Moreover, the datasets contains categorical features which indicate whether a certain gene is abstent (A), present (P) or marginal (M), which indicates that it is too close to call. Liu et al. (2002) further explain the construction of these call columns. There are exactly 7126 numerical and categorical features, where each numerical and categorical feature corresponds to a gene expression.

Lastly, we note that the initial and training datasets are both imbalanced. The initial dataset contains 38 observations, including 27 ALL patients and 11 AML patients. While the independent dataset contains 34 observations consisting of 20 ALL patients and 14 AML patients. Thus, the imbalance ratio is 2.45 for the training data and 1.43 for the independent dataset.

4 Methodology

Several algorithms have been proposed to solve convex clustering efficiently. Touw et al. (2023) introduce convex clustering through majorization-minimization (CCMM), employing iterative procedures involving cluster fusions and a highly efficient updating scheme utilizing diagonal majorization. On the other hand, Sun et al. (2021) opted for a semismooth Newton-based augmented Lagrangian method. Through extensive numerical experiments conducted on simulated and real data, they demonstrate that their algorithm is highly efficient and robust for solving large-scale problems. Additionally, Chi and Lange (2015) put forth two splitting methods based on Alternating Direction Method of Multipliers (ADMM) and Alternating Majorization-Minimization Algorithm (AMA). Notably, they find that AMA outperforms ADMM significantly. In this thesis, the choice falls on the AMA algorithm for two reasons. Firstly, AMA exhibits notable computational efficiency, which is crucial for addressing high-dimensional convex clustering problems effectively. Secondly, the algorithm has an elegant simplicity, marked by straightforward steps and a rigorous stopping criterion for achieving optimality. Hence, the AMA algorithm is selected as the preferred approach for the clustering tasks undertaken in this thesis.

The methodology section is organized as follows: Section 4.1 provides an overview of the general convex clustering framework. The extension of this framework to accommodate both numerical and categorical data is discussed in Section 4.2. Following this, Section 4.3 addresses tuning of the weights in convex clustering and balancing the contributions of numerical and categorical features. Choosing the weights smartly can drastically improve cluster accuracy and computational efficiency. Subsequently, we detail the implementation of convex clustering for mixed data in Section 4.4. This section covers the main topics including an explanation of the modified AMA algorithm, cluster fusions,feature selection scheme and tuning considerations. Lastly, we introduce the benchmark models utilized in this thesis in Section 4.5, and the evaluation metrics employed are described in Section 4.6.

4.1 Convex Clustering Framework

Convex clustering calculates cluster centroids by solving the minimization of a convex function. Rather then assigning each observation to a cluster, it assigns each observation to a cluster centroid. If the cluster centroids of two observations are sufficiently similar, then they belong to the same cluster. Let $\mathbf{X} \in \mathbb{R}^{n \times p}$ denote the data matrix, where n is the amount of observations and p the amount of features, and $\mathbf{A} \in \mathbb{R}^{n \times p}$ indicate the cluster centroids associated with each observation. Hence, each $\mathbf{x}_i \in \mathbb{R}^p$ for $i = 1, \dots, n$ is assigned to a cluster centroid $\mathbf{a}_i$ for $i = 1, \dots, n$. The convex clustering objective can then be formulated as follows:

$$\min_{\mathbf{A} \in \mathbb{R}^{n \times p}} \quad \frac{1}{2} \sum_{i=1}^{n} ||\mathbf{x}_{i\cdot} - \mathbf{a}_{i\cdot}||_q^2 + \gamma \sum_{l \in \mathcal{E}} w_l ||\mathbf{a}_{l_1} - \mathbf{a}_{l_2}||_q, \tag{6}$$

where $\mathcal{E} = \{l = (l_{i_1}, l_{i_2}) : w_l > 0, i_1 < i_2\}$ is the set defined over all pairs of centroid centers with non-zero weights w_l. Moreover γ controls for the amount of cluster fusions. The larger the value of γ, the more cluster centroids become fused together. When $\gamma = 0$, the optimal solution for $\hat{\mathbf{A}}$ is the input matrix $\mathbf{X}$. Conversely, as γ tends to infinity, the optimal solution $\hat{\mathbf{A}}$ converges to the average matrix $\bar{\mathbf{X}}$.

4.2 Convex Clustering with numerical and categorical data

The previous described convex clustering framework is only applicable to numerical data. It is not uncommon for datasets to contain a mix of both numerical and categorical data. Effectively utilizing both types of data in the clustering process may improve the clustering performance. As such, we extend the convex clustering framework to accommodate both numerical and categorical variables similar to M. Wang and Allen (2021). The mixed convex clustering objective is then formulated as follows

$$\min_{\mathbf{A}\in\mathbb{R}^{n\times p}} \quad \sum_{i=1}^{n} \alpha\mathcal{L}^{num}(\mathbf{x}_i^{num}, \mathbf{a}_i^{num}) + (1-\alpha)\mathcal{L}^{cat}(\mathbf{x}_i^{cat}, \mathbf{a}_i^{cat}) + \gamma \sum_{l\in\mathcal{E}} w_l ||\mathbf{a}_{l_1} - \mathbf{a}_{l_2}||_q, \tag{7}$$

where $\mathcal{L}^{num}$ and $\mathcal{L}^{cat}$ denote the loss functions for numerical and categorical variables respectively. Moreover, $\mathbf{x}_i^{num}, \mathbf{a}_i^{num} \in \mathbb{R}^m$ and $\mathbf{x}_i^{cat}, \mathbf{a}_i^{cat} \in \mathbb{R}^{m-p}$ are the vectors corresponding to numerical and categorical data assuming there are m numerical features and $p-m$ categorical features. An important aspect is that both loss functions have to be convex in order to ensure convexity of the problem. Lastly, we add weights α to control for the trade off between numerical and categorical loss functions which is a similar approach as the K-Prototypes algorithm by Huang (1998). These weights are necessary as the categorical or numerical loss functions may dominate over one another. In our experiments, we find that the output of the categorical loss function tends to be larger than the numerical loss function.

4.2.1 Loss functions

We employ the standard convex squared Euclidean norm for numerical data which is used in most convex clustering applications. This loss function penalizes large differences proportionally more than small differences, owing to the squaring operation.

$$\mathcal{L}^{num} = \frac{1}{2}||\mathbf{x}_i^{num} - \mathbf{a}_i^{num}||^2, \quad i = 1, \dots, n \tag{8}$$

For categorical features, there is no direct translation from categorical to numerical values. Therefore, we utilize dummy coding to represent the categorical data. Naturally, in this context, it is more appropriate to select a binary categorical loss function. Additionally, since we aim for interpretable cluster centers, the ideal range for a center's value should be between 0 and 1. Hence, we employ the binomial deviance formulated as:

$$\begin{aligned} \mathcal{L}^{cat} &= \sum_{j=m+1}^{p} \left\{ \sum_{k=1}^{K} -x_{ijk}\log(a_{ijk}) - (1-x_{ijk})\log(\mathbf{1} - a_{ijk}) \right\}, \quad && i = 1, \dots, n \\ &= -\mathbf{x}_i^{cat} \cdot \log(\mathbf{a}_i^{cat}) - (1 - \mathbf{x}_i^{cat}) \cdot \log(\mathbf{1} - \mathbf{a}_i^{cat}), && i = 1, \dots, n \end{aligned} \tag{9}$$

where we assume that $\log(0) = 0$. This scenario arises in the case of perfect separation between clusters for a binary categorical variable.

A rather intuitive understanding is that these loss functions tend to shrink the numerical feature centroids towards the average value, while for categorical features, they shrink each categorical center towards the proportion associated with a specific category within that feature.

We refer to these properties as the loss-specific center since these values minimize the convex clustering objective 7 if $\gamma = 0$.

Lastly, note that the numerical loss function is the squared Euclidean norm which typically implies that the retrieved clusters should be spherical. At first glance, this raises questions about how the convex clustering framework can handle non-convex cluster shapes, as demonstrated in Hocking et al. (2011). Essentially, this can be understood by viewing which forces impact the estimated cluster centroids $\mathbf{A}$.

The numerical and categorical loss functions aim to keep each cluster centroid close to the original observation. Simultaneously, the penalty on the differences between cluster centroids encourages the cluster centroids with positive weights to be similar. As γ increases, the optimal solution to the objective function balances these two forces.

Non-convex cluster shapes are generally characterized by irregular shapes, varying densities, and connected components. By carefully choosing which cluster centroids $\mathbf{a}_i$, receive a positive weight w_l, we can capture these non-convex shapes. The key idea is to establish positive weights only between observations that are close to the original observations. This approach allows for the formation of a long chain of connected cluster centroids. The construction of these weights will be described in the next section.

4.3 Weights and balancing variable types contribution

Hocking et al. (2011) are likely the first reference to include this weight w_j in Equation 6, which aimed at producing a clusterpath that is sensitive to local density in the data. Chi and Lange (2015) employed similar exponential decaying weights, while also utilizing the K-Nearest Neighbors (K-NN) algorithm. They note that the choice of the weights can improve both computational efficiency and clustering quality. Consequently, we utilize these weights as well. The weights can be formulated as

$$w_l = \mathbb{I}\{i_2 \text{ K-NN of } \ i_1\}\exp(-\phi d(\mathbf{x}_{i_1}, \mathbf{x}_{i_2})), \quad \forall i_1, i_2 \tag{10}$$

where the first part reflects the K-NN and the second part a Gaussian kernel that slows the coalescence of distant observations. Moreover, ϕ is non-negative and controls for how fast the weights decrease to zero and $d(\cdot,\cdot)$ denotes the distance between two points.

A common method to calculate the dissimilarity of two items with numerical and categorical variables is the Gower distance. The Gower distance adopts Max-min standardization for numerical variable and a simple matching scheme for categorical variables.

$$d(\mathbf{x}_{i_1}, \mathbf{x}_{i_2}) = \sum_{j=1}^{m} \frac{|x_{i_1,j} - x_{i_2,j}|}{|\max(j) - \min(j)|} + \sum_{j=m+1}^{p} \mathbb{I}[x_{i_1,j} = x_{i_2,j}], \tag{11}$$

However, we emphasize the importance of achieving balanced contributions from each feature to both the convex clustering objective 7 and the weighting scheme 11. If certain variables are defined on larger scales, they might negatively influence the analysis, potentially reducing the contributions of other variables. Consequently, it is common practice to standardize the data prior to analysis, similar to methods such as PCA. However, standardization alone may not fully address this issue in our context.

For instance, in our weighting calculations, numerical variables undergo min-max standardization to confine them within the [0,1] range, mirroring the treatment of categorical variables in terms of scale [0,1]. Despite being standardized, we note that the average contribution of variables may not necessarily be equal. While categorical features yield a distance of 1 when dissimilar, numerical variables attain a distance of 1 only if the observations represent the maximum and minimum values. Given that most observations fall between these extremes, they rarely achieve a distance of 1. This disparity can potentially skew the weighting scheme, leading to less accurate neighbors to be chosen in the K-NN scheme.

Consequently, we propose the following idea in order to let the numerical and categorical features contribute equally to the calculation of the weights 11 and the convex clustering objective 7. Let $d_{i_1,i_2,j} = |x_{i_1,j} - x_{i_2,j}|$ be the distance between observation i_1 and i_2 for feature j. Then, $\mathrm{E}[d_{i_1,i_2,j}]$ denotes the expected value of the distance between two random observations i_1 and i_2 for feature j. Our goal is to set $\mathrm{E}[d_{i_1,i_2,j}] = 1$ for every feature j. We aim to achieve this as follows. First, calculate the estimate of the expected distance for feature j or $\hat{\mathrm{E}}[d_{i_1,i_2,j}]$ using the plug-in estimator:

$$\hat{\mathrm{E}}[d_{i_1,i_2,j}] = \frac{n(n-1)}{2} \sum_{i_1<i_2} d_{i_1,i_2,j} \tag{12}$$

then, we scale all observations of feature j with

$$\frac{1}{\hat{\mathrm{E}}[d_{i_1,i_2,j}]} \tag{13}$$

Moreover, let $\mathbf{x}_j$ be the column that contains all n observation for feature j. Then we continue to work with the scaled features:

$$\mathbf{x}_j^{new} = \frac{\mathbf{x}_j^{old}}{\hat{\mathrm{E}}[d_{i_1,i_2,j}]}, \quad j = 1, \dots, p \tag{14}$$

In the Gower distance, we use Max-min standardization for numerical features and a matching scheme for categorical variables. We can achieve a similar expected value for each feature in the following manner. The expected Max-min standardization distance for feature j is:

$$\hat{\mathrm{E}}[d_{i_1,i_2,j}] = \frac{n(n-1)}{2} \sum_{i_1<i_2} \frac{|x_{i_1} - x_{i_2}|}{|max(j) - min(j)|} = b \tag{15}$$

Then if we scale the features as shown in Equation 14, the expected distance for feature j is:

$$\begin{aligned}
\hat{\mathrm{E}}[d_{i_1,i_2,j}] &= \frac{n(n-1)}{2} \sum_{i_1<i_2} \frac{|\frac{x_{i_1}}{b} - \frac{x_{i_2}}{b}|}{|max(j) - min(j)|} \\
&= \frac{n(n-1)}{2} \frac{1}{b} \sum_{i_1<i_2} \frac{|x_{i_1} - x_{i_2}|}{|max(j) - min(j)|} \\
&= \frac{n(n-1)}{2} \frac{2}{n(n-1)} \sum_{i_1<i_2} \frac{|max(j) - min(j)|}{|x_{i_1} - x_{i_2}|} \sum_{i_1<i_2} \frac{|x_{i_1} - x_{i_2}|}{|max(j) - min(j)|} \\
&= 1
\end{aligned} \tag{16}$$

Note that Max-min standardization is not necessary because the ranges of each feature cancel out in the final derivations. Intuitively, Max-min standardization is essentially the absolute distance multiplied by a constant, which is the range of the feature. By multiplying each observation by the expected value, we 'override' this scaling factor, ensuring that the average expected absolute distance is one.

And similarly for the categorical variables, we have

$$\hat{\mathrm{E}}[d_{i_1,i_2,j}] = \frac{n(n-1)}{2} \sum_{i_1<i_2} \mathbb{I}[x_{i_1,j} = x_{i_2,j}] = b \tag{17}$$

Thus, if we scale the categorical distances by b, we obtain an expected value of one:

$$\begin{aligned}\hat{\mathrm{E}}[d_{i_1,i_2,j}] &= \frac{n(n-1)}{2}\frac{1}{b} \sum_{i_1<i_2} \mathbb{I}[x_{i_1,j} = x_{i_2,j}] \\ &= \frac{n(n-1)}{2}\frac{2}{n(n-1)}\frac{\sum_{i_1<i_2}\mathbb{I}[x_{i_1,j} = x_{i_2,j}]}{\sum_{i_1<i_2}\mathbb{I}[x_{i_1,j} = x_{i_2,j}]} \\ &= 1\end{aligned} \tag{18}$$

4.4 Implementation

Various algorithm have been proposed to solve the convex clustering algorithms. In this thesis, we adopt a splitting method proposed by Chi and Lange (2015) for convex clustering. Chi and Lange (2015) present two splitting techniques: one utilizing the ADMM, and the other employing an instance of AMA. Given that AMA demonstrated significantly superior efficiency compared to ADMM, we opt to implement the AMA algorithm.

4.4.1 Mixed-AMA based convex clustering (MAMAC)

AMA can be motivated as a variant of the augmented Lagrangian method (ALM). The steps to arrive at the AMA formulation are as follows. We first recast the problem as a constrained optimization problem.

$$\begin{aligned}&\min_{\mathbf{A}\in\mathbb{R}^{n\times p}} \quad \sum_{i=1}^{n} \alpha\mathcal{L}^{num}(\mathbf{x}_i^{num}, \mathbf{a}_i^{num}) + (1-\alpha)\mathcal{L}^{cat}(\mathbf{x}_i^{cat}, \mathbf{a}_i^{cat}) + \gamma \sum_{l\in\mathcal{E}} w_l ||\mathbf{v}_l||_q, \\ &\text{subject to } \mathbf{a}_{l_1} - \mathbf{a}_{l_2} - \mathbf{v}_l = 0 \quad \forall l \in \mathcal{E}\end{aligned} \tag{19}$$

ALM solves the equivalent problem in Equation 20. The equivalence between Equations 19 and 20 arises from the fact that their objective functions coincide for any point that satisfies the equality constraint.

$$\begin{aligned}&\min_{\mathbf{A},\mathbf{V}\in\mathbb{R}^{n\times p}} \quad \sum_{i=1}^{n} \alpha\mathcal{L}^{num}(\mathbf{x}_i^{num}, \mathbf{a}_i^{num}) + (1-\alpha)\mathcal{L}^{cat}(\mathbf{x}_i^{cat}, \mathbf{a}_i^{cat}) + \gamma \sum_{l\in\mathcal{E}} w_l ||\mathbf{v}_l||_q + \frac{\sigma}{2}\sum_{l\in\mathcal{E}} ||\mathbf{v}_l - \mathbf{a}_{l_1} + \mathbf{a}_{l_2}||_2^2, \\ &\text{subject to } \mathbf{a}_{l_1} - \mathbf{a}_{l_2} - \mathbf{v}_l = 0 \quad \forall l \in \mathcal{E}\end{aligned} \tag{20}$$

Moreover, minimizing an equality constrained optimization problem is equivalent to the identifying the saddle point of the associated Lagrangian function. The augmented Lagrangian function that AMA minimizes over for a given parameter $\sigma > 0$ is formulated as

$$\begin{aligned}\mathscr{L}(\mathbf{a}, \mathbf{v}, \Lambda) = &\sum_{i=1}^{n} \alpha \mathcal{L}^{num}(\mathbf{x}_i^{num}, \mathbf{a}_i^{num}) + (1-\alpha)\mathcal{L}^{cat}(\mathbf{x}_i^{cat}, \mathbf{a}_i^{cat}) + \gamma \sum_{l \in \mathcal{E}} w_l ||\mathbf{v}_l||_q \\ &+ \frac{\sigma}{2} \sum_{l \in \mathcal{E}} ||\mathbf{v}_l - \mathbf{a}_{l_1} + \mathbf{a}_{l_2}||_2^2 + \sum_{l \in \mathcal{E}} \langle \boldsymbol{\lambda}_l, \mathbf{v}_l - \mathbf{a}_{l_1} + \mathbf{a}_{l_2} \rangle\end{aligned} \tag{21}$$

As the optimization over u and a jointly is generally difficult for the augmented Lagrangrian, AMA (and ADMM) employ a different strategy in simplying the minimization subproblems. In AMA, this is achieved by minimizing the augmented Lagrangian one block of variables at a time. The process can be described as follows:

$$\begin{aligned}\mathbf{a}^{t+1} &= \arg\min_{\mathbf{a}} \mathscr{L}_v(\mathbf{a}, \mathbf{v}^t, \boldsymbol{\Lambda}^t) \\ \mathbf{v}^{t+1} &= \arg\min_{\mathbf{v}} \mathscr{L}_v(\mathbf{a}^{t+1}, \mathbf{v}, \boldsymbol{\Lambda}^t) \\ \boldsymbol{\Lambda}^{t+1} &= \arg\min_{\Lambda} \mathscr{L}_v(\mathbf{a}^{t+1}, \mathbf{v}^{t+1}, \boldsymbol{\Lambda})\end{aligned} \tag{22}$$

Similar to Chi (2015), we update the steps for $(\mathbf{v}, \boldsymbol{\Lambda})$ in parallel to theirs, as these variables are independent of the loss functions. However, due to the addition of another loss function in the objective, the steps to derive the final algorithm differ slightly, although ultimately resuls in a similar algorithm. Detailed derivations for the steps of $(\mathbf{a}, \mathbf{v}, \boldsymbol{\Lambda})$ are provided in Appendix A. The solutions are as follows:

$$\begin{aligned}\mathbf{a}_{num}^{t+1} &= \frac{\alpha \mathbf{x}_i^{num} + \sum_{l_1 = i} \boldsymbol{\lambda}_l^{num} - \sum_{l_2 = i} \boldsymbol{\lambda}_l^{num}}{\alpha} \\ \mathbf{a}_{cat}^{t+1} &= \frac{(1-\alpha)\mathbf{x}_i^{cat} + \sum_{l_1 = i} \boldsymbol{\lambda}_l^{cat} - \sum_{l_2 = i} \boldsymbol{\lambda}_l^{cat}}{1-\alpha}) \\ \mathbf{v}_l^{t+1} &= \left[1 - \frac{\frac{\gamma w_l}{\sigma}}{||\mathbf{a}_{l1}^{t+1} - \mathbf{a}_{l2}^{t+1} - \sigma^{-1}\boldsymbol{\lambda}_l^m||_2}\right]_+ (\mathbf{a}_{l1}^{t+1} - \mathbf{a}_{l2}^{t+1} - \sigma^{-1}\boldsymbol{\lambda}_l^m) \\ \boldsymbol{\Lambda}^{t+1} &= \boldsymbol{\lambda}_l^t + \sigma(\mathbf{v}_l^{t+1} - \mathbf{a}_{l_1}^{t+1} + \mathbf{a}_{l_2}^{t+1})\end{aligned} \tag{23}$$

Moreover, Chi and Lange (2015) note that simplifications can be made by using Moreau's decomposition, which we also briefly cover in Appendix A. As a result, we do not need to updat v_l and instead only need to store the updates for $\mathbf{a}$ and Λ. The AMA algorithm is shown in Algorithm 1.

This algorithm first starts by calculating the clustering centers. For this, we need the differences between the dual variables λ which is stored in Δ. Then, we calculate the differences between the cluster centroids which is stored in $\mathbf{g}$. Following this, we update λ by projecting the differences between the previous λ and $\mathbf{g}$ on the set C_l. This iterative process continues until convergence, which we will elaborate on in the subsequent section

Lastly, as noted out by Chi and Lange (2015), this bears resemblance to a projected gradient algorithm. Additionally, they note that Tseng (1991) demonstrates that AMA effectively per-

Algorithm 1 AMA Algorithm for mixed data

```
1: Initialize $\boldsymbol{\lambda}^0$, $\alpha_o = 1$
2: for t = 1,2,3,... do
3:     for i = 1,..., n do
4:         $\boldsymbol{\Delta}_i^t = \sum_{l_1=i} \boldsymbol{\lambda}_l^{t-1} - \sum_{l_2=i} \boldsymbol{\lambda}_l^{t-1}$
5:     end for
6:     for all l do
7:         $\mathbf{g}_{l,\text{num}}^t = \frac{\alpha(\mathbf{x}_{l_1} - \mathbf{x}_{l_2}) + \boldsymbol{\Delta}_{l_1}^t - \boldsymbol{\Delta}_{l_2}^t}{\alpha}$
8:         $\mathbf{g}_{l,\text{cat}}^t = \frac{(1-\alpha)(\mathbf{x}_{l_1} - \mathbf{x}_{l_2}) + \boldsymbol{\Delta}_{l_1}^t - \boldsymbol{\Delta}_{l_2}^t}{1-\alpha}$
9:         $\boldsymbol{\lambda}_l^t = \text{proj}_{C_l}(\boldsymbol{\lambda}_l^{t-1} - \sigma(\mathbf{g}_l^t))$        ▷ $C_l = \{\boldsymbol{\lambda}_l : ||\boldsymbol{\lambda}||_2 \leq \gamma w_l\}$
10:    end for
11:    $\alpha_t = (1 + \sqrt{1 + 4\alpha_{t-1}^2})/2$        ▷ Accelerated AMA
12:    $\lambda_l^{t+1} = \lambda_l^t + \frac{\alpha_{t-1}}{\alpha_t}(\lambda_l^t - \lambda_l^{t-1})$        ▷ Accelerated AMA
13: end for
```

forms proximal gradient ascent to maximize the dual problem. We also point out a fast AMA proposed by Chi and Lange (2015) which is an accelerated variant based on Nesterov accelerated alternative.

4.4.2 Mixed-AMA stopping criterion

We can utilize the dual of the convex clustering problem 19. As the duality gap at the tth iteration provides an upper bound on how far we are from the optimal minimum of the objective function, we can interpret it as an indicator of optimality. First, we write the convex clustering objective in a more general form:

$$\begin{aligned} &\min f(\mathbf{u}) + g(\mathbf{v}) \\ &\text{subject to } \mathbf{Z}\mathbf{u} + \mathbf{B}\mathbf{c} = \mathbf{c}, \end{aligned} \tag{24}$$

which in our case implies that $\mathbf{Z} = [\mathbf{Z}_1 \dots \mathbf{Z}_\varepsilon]$, where ε is the number of non-zero edges and $\mathbf{Z}_l = [\mathbf{e}_{l_1}^t - \mathbf{e}_{l_2}^t] \otimes \mathbf{I}_p$ and $\mathbf{B} = -\mathbf{I}_{p\varepsilon}$ Detailed derivations are presented in Appendix B.

The dual objective can be expressed as:

$$D(\boldsymbol{\lambda}) = -f^*(\mathbf{Z}^t\boldsymbol{\lambda}) - g^*(\boldsymbol{\lambda}), \tag{25}$$

where

$$\begin{aligned} f^*(\mathbf{Z}^t\boldsymbol{\lambda}) =& \frac{1}{2}||\mathbf{Z}^t\boldsymbol{\lambda}^{num}||_2^2 + \langle \mathbf{Z}^t\boldsymbol{\lambda}^{num}, \mathbf{x}^{num}\rangle + ||\mathbf{Z}^t\boldsymbol{\lambda}^{cat}||_2^2 + \langle \mathbf{Z}^t\boldsymbol{\lambda}^{cat}, \mathbf{x}^{cat}\rangle \\ &+ \mathbf{x}^{cat}\text{log}(\mathbf{Z}^t\boldsymbol{\lambda}^{cat} + \mathbf{x}^{cat}) + (\mathbf{1} - \mathbf{x}^{cat})\text{log}(\mathbf{1} - \mathbf{Z}^t\boldsymbol{\lambda}^{cat} - \mathbf{x}^{cat}) \end{aligned} \tag{26}$$

and

$$g^*(\boldsymbol{\lambda}) = \delta_{C_l}(\boldsymbol{\lambda}_l) \tag{27}$$

which is the convex indicator function of the set C_l, namely $\{\boldsymbol{\lambda}_l : ||\boldsymbol{\lambda}||_\dagger \leq \gamma w_l\}$. The convex indicator function assigns a value of zero when an observation is part of the set and infinity otherwise. As evident from the AMA algorithm in 1, the $\boldsymbol{\lambda}_l$ are projected onto the set C_l, ensuring that the conditions are always satisfied.

Thus, if the difference in the primal objective 19 and the dual 25 at iteration t is below a certain threshold τ, we terminate the algorithm.

4.4.3 Feature Selection

In high-dimensional datasets, it is common for only a subset of features to significantly contribute to the clustering process. Selecting these relevant features is crucial as redundant ones may adversely affect clustering accuracy. B. Wang et al. (2018) and M. Wang and Allen (2021) propose a novel method involving a modified group-lasso penalty in the convex clustering objective. This penalty encourages cluster centroids to converge towards a center specific to the loss function. The aim is to equalize the values of redundant features, minimizing their impact on clustering. Features diverging from this loss-centered value are identified as potentially relevant. Their approach demonstrates notable empirical improvements in clustering performance.

Consequently, we introduce an additional penalty term into the objective. Specifically, we incorporate a squared Euclidean norm (Ridge) penalty into the convex clustering objective, as illustrated in Equation 28. The underlying principle behind this penalty is to drive irrelevant features towards the average $\tilde{\mathbf{a}}$ of their respective feature set. This form of shrinkage is referred to as egalitarian ridge in the literature Diebold and Shin (2019). Importantly, since the Ridge penalty is convex, the overall optimization problem retains its convex nature.

$$\begin{aligned}\min_{\mathbf{A}\in\mathbb{R}^{n\times p}} \quad & \sum_{i=1}^{n} \mathcal{L}^{num}(\mathbf{x}_i^{num}, \mathbf{a}_i^{num}) + \mathcal{L}^{cat}(\mathbf{x}_i^{cat}, \mathbf{a}_i^{cat}) + \gamma \sum_{l\in\mathcal{E}} w_l ||\mathbf{v}_l||_q + \beta \sum_{j=1}^{p} ||\mathbf{a}_j - \tilde{\mathbf{a}}_j||_2^2, \\ & \text{subject to } \mathbf{a}_{l_1} - \mathbf{a}_{l_2} - \mathbf{v}_l = 0 \quad \forall l \in \mathcal{E}\end{aligned} \tag{28}$$

Since we added a penalty term, the optimization steps undergo slight modifications in the AMA scheme. However, note that this change only affects updates for $\mathbf{a}$ since it does not involve $\boldsymbol{\lambda}$ or $\mathbf{v}$. Detailed derivations are provided in Appendix . The updated equations for $\mathbf{a}$ are as follows:

$$\mathbf{a}_i^{num} = \frac{\alpha \mathbf{x}_i^{num} + \sum_{l_1=i} \boldsymbol{\lambda}_l^{num} - \sum_{l_2=i} \boldsymbol{\lambda}_l^{num} + \beta \tilde{\mathbf{a}}_i^{num}}{(\alpha+\beta)} \tag{29}$$

$$\mathbf{a}_i^{cat} = \frac{(1-\alpha)\mathbf{x}_i^{cat} + \sum_{l_1=i} \boldsymbol{\lambda}_l^{cat} - \sum_{l_2=i} \boldsymbol{\lambda}_l^{cat} + \beta \tilde{\mathbf{a}}_i^{cat}}{1-\alpha+\beta} \tag{30}$$

Furthermore, M. Wang and Allen (2021) incorporate weights for each feature in the penalty term, following a common approach in the literature (H. Wang and Leng (2008); Zou (2006)). They utilize weights ζ_j that are proportional to loss-specific centers, calculated as $\zeta_j = \frac{1}{||x_j - \tilde{x}||}$, where $\tilde{x}$ denotes the loss-specific center for feature j. However, we depart from this approach as our objective is to implement a more rigorous feature selection strategy.

In our feature selection algorithm, similar to M. Wang and Allen (2021), we employ an adaptive scheme for feature selection. Initially, we obtain an estimate with no feature weights, and then gradually adjust the weights to diminish the influence of less important features while increasing those of more important ones, a strategy alike of other adaptive methodologies such as B. Wang et al. (2018) and Zou (2006). Since the Ridge penalty ensures that none of the features are exactly zero but rather close to zero, we can not discard features that have shrunken exactly to their respective loss-specific centers. Therefore, we select features based on the magnitude of the differences with respect to their loss-specific centers.

In general, selecting the number of features in this step requires careful consideration and is typically case-specific. In our simulations, we find that typically choosing between five and fifteen numerical and categorical features works well. However, if one has domain knowledge suggesting a different number of relevant features, the amount of features can be chosen as such. M. Wang and Allen (2021) scale the importance of these features in subsequent features based on this initial estimate. However, features chosen in the initial iteration may not necessarily capture the signal of interest, especially in high-dimensional data where many patterns can occur by chance. Thus, to minimize this potential bias, we also employ a feature selection framework to further refine the feature selection process.

Here, we utilize the feature selection framework for clustering introduced by Witten and Tibshirani (2010) propose a framework that selects features by maximizing the between-cluster sum of squares, incorporating a lasso-type penalty to control sparsity via a hyperparameter. Their framework is outlined as follows:

$$\begin{aligned} &\max_{\mathbf{w};\Theta \in D} \quad \left\{ \sum_{j=1}^{p} \zeta_j f_j\left(\mathbf{x}_j, \Theta\right) \right\} \\ &\text{subject to} \quad ||\boldsymbol{\zeta}||^2 \leq 1, \quad ||\boldsymbol{\zeta}||_1 \leq s, \quad \zeta_j \geq 0 \quad \forall j, \end{aligned} \tag{31}$$

where f_j is a function that only involves feature j. Within the framework, we seek to maximize the total influence of each function, denoted as f_j, which represents the contribution of feature j alone. A larger influence for f_j translates to a higher weight, ζ_j, assigned to that feature. Here, the lasso penalty within the framework ensures that the sum of all weights remains below a certain threshold s, which encourages sparsity by driving some weights to zero. Conversely, the Ridge penalty generally allows for multiple non-zero weights. Thus, the importance of feature j is encapsulated within f_j, which raises the need for a metric that gauges whether f_j effectively indicates the relevance of feature j in the clustering. Witten and Tibshirani (2010) assess the importance of each feature j based on its ability to drive a significant difference between the between-cluster and within-cluster distances. Note that this requires that the full clusters should be known. The between-cluster distance is defined as follows:

$$f_j\left(\mathbf{x}_j, \Theta\right) = \left(\frac{1}{n} \sum_{i=1}^{n} \sum_{i'=1}^{n} d_{i,i',j} - \sum_{k=1}^{K} \frac{1}{n_k} \sum_{i,i' \in C_k} d_{i,i',j} \right), \tag{32}$$

where $d_{i,i',j}$ denotes the dissimilarity between observation i and observation i' along feature j. Note that we use the original data matrix $\mathbf{X}^{n\times p}$ and not the estimated centroids $\hat{\mathbf{A}}^{n\times p}$.

The first part in this equation represents the average distance between all observations for feature j, while the second part denotes the average distance among observations within their respective clusters. Intuitively, important features in clustering should exhibit a larger between-cluster distance compared to less important features, which results in a higher weight assignment. We also note that all variables have been standardized before calculating the overall average distance. This standardization, detailed in Section 4.3, ensures that all features contribute equally to the first term in Equation 32, regardless of their original units. Consequently, the resulting weights are primarily driven by the features that minimize the within-cluster distance. However, one might note that that features with minimal variance might inaccurately get a high weight due to their tendency to minimize within-cluster distance across any given clustering $C_1, \ldots, C_k$. Nevertheless, such features are less likely to be selected, as they are typically shrunk faster towards the loss-specific center by the convex clustering method than features that are crucial for the clustering.

The proposition in Witten and Tibshirani (2010) states that the solution to this convex problem is $\lambda = \frac{S(\mathbf{y}_+, c)}{||S(\mathbf{y}_+, c)||_2}$, where $\mathbf{y}_+$ denotes the positive part of $\mathbf{y}$. Here, $\mathbf{a}$ is a vector which contains the values of $f_j(\mathbf{x}_j, \Theta)$ for every feature j. Moreover, the soft-thresholding operator S is defined as $S(\mathbf{x}, c) = \text{sign}(\mathbf{x})(|\mathbf{x}| - c+)_+$. If $||\boldsymbol{\zeta}||_1 \leq s$, then $c = 0$; otherwise, the value for $c > 0$ is chosen such that $||\lambda||_1 = s$.

Moreover, our approach to feature selection adapts to handle the specific characteristics of categorical data. Since these features are binary-coded, the average distance between them tends to be much smaller compared to numerical features. This is because many features will have zero values for most observations. To avoid this bias, we estimate weights for categorical and numerical variables separately. Furthermore, for categorical features, we capture their overall contribution by summing the distances of all their respective categories.

Our final feature selection approach is comprised of two distinct steps. Initially, we identify potentially relevant features by examining the deviations between the estimated feature $\hat{\mathbf{a}}_j$ and the loss-specific center. Then, given cluster assignments $C_1, \ldots, C_k$, we can calculate the feature importance f_j of those chosen features. However, Witten and Tibshirani (2010) obtain these clusters $C_1, \ldots, C_k$ by alternating K-means with the feature selection approach, until the features weights converge. We typically do not have enough fusions between iterations to construct the full clusters. To address this, we exploit the local structure discovered during the clustering process.

For each data point, we find its closest neighbors based on the estimated cluster center $\hat{\mathbf{A}}_t$ at the current stage of the clustering algorithm. We deliberately limit the number of neighbors to a small number to exploit local data density and locate features that influence cluster similarity. Therefore, rather than directly constructing K clusters using methods like K-means, we form many small clusters. This approach intuitively aligns with the capability of convex clustering to accommodate non-convex cluster structures. While the clustering method iteratively merges nearby points into larger clusters, our approach identifies features that drive the similarity within these local regions. This mirrors the iterative merging process of convex clustering.

Summarizing, the features that drive feature cluster separation within $\hat{A}$, should also provide cluster separation within the original data matrix $\hat{X}$. Otherwise, our algorithm will continue to run since the features will not stabilize. The detailed steps of our feature selection approach are presented in Algorithm 3.

Algorithm 2 Feature selection algorithm

```
    Input: Data matrix $\mathbf{X}^{n\times p}$; Centroid estimation $\mathbf{A}^{n\times p}$; Amount of features $r$ ; nearest neighbors $K$;$s$
    Output: Feature weights $\boldsymbol{\zeta}$
 1: Let $\mathbf{h}$ be the vector that contains the deviations from each feature with their loss-specific
 2: for j=1,...,p do
 3:     Calculate the sum of deviation from $a_j$ with the loss-specific center $\tilde{a}_j$, $h_j = \sum_{i=1}^{n} a_{ij} - \bar{x}_j$
 4: end for
 5: Select the features corresponding to the $r$ highest values among $\mathbf{h}$
 6: Construct a reduced centroid $\hat{\mathbf{A}}^{n\times r}$ with the $r$ selected features
 7: Compute a dissimilarity matrix $\mathbf{D}^{n\times n}$ using the reduced centroid $\hat{\mathbf{A}}^{n\times r}$
 8: for i=1,...,n do
 9:     $C_i = \{K\text{-NN of observation } i \text{ in } \mathbf{D}\}$
10: end for
11: Calculate the between-cluster distance $f_j$ for $j = 1,\dots,p$ defined in Equation 32
12: Solve for $\boldsymbol{\zeta}$ in Equation 31
13: Return the feature weights $\boldsymbol{\zeta}$
```

4.4.4 Cluster Fusions

Another important method to speed up computations is by fusing clusters. By fusing clusters, we reduce the computations since we replace two observations in the dataset with one new observation. Hocking et al. (2011) and Touw et al. (2023) implement cluster fusions, which is something we advocate for as well. Consider two clusters, C_1 and C_2. If we find that $|\mathbf{a}_{C_1} - \mathbf{a}_{C_2}| < \varepsilon$, we apply the following merging scheme.

$$\mathbf{a}_C = \frac{|C_1|\bar{\mathbf{a}}_{C_1} + |C_2|\bar{\mathbf{a}}_{C_2}}{|C_1| + |C_2|} \tag{33}$$

This new observation replaces C_1 and C_2 in the original dataframe $\mathbf{X}$. Intuitively, merging clusters is logical because, in the final convex clustering solution, we anticipate that the centers of all observations will converge for a sufficiently large γ. Essentially, we are preemptively merging some clusters that we have identified as sufficiently close.

Next, we update the set of edges $\mathcal{E}$. One posibility to determine the weights is as follows. All observations that previously had weights associated with or now form a new edge. We set the new weight as the sum of all positive weights between clusters C_1 and C_2. Alternatively, we could use the mean weight, but we find that this does not fundamentally alter the outcome since can still adjust the total weight through γ. The key point is that C_1 and C_2 are now connected. Another option is to recalculate the weights; however, this approach does not account for the local density of the data points. While this might not affect the shapes in convex clustering, for datasets like the half moons dataset, where density is more significant than distance, it is better practice to use the original edges.

However, we slightly modify this intuitive approach to determine the weights. An example is shown in Figure 1. Assume the first two rows are fused. We then sum these two rows and select the K-highest values, using the same K as in the K-NN scheme, which is three in this case. Next, we aggregate the columns since the columns at indices zero and one represent these fused observations. Naturally, the dissimilarity between these fused observations is zero. In this example, we observe that the observation at index three is among the nearest neighbors. This result indicates that while the first observation is not originally close to the third observation, the second observation is. Thus, this weighting scheme effectively utilizes the local density within the data.

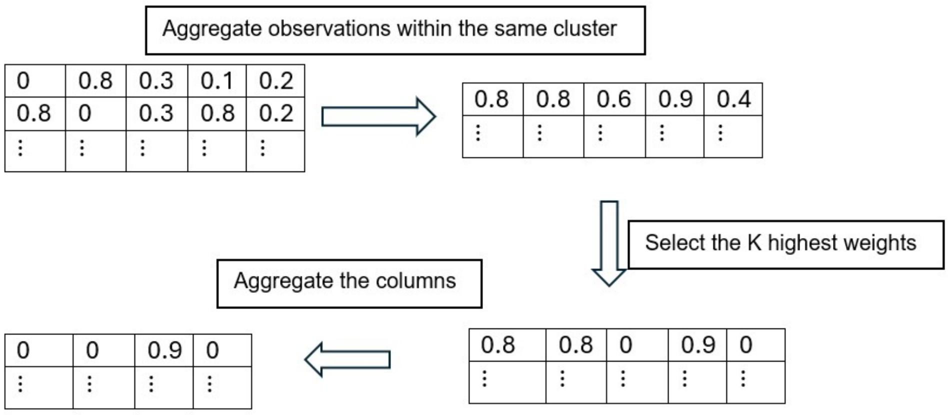


Figure 1: Weighting scheme for cluster fusion example

Moreover, we can control the number of cluster fusions possible in a single iteration. There are two approaches: either allow centroids to be fused multiple times or restrict each centroid to a single fusion. Both approaches yield similar outcomes, with the main difference being the order in which observations are fused. Additionally, we find that by allowing multiple fusions we may increase computational speed.

Lastly, we point a small inconsistency in our merging process shown in Equation 33. While averaging is appropriate for numerical data, it is not suitable for binary variables. Therefore, our approach sets all binary variables to 0 if they are less than 0.5, and to 1 otherwise. This method not only enhances interpretability but also reduces the impact of noisy features.

4.4.5 Full algorithm

With all components in hand, we can now finalize the algorithm. We have adapted the AMA-based convex clustering algorithm by Chi and Lange (2015) to accommodate both numerical and categorical features. Additionally, we have integrated a penalty term into the objective, inspired by the work of B. Wang et al. (2018) and M. Wang and Allen (2021), for feature selection. Furthermore, we've introduced a framework proposed by Witten and Tibshirani (2010) to complement the penalty feature selection approach, thereby enhancing the assessment of feature importance. The full algorithm is shown in Algorithm R.

Our algorithm begins by solving the convex clustering objective using the complete set of features. If certain observations are sufficiently close, we merge the clusters as outlined in Section 4.4.6 .Subsequently, once the primal and dual convex clustering objective are sufficiently close, we apply our feature selection algorithm to choose the most relevant features. Since, in practice, there is no assurance that these selected features are indeed the most relevant, we suggest another iteration of the convex clustering method. We re-solve the convex clustering method using only the selected features and repeat the feature selection process. This iterative cycle continues until the selected features stabilize across iterations. Finally, based on the output of the final cluster centroids A, we can allocate observations to clusters.

Algorithm 3 MAMACCAFS with Ridge penalty

```
Input: Data matrix $\mathbf{X}^{n\times p}$; neighbors $K$ in $K$-NN algorithm; tuning parameters $\phi$, $\gamma$, $\sigma$, $s$
Output: Cluster centroids $\mathbf{A}^{n\times p}$; cluster assignments; feature weights $\boldsymbol{\zeta}$
Functions: We use the followings functions for this algorithm
  • calculateWeights($\mathbf{X}$,K,$\phi$): returns the weights $\mathbf{w}$ as described in Section 4.3
  • performAMAIteration($\mathbf{X}$,$\boldsymbol{\lambda}$,$\sigma$): returns centroids $\mathbf{A}$ and dual variables $\boldsymbol{\lambda}$ by executing
    a single iteration of the AMA algorithm as described in Section 4.4.1
  • calculatePrimalObjective($\mathbf{X}$,$\mathbf{A}$,$\mathbf{w}$,$\gamma$): return the value of the primary objective of the
    convex clustering objective as shown in Equation 7
  • calculateDualObjective($\mathbf{X}$,$\boldsymbol{\lambda}$): return the value of the dualy objective of the convex
    clustering objective as shown in Equation 25
  • calculateFeatureWeights($\mathbf{X}$,$s$): returns the feature weights $\boldsymbol{\zeta}$ using the feature selection
    algorithm in Section 4.4.3
  • mergeObservations($\mathbf{A}^t$,threshold): checks whether cluster centroids are sufficiently
    close in order to perform a cluster fusion as described in Secion 4.4.6
Initialize: $\boldsymbol{\zeta}^0 \leftarrow [\zeta_1 \dots \zeta_p]$ where $\zeta_i = \frac{1}{\sqrt{p}}$ , $\boldsymbol{\lambda} \leftarrow [0 \dots 0]$, $t \leftarrow 0$
1:  $\mathbf{w}^t \leftarrow$ calculateWeights($\mathbf{X}$,K,$\phi$),
2:  while $||\boldsymbol{\zeta}^t - \boldsymbol{\zeta}^{t-1}|| < \varepsilon$ do
3:      while primal - dual $> \tau$ do
4:          $\mathbf{A}^t$,$\boldsymbol{\lambda}^t \leftarrow$ performAMAIteration($\mathbf{X}$,$\boldsymbol{\lambda}^{t-1}$,$\sigma$)
5:          mergeObservations($\mathbf{A}^t$,threshold)
6:          primal $\leftarrow$ calculatePrimalObjective($\mathbf{X}$,$\mathbf{A}^t$,$\mathbf{w}^t$,$\gamma$)
7:          dual $\leftarrow$ calculateDualObjective($\mathbf{X}$,$\boldsymbol{\lambda}^t$)
8:          $t \leftarrow t + 1$
9:      end while
10:     $\boldsymbol{\zeta}^t \leftarrow$ calculateFeatureWeights($\mathbf{A}^t$,$s$)
11: end while
12: Return $\mathbf{A}^t$, $\boldsymbol{\zeta}^t$
```

4.4.6 Cluster assignments

In order to obtain a clustering, we may look at the difference vectors $\mathbf{v}_l$. Chi and Lange (2015) merge two observations if the difference vector $\mathbf{v}_l = 0$ and then use a breadth-first search algorithm to locate the connected nodes. The update for $\mathbf{v}_l$ in AMA is given in Equation 23. However, our approach differs from Chi and Lange (2015) since we incorporate cluster fusions in the algorithm. As such, when the known amount of K clusters is reached, we terminate the algorithm.

Moreover, one aspect of cluster fusions is the possibility of ending up with multiple unconnected clusters. This naturally occurs when the differences between cluster centroids are not sufficiently small, indicating that not all observations are connected. Consequently, they have never been instructed to merge in the first place. In this thesis, we assume the existence of distinct clusters, with a known quantity in advance, which may result in unconnected graphs. Touw et al. (2023) propose a method to connect the graph. However, their work specifically focuses on (hierarchical) convex clustering, which necessitates the connection of all data points within the final hierarchy. Given our assumption of a predetermined number of clusters, encountering more unconnected clusters than the predetermined clusters signals two potential issues: either the selected features inadequately capture the underlying cluster structure, or the clustering parameters need fine-tuning.

4.4.7 Tuning of the hyperparameters

This convex clustering framework involves tuning of various hyperparameters. A broad guideline can be derived by comparing values found in various literature. Below, we outline some general recommendations based on our experimentation.

Tuning K in the K-NN scheme and ϕ . Section 4.3 discusses the construction of the weights $\mathbf{w}$. In datasets containing predominantly relevant features, this weighting scheme typically yields distances that accurately capture the proximity between different objects. Consequently, nearby observations tend to be pulled closer to each other. However, in datasets with noise, this weighting scheme may not perform optimally, leading to incorrect observations being drawn towards each other. This is expected as when the amount of relative features is much smaller than the amount of noisy features, the noisy features may impact the K-NN weighting scheme. Another perspective is that we notice consistency, especially with low K values, in the K-nearest neighbors (K-NN) approach utilizing the weights ($\mathbf{w}$) when calculating K-NN with the estimated output matrix $\hat{\mathbf{A}}$. This is expected as we encourage these observations to converge. However, we observe that this issue can be alleviated by opting for a relatively high number of neighbors, which increases the likelihood of true centroids converging towards each other. Thus, we have to adaptively change the amount of neighbors K in the algorithm. In the first stage, when the dataset contains a lot of noise, the amount of neighbors should be high to increase the chances of observations within the same cluster to converge to another. Then, if the feature selection scheme has selected certain features, the amount of neighbors should be relatively low to exploit locality of the data.

Furthermore, we find that the choice of ϕ does not impact the clustering too much. This is due to the fact that γ is mainly responsible for the cluster centroid penalty. Throughout our research, we used a value of $\phi = 2$, which means that weights decrease quadratically with increasing distance.

Tuning α. In the convex clustering objective 19, we introduced weights to balance the contribution of the numerical and categorical loss function. When $\alpha = 1$, all weights are directed towards numerical features, while for $\alpha = 0$, all weights prioritize categorical features. To achieve a balanced contribution, we iterate the convex clustering method for a range of different α until the numerical and categorical objectives are approximately comparable.

Tuning γ. The parameter γ imposes a penalty on differences among the cluster centroids $\mathbf{A}$. Consequently, a low γ results in minimal fusion among the cluster centers, while a high γ leads to a significant fusions between the centroids. We find that the optimal value of γ varies depending on the number of observations and features. Therefore, a more general approach, as suggested by Hocking et al. (2011), is to utilize a clusterpath. This method illustrates how observations merge as γ increases. In general, γ can be increased until we obtain exactly the required number of clusters.

Tuning σ. The tuning parameter σ used in the AMA for convex clustering which essentially controls for how fast the cluster centers shrink towards one another. An appropriately selected σ can substantially affect computation time. Typically, we advise choosing a value within the range of [0.01, 0.2]. Furthermore, if the number of neighbors (K) in the K-NN scheme is low, we recommend to increase σ, whereas for a higher number of neighbors, we suggest opting for a lower σ. Thus, this parameter does not impact the clustering quality, but rather computational speed.

Tuning β. This parameter penalizes differences of the estimated feature centroids $\hat{\mathbf{a}}_j$ with their loss-specific center $\tilde{\mathbf{a}}_j$. We find that this parameter is crucial in this framework as it diminishes the impact of noisy variables by pulling them closer to the loss-specific centers. The value β depends on the size of the dataset. In general, we recommend a low value of β, typically between 1 and 5, which encourages slight shrinkage towards the loss specific center. The main idea is that redundant features shrink faster to their respective loss-specific center compared to non-relevant features.

Tuning s. This parameter controls the number of features selected in the feature selection scheme. In this thesis, we used a value of $s = 2$ because it generally resulted in selecting most of the relevant features. Setting s to a value less than 2 usually retains only a few features, while values greater than 2 can lead to instability in feature selection. Higher values might include relevant features but also tend to incorporate noisy features, which can negatively impact the convergence of the feature selection process.

4.5 Benchmark models

We evaluate our convex clustering algorithm designed for categorical data, employing the AMA algorithm, against various established mixed data clustering methods. One such method is the K-prototypes method introduced by Huang (1998), which operates on a similar principle to the K-means algorithm. A downside of this this algorithm typically does not converge to a global solution.

Additionally, we compare our approach to the KAy-means for MIxed LArge data (KAMILA) method proposed by Foss, Markatou, Ray and Heching (2016). KAMILA addresses high-dimensional data while ensuring fair consideration of both continuous and categorical variables in the clustering process. Chavent, Lacaille, Mourer and Olteanu (2020) also propose a method for high-dimensional data. Their sparse mixed clustering method is also based on the framework introduced by Witten and Tibshirani (2010).

Lastly, we utilize Hierarchical Agglomerative Clustering (HAC). HAC iteratively merges observations based on different criteria (linkages). In this thesis, we will use different linkages and select the one with the best results.

4.6 Evaluation metric

In order to compare the performance of the clustering methods, we employ the Classification Error Rate (CER). Consider two partitions each of length n, denoted by P and Q. Partition P represents the true class labels, while partition Q represents a partition obtained through clustering. Assuming we have C clusters, both $P = \{P_1, \ldots, P_C\}$ and $Q = \{Q_1, \ldots, Q_C\}$ partition the observations in C subsets. Let $1_{P(i,i')}$ be an indicator whether partition P places observations i and i' in the same group, and the same applies for $1_{Q(i,i')}$. Then, the CER is defined as

$$\text{CER} = \sum_{i>i'} \frac{|1_{P(i,i')} - 1_{Q(i,i')}|}{\binom{n}{2}} \tag{34}$$

Note that the CER is zero if the partitions P and Q are the same.

In addition, we use the Adjusted Rand Index (ARI) to compare the results of our clustering. The standard Rand Index (RI) can be interpreted as a measure of accuracy. In this context, true positives (TP) refer to pairs of elements that are in the same subset in both P and Q, and true negatives (TN) refer to pairs of elements that are in different subsets in both P and Q. False positives (FP) are pairs of elements that are in different subsets in P but the same subset in Q, and false negatives (FN) are pairs of elements that are in the same subset in P but different subsets in Q. The RI is then calculated as follows:

$$\text{RI} = \frac{TP + TN}{TP + TN + FP + FN} \tag{35}$$

The Adjusted Rand Index (ARI) is a corrected-for-chance version of the Rand Index (Rand, 1971). This correction establishes a baseline by considering the expected similarity of all pairwise comparisons between clusterings under a random model. We can form a contingency matrix as follows. Let $n_{i,j} = |P_i \cap Q_j|$ be the intersection of subset i of P and subset j of Q:

$$\begin{array}{c|cccc|c} & Y_1 & Y_2 & \cdots & Y_s & \\ \hline X_1 & n_{11} & n_{12} & \cdots & n_{1s} & a_1 \\ X_2 & n_{21} & n_{22} & \cdots & n_{2s} & a_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ X_r & n_{r1} & n_{r2} & \cdots & n_{rs} & a_r \\ \hline & b_1 & b_2 & \cdots & b_s & \end{array} \tag{36}$$

The Adjusted Rand Index is then formulated as:

$$\text{ARI} = \frac{\left(\sum_{ij} \binom{n_{ij}}{2} - \frac{\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}}{\binom{n}{2}}\right)}{\left(\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \frac{\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}}{\binom{n}{2}}\right)} \tag{37}$$

Note that while the standard RI is defined between 0 and 1, the ARI can be negative if the index is less than the expected index.

4.7 Software

We implemented our method using Python version 3.11. For the benchmark models, we used R for the following packages: *clustMixType* for K-prototypes (Szepannek, 2018), *hclust* for HAC (R Core Team, 2023),*kamila* for KAMILA (Foss et al., 2016), and *vimpclust* for Sparse k-means for mixed data via group-sparse clustering (Chavent et al., 2020).

5 Simulation study

First, we evaluate the performance of our convex clustering method on simulated data. In our simulated data, we generate both numerical and categorical data with noise. The goal is to asses whether our feature selection scheme effectively chooses the correct features and clusters the correct groups. Moreover, we compare the performance of the clustering with other popular mixed clustering models.

5.1 Simulation

Spherical cluster without noise. We assume a number of $C = 3$ classes. The initial $q = 20$ features consist of $m = 10$ numerical variables and $q - m = 10$ categorical variables, and are the main important features that differentiate between the clusters. The numerical data for $j \leq m$ is derived from a multivariate normal distribution:

$$\begin{bmatrix} x_{j1} \\ x_{j2} \\ x_{j3} \end{bmatrix} \sim \mathrm{N}\left(\begin{bmatrix} \mu_{j1} \\ \mu_{j2} \\ \mu_{j3} \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \right) \quad j = 1, \ldots, m \tag{38}$$

Specifically, $X_{ijc} \sim N(\mu_{jc}, 1)$ denotes the value for observation i for feature j belonging to cluster c, where $\mu_{j1} = 1$, $\mu_{j2} = 2$, and $\mu_{j3} = 3$. Furthermore, we generate categorical data from a multinomial distribution with $K = C$ categories for $m < j \leq q$, where $X_{ijc} \sim \text{Mult}_3(N_i, p_{1c}, p_{2c}, p_{3c})$:

$$X_{ijc} = \begin{bmatrix} x_{j1} \\ x_{j2} \\ x_{j3} \end{bmatrix} \sim \text{Mult}\left(N_i, \begin{bmatrix} \mathrm{p}_{j1c} \\ \mathrm{p}_{j2c} \\ \mathrm{p}_{j3c} \end{bmatrix} \right) \quad j = m+1, \ldots, q \tag{39}$$

We control the strength of the signal by adjusting the probability of the respective important cluster category for each feature. Assuming category k possesses the highest proportion in cluster $C = k$, all clusters feature a distinct majority class per feature, assigned a probability of 0.75. The probabilities of the remaining three classes are set to be equal and sum up to 0.25.

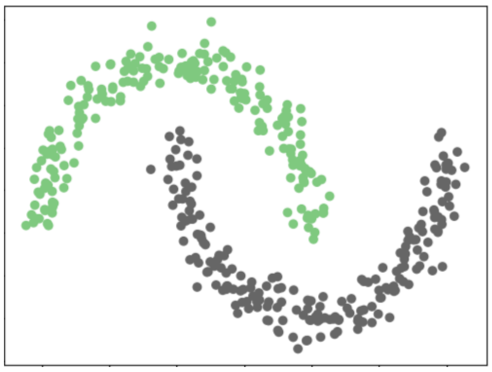
Spherical cluster with noise. In this scenario, we maintain the same setting as the spherical cluster without noise. However, we introduce additional variables that are irrelevant to the clustering task. An equal number of numerical and categorical noise variables are generated. For $j > q$, the numerical variables are generated from a normal distribution with mean 0 and variance 2, denoted as $X_{ij} \sim \mathrm{N}(0, 2)$, while the categorical variables follow a categorical distribution with three categories, each with equal probability $X_{ij} \sim \text{Cat}_3(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$. We repeat this process for various levels of feature amounts p, namely $p = 20, 100, 500$ and 1000.

Table 1: Simulation experiments for spherical shapes. Bold values denote the best performing method. The simulation has been repeated 20 times.

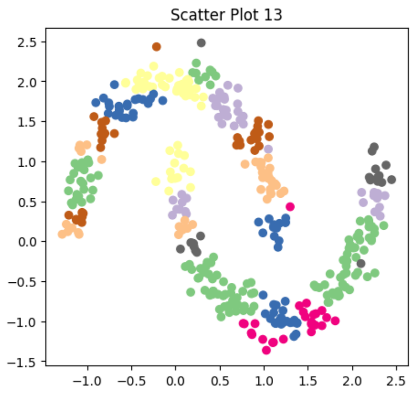
		p=20		p=100		p=500		p=1000	
		CER	ARI	CER	ARI	CER	ARI	CER	ARI
MCC	Mean	0.056	0.940	0.09	0.837	**0.149**	**0.714**	**0.103**	**0.786**
	SE	0.104	0.066	0.137	0.205	0.148	0.232	0.121	0.168
K-prototypes	Mean	0.033	0.915	0.368	0.317	0.587	0.000	0.594	-0.001
	SE	0.034	0.05	0.368	0.036	0.009	0.004	0.006	0.005
KAMILA	Mean	**0.003**	**0.990**	0.263	0.474	0.666	0.000	0.666	0.000
	SE	0.002	0.006	0.027	0.038	0.000	0.000	0.000	0.000
Sparse k-means	Mean	0.004	0.987	**0.070**	**0.840**	0.590	-0.004	0.588	-0.007
	SE	0.002	0.006	0.031	0.053	0.006	0.005	0.006	0.003
HAC	Mean	0.023	0.933	0.100	0.723	0.565	0.011	0.573	0.005
	SE	0.004	0.011	0.009	0.025	0.008	0.006	0.010	0.006

The results of the simulation are presented in Table 1. When there are no noise features ($p = 20$), the K-prototypes algorithm outperforms MCC because it makes use of the entire set feature set. However, as noise is introduced, K-prototypes' performance becomes worse as it tries to find patterns with all features. Conversely, MCC maintains consistent performance even with significant noise, due to its feature selection capability, which discards unimportant features. It is important to note that we did not tune the methods and relied on the default settings. Additionally, the errors primarily stem from our data generation process, as the algorithm generally selects features from the important fist 20 features. The variability introduced in our DGP could account for some inconsistencies. Nevertheless, this simulation illustrates that our MCC algorithm effectively performs feature selection while maintaining good clustering quality.

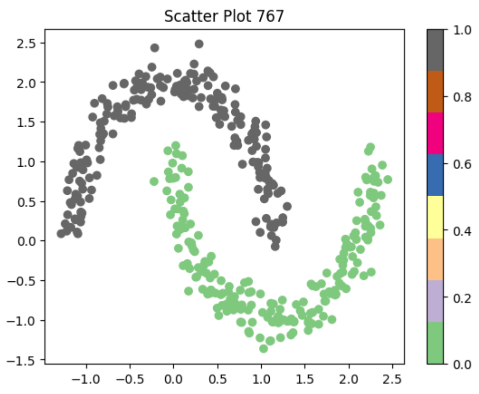
Non-spherical cluster with small noise. Another strength of convex clustering is that it is capable of handling non-convex cluster shape. We generate half moon data with $n = 400$ and 8% noise. The simulated data of two interlocking half-moons is one of the most popular test examples in clustering. Firstly, we generate two numerical features which represent the x and y coordinates of the points. We show the performance when there are only numerical variables.



(a) Simulated Half Moons data with $n = 400$ and 8% noise



(b) Merging process



(c) Final clustering result

Figure 2: Half Moon data clustering with convex clustering. Note that in plot (b) similar colours indicate the density of the grouped objects.

Figure 2 shows how convex clustering groups the Half Moons data. It starts by shrinking centroids of observations that are similar to one another. These centroids are then iteratively merged until we obtain the correct half moon clustering result. Note that since we cluster density data, the amount of selected neighbors in our weighting scheme is crucial. Selecting a relatively low amount of neighbors compared to the total number of observation ensures that the correct observations obtain a weight w_l. In this instance, we selected 5 neighbors, guaranteeing that all observations in the upper and lower half moons are exclusively connected to each other. Consequently, the task is to increase γ until reaching the desired number of clusters. Typically, datasets contain noise, so we do not always add a weight between observations within the same cluster. However, the same outcome can be attained by choosing a higher number of neighbors, albeit with slightly more careful tuning of γ.

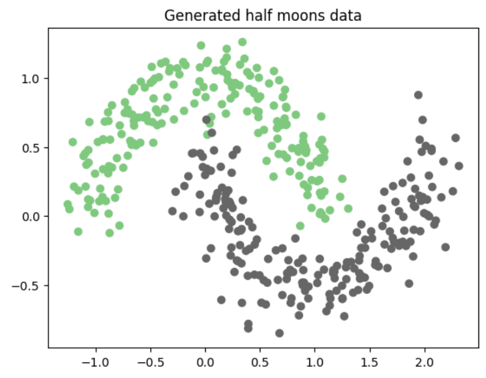
Non-spherical cluster with moderate noise. We now repeat the half-moon data experiment but with roughly double the noise. We generate 200 observations with 15% noise. At this increased noise level, we expect the convex clustering to perform slightly worse, as the weighting scheme assigns weights between observations in different clusters regardless of the amount of neighbors. We now introduce a single categorical variable, which consists of 2 categories. Each cluster has a majority category with a probability of 70%, while the minority category has a probability of 30%. Our objective is to assess how various methods can effectively combine the two data types.

The ARI scores are shown in Table 2. We find that the convex clustering algorithm is the only method that can retrieve the true clusters. As discussed in Section 4.2.1, our algorithm achieves this by smartly constructing the weights between the cluster centroids, allowing it to handle non-convex shapes. In contrast, all other methods produce clusters that are spherical in nature, which limits their effectiveness with non-convex cluster shapes. Moreover, while HAC is generally capable of identifying non-spherical shapes, it typically requires minimal noise or very high density to function properly. These conditions are not met with this dataset as we generated the data with noise and kept the number of observations low.

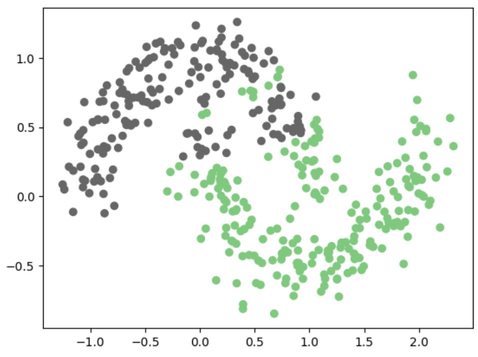
Table 2: Simulation experiments for two interlocking half-moon data. Bold value denotes the best performing method.

Cluster algorithm	Adjusted Rand Index
MCC	**0.917**
K-prototypes	0.279
KAMILA	0.301
sparse k-means	0.448
HAC	0.329

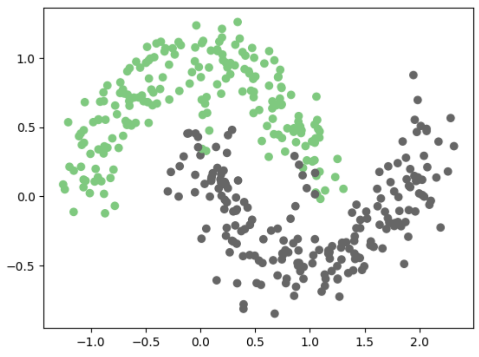
Moreover, the results of this simulation can also be shown visually through figures. Figure 3(a) illustrates that the data points in the inner components of the half moons are closer to each other compared to the previous simulation with 8% noise. When we apply convex clustering only to the numerical part of the data, the clustering accuracy at the inner border declines due to increased noise, resulting in more positive weights between observations of the two clusters. This is shown in Figure 3(b). Whereas, applying mixed convex clustering to the mixed data in Figure 3(c) notably improves accuracy, although it is not perfect due to noise in the categorical data. Overall, the method effectively captures the underlying structure of the data by utilizing the categorical data.



(a) Simulated Half Moons data with $n = 400$ and 8% noise



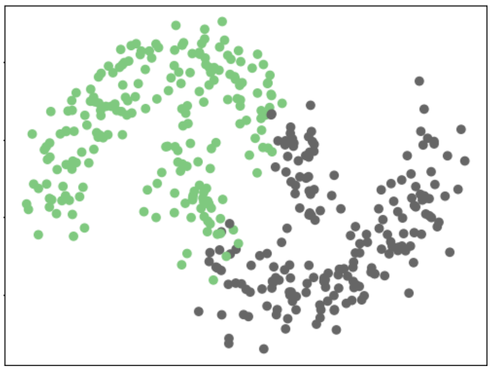
(b) Only numerical data



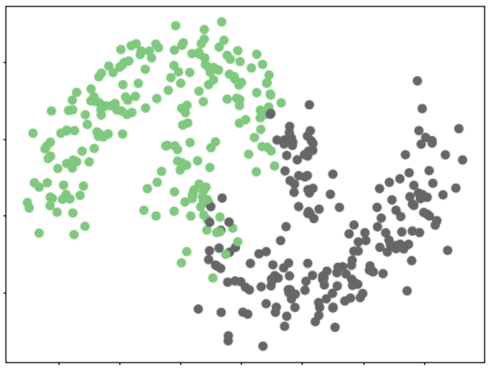
(c) Numerical and categorical

Figure 3: Half Moon and categorical data clustering with convex clustering

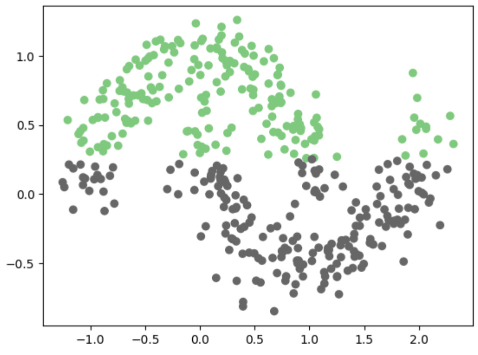
When we apply other mixed clustering methods to the same Half Moon dataset containing one categorical feature, they fail to identify the actual structure within the data. The clustering results are shown in Figure 4. Since these methods are based on the K-means algorithm, they tend to produce clusters that are spherical in nature. Moreover, Figure 4 shows that sparse K-means divides the half-space in two to determine the two clusters. Lastly, HAC results in a cluster shape that is spherical as well.



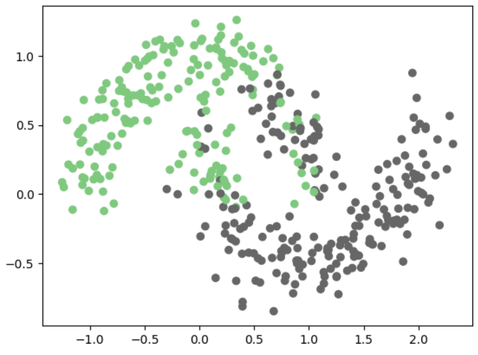
(a) K-Prototypes



(b) KAMILA



(c) Sparse K-means mixed data



(d) HAC

Figure 4: Clustering results of (a) K-prototypes, (b) KAMILA, (c) Sparse k-means and (d) HAC for Half Moon data with 15% noise and one categorical feature.

5.2 Sensitivity analysis

In our simulation experiments, we observed that certain hyperparameters significantly influence the convex clustering algorithm. The recommended default values for these hyperparameters are detailed in Section 4.4.7. The most critical parameters are γ and the number of neighbors in the K-NN weighting scheme. The γ parameter determines whether cluster centroids fuse, while the K-NN weighting scheme dictates the number of cluster centroids that are fused. Therefore, the algorithm should be tuned to ensure that γ is large enough to shrink the cluster centroids $\hat{A}$ towards one another. However, when we choose too many neighbors, we risk clustering all observations into one single cluster.

We present the ARI for the case of spherical clusters with noise. Specifically, when there are $p = 1000$ features, 980 of which are noisy, accounting for 98% of the features. We vary the values of K in the K-NN scheme and γ, and display the ARI values in Table 3.

Table 3: Sensitivity analysis for spherical clusters with noise. Bold values denote ARI scores higher than 0.7

K/ γ	0.01	0.20	0.40	0.60	0.80	1.00	1.20	1.40	1.60	1.80	2.00
1	0.0	0.04	0.03	0.04	0.04	0.07	0.04	0.05	0.05	0.06	0.07
2	0.0	0.1	0.16	0.17	0.17	0.17	0.16	0.16	0.19	0.19	0.19
3	0.0	0.06	0.03	0.09	0.06	0.09	0.1	0.14	0.16	0.11	0.1
4	0.0	0.48	0.66	0.51	0.52	0.47	0.52	0.6	0.45	0.49	0.55
5	0.0	0.61	0.62	**0.75**	0.62	0.56	0.43	0.39	0.49	0.49	0.62
6	0.0	-0.0	0.0	-0.0	-0.01	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
7	0.02	**0.88**	**0.85**	**0.88**	**0.9**	**0.9**	**0.9**	**0.9**	**0.9**	**0.9**	**0.9**
8	0.02	**0.9**	**0.9**	**0.88**	**0.88**	**0.84**	**0.85**	**0.88**	**0.88**	**0.93**	**0.88**
9	0.03	**0.81**	0.67	**0.81**	**0.83**	**0.78**	**0.72**	**0.9**	**0.72**	**0.72**	**0.76**
10	0.02	**0.9**	**0.9**	**0.88**	**0.98**	**0.87**	**0.87**	**0.87**	**0.87**	**0.82**	**0.9**
11	0.01	**0.73**	**0.88**	0.07	0.07	**0.81**	**0.79**	0.02	**0.73**	**0.73**	**0.73**
12	0.01	**0.88**	0.57	**0.95**	**0.81**	**0.83**	**0.85**	0.54	0.5	**0.81**	**0.81**
13	0.02	**0.83**	**0.95**	0.57	0.45	0.68	**0.75**	0.78	**0.81**	**0.77**	0.56
14	0.04	**0.85**	**0.86**	**0.86**	**0.73**	**0.72**	**0.9**	**0.9**	**0.85**	**0.9**	**0.81**
15	0.04	**0.9**	0.55	**0.81**	0.56	**0.83**	0.56	0.65	0.61	0.42	0.45
16	0.04	**0.88**	**0.83**	0.43	0.53	0.32	0.32	0.32	0.32	0.4	0.4
17	0.04	**0.93**	0.48	**0.85**	0.44	0.49	**0.72**	0.01	0.48	0.0	0.36
18	0.03	**0.71**	**0.9**	0.55	0.65	0.0	0.67	0.48	0.42	0.42	0.47
19	0.02	**0.95**	0.57	0.69	0.47	0.37	0.0	0.0	0.0	0.0	0.0
20	0.01	**0.81**	**0.72**	0.02	0.11	-0.0	0.0	0.04	0.0	0.06	0.0
21	0.02	0.49	-0.0	0.0	0.04	0.01	0.0	0.0	0.0	0.0	0.0
22	0.05	**0.93**	0.42	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
23	0.03	0.0	0.29	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
24	0.03	0.01	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
25	0.03	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
26	0.02	0.0	0.0	-0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
27	0.01	-0.0	0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
28	0.05	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
29	0.03	0.0	0.08	-0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Firstly, we note that the ARI never reaches one due to the variance introduced in our Data Generating Process (DGP). Typically, the algorithm successfully forms three main clusters that accurately group the three distinct sets of observations, although a few observations may end up in smaller, separate clusters. In practice, these smaller clusters could be reassigned to the nearest larger cluster. Additionally, the algorithm usually selects the subset of features that are most influential in driving cluster separation.

Low ARI values generally indicate one of two scenarios: either only a few cluster centroids have fused, resulting in a high number of clusters, or all cluster centroids have fused together into a single cluster. While we could have terminated the algorithm if all observations fused, we deliberately did not do this to demonstrate the variability in clustering performance.

Secondly, we find that when the number of neighbors K is relatively low, the ARI reflects poor clustering performance due to the formation of unconnected graphs among observations within the same group. For K values around 10% to 20% of the total observations, we observe relatively better and more consistent performance across all values of γ. Conversely, for larger values of K, good ARI scores are only obtained for small values of γ. Since each cluster contains 20 observations, it is ideal to select fewer than 20 neighbors to avoid fusing observations from different clusters. Additionally, to ensure a connected graph, the number of neighbors should not be too low, especially given that 98% of the features in our data are noisy, which can further impact the accuracy of the K-NN scheme.

These findings align with the empirical results of Chi (2015), who concluded that it is not necessary to find the exact K-NN and hypothesized that clustering quality would not decline if approximate nearest neighbors are used. Therefore, based on these findings, we recommend selecting K to be roughly between 10% and 20% of the total number of observations.

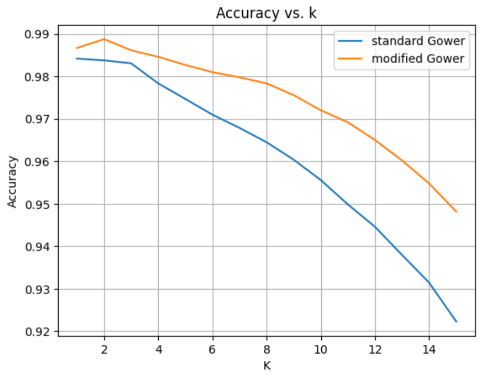
5.3 Weighting scheme analysis

In our next analysis, we compare the performance of the standard Gower distance with our modified Gower distance as described in Section 4.3. The accuracy of the distance measure in the K-NN weighting scheme is crucial as it determines which observations are fused together. Our modified Gower distance is scaled so that the expected distance for each feature equals one. We measure performance by calculating the average percentage of correctly chosen neighbors across different levels of noise.

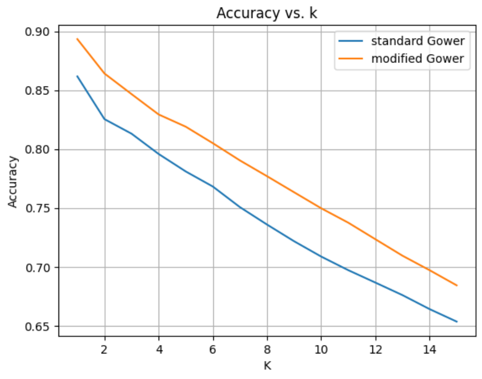
Figure 5 displays the accuracy of the K-NN scheme for the simulated spherical clusters, based on 20 different simulated datasets. The results show that the modified Gower distance consistently outperforms the standard Gower distance. However, the performance of both measures declines as the number of noisy features in the dataset increases. This decline is expected, as both dissimilarity measures aim to ensure equal variable contribution to the dissimilarity: the standard Gower distance does this by confining the range of dissimilarity for each feature, while the modified Gower distance sets the expected distance of each feature to one. Thus, both relevant and noisy features contribute equally, which results in the noisy feature overpowering the relevant features.

Moreover, in our previous sensitivity analysis, we observed that the consistency of the ARI drops when selecting around 15 neighbors, which is 25% of the total observations, in the K-NN scheme, particularly in the case with 98% noise. Based on Figure 5(d), this selection results in an accuracy of less than 50% for the K-NN scheme. Thus, we conjecture that the convex clustering algorithm achieves consistent and adequate performance when the average accuracy of the K-NN scheme is at least 50%.

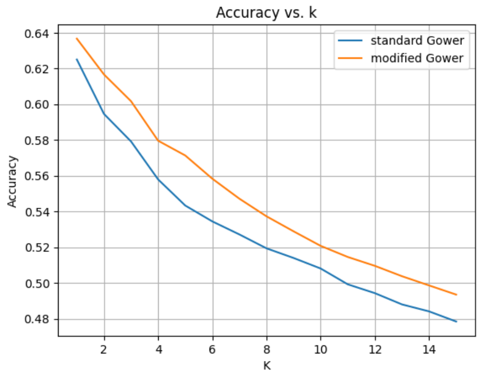
Summarizing, the simulation results indicate that the K-NN weighting scheme can be significantly affected by noise in the features, potentially negatively impacting clustering performance. Given that datasets may contain thousands or even millions of features, correctly determining the neighbors in the K-NN weighting scheme becomes a major challenge for the convex cluster-



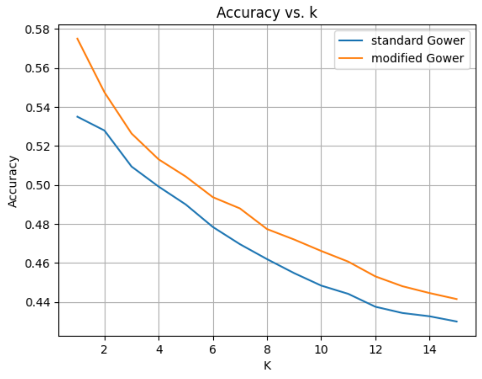
(a) 0% noise



(b) 80% noise



(c) 96% noise



(d) 98% noise

Figure 5: Accuracy of the K-NN scheme for selecting neighbors in the same cluster in different noise scenarios

ing framework. In summary, while the convex clustering framework can handle moderate noise, the presence of many noisy features can adversely affect clustering quality.

6 Results

In this section, we present the results of our proposed mixed convex clustering method, which combines the AMA algorithm by Chi and Lange (2015) and the feature selection framework by Witten and Tibshirani (2010), applied to a real gene expression dataset. The dataset includes samples from ALL and AML patients, and our objective is to accurately cluster these two groups.

Initially, we cluster a dataset comprising 38 observations: 27 ALL patients and 11 AML patients, across 7129 numerical and categorical features. This dataset is particularily challenging due to its high dimension and class imbalance. The results of our clustering are detailed in Table 4.

Table 4: Adjusted Rand Index for Different Cluster Algorithms

Cluster Algorithm	**Adjusted Rand Index**
MCC	**0.911**
K-prototypes	0.041
KAMILA	-
sparse k-means	0.041
HAC	0.011

Firstly, our convex clustering method performs significantly better than the benchmark models based on the ARI scores. Additionally, it is worth noting that the KAMILA algorithm failed to cluster this dataset due to its interpolation method detecting NA (missing) values, despite our confirmation that the dataset contains no missing values. All other methods also did not detect any missing values. Thus, our convex clustering method demonstrates capability in performing feature selection effectively in high-dimensional data.

One of the primary reasons the other methods perform poorly is their lack of feature selection capability, except for the sparse weighted k-means for mixed data. This capability is crucial for identifying the correct features that drive cluster separation. Given that the dataset contains 14252 features, these algorithms will attempt to cluster the observations using the full set of features, which is likely to degrade clustering performance. Additionally, all these methods are prone to getting stuck in local optima due to their non-convex nature. Therefore, the sparse weighted k-means for mixed data is the only real contender compared to our method for this dataset. However, due to its tendency to get stuck in local optima, its performance is comparable to the other non-convex clustering methods.

However, it is important to note that ARI scores for our method fluctuates depending on the number of neighbors chosen in the K-NN scheme. This variability stems from the fact that we did not fine-tune γ for each value of K. Instead, we used a fixed value of $\gamma = 100$ for all values of K. As a result, the clustering often produced multiple clusters rather than exactly two clusters. However, these clusters effectively separated the underlying groups—ALL and AML patients. Specifically, the convex clustering algorithm formed sub-clusters within each true cluster. Therefore, any errors primarily arise from clusters not merging due to γ being set too low, rather than incorrect pair assignments.

Consequently, we plot the values of the ARI against K in Figure 6. We observe that a high ARI score is achieved when selecting five or six neighbors, corresponding to 13% and 16% of the total observations, respectively. This finding aligns closely with the results of our previous simulation experiments, where consistent clustering results were obtained for K values between 10% and 20% of the dataset.

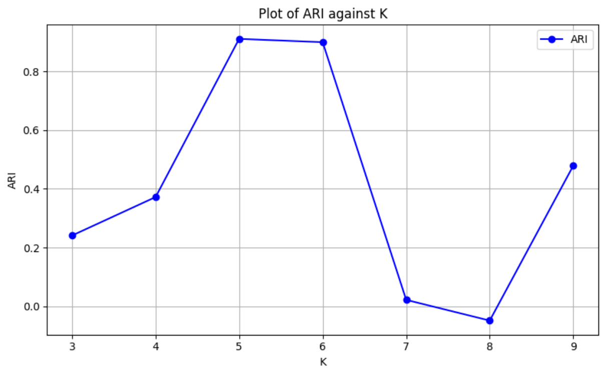


Figure 6: The ARIs obtained by our convex clustering method, for a range of values of neighbors K in the K-NN weighting scheme

Next, we apply our clustering method to an independent dataset. This dataset is less imbalanced compared to the training dataset, with 20 ALL patients and 14 AML patients. The results are summarized in Table 5. Once again, our method demonstrates superior performance compared to other methods. However, unlike the previous training dataset, we hesitate to conclude that our algorithm consistently selects the correct clusters. In this case, the lower ARI scores are a result of incorrect pairings rather than there being multiple subclusters within the true clusters. Figure 7 also suggests that the true clusters may not be reliably identified by our method.

Table 5: ARI scores for cancer gene expression dataset. Bold value denotes the best performing method.

Cluster algorithm	Adjusted Rand Index
MCC	**0.758**
K-prototypes	0.144
KAMILA	-
sparse k-means	0.144
HAC	0.144

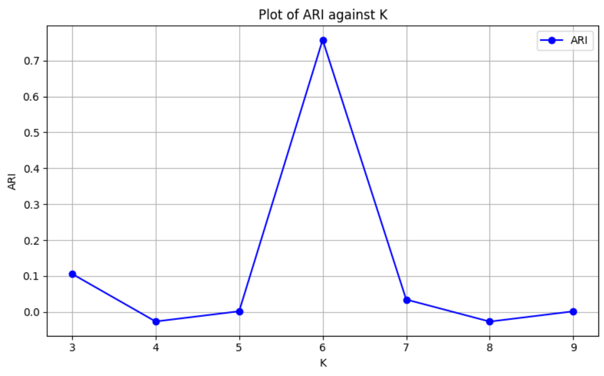
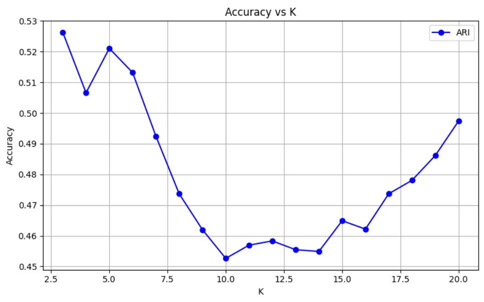


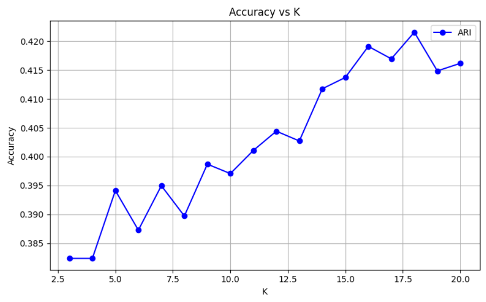
Figure 7: The ARIs obtained by our convex clustering method, for a range of values of neighbors K in the K-NN weighting scheme

We further investigate potential reasons for these outcomes. Our intuition suggests that the

K-NN weighting scheme may be selecting incorrect neighbors in the K-NN weighting scheme. Therefore, we conduct a similar experiment to assess the accuracy of the K-NN weighting scheme in correctly assigning positive weights between observations that belong to the same cluster. The results are presented in Figure 8.



(a) Training dataset



(b) Independent dataset

Figure 8: Accuracy of the K-NN scheme for selecting neighbors in the same cluster in different noise scenarios

Based on Figure 8, we note that the accuracy of the weighting scheme in the training dataset is higher than the independent dataset. Thus, we have identified a potential reason for the relatively poorer performance on the independent dataset compared to the training dataset. Specifically, we observed that the training dataset consistently achieves an average accuracy of 50% or higher in the weighting scheme, whereas the independent dataset falls below 50%. In our simulation experiments, we established that our convex clustering algorithm performs more reliably when the weighting scheme averages a 50% accuracy in neighbor selection. This consistency was not observed in the independent dataset, which may have contributed to the observed results.

Therefore, while our convex clustering algorithm can identify a global solution based on the provided features, the effectiveness of the clustering itself is closely tied to number of noisy features. While we have mitigated the instability of non-convex methods like K-means by incorporating convex clustering, we still have to deal with the variability introduced by the selection of features. In high-dimensional data, there are often multiple clusters that may achieve a lower objective loss than the true signal we seek to identify.

In conclusion, our convex clustering algorithm consistently outperforms other popular clustering methods across both simulation datasets and gene expression data. We find that it successfully identifies the true clusters within the population, provided that the weighting scheme is not overly influenced by noise.

7 Conclusion

In this thesis, we formulated an answer to the research question: "How can we extend the convex clustering framework to incorporate both numerical and categorical data, while also performing feature selection in high-dimensional datasets?". We choose for a convex clustering method since traditional clustering methods such as hierarchical agglomerative clustering and k-means are known to suffer from algorithmic instability. Additionally, datasets often include both numerical and categorical features, necessitating methods that can accommodate both types since both feature types may play a critical role in the clustering. Moreover, in high-dimensional datasets, it is likely that only a fraction of features contribute meaningfully to clustering, underscoring the importance of feature reduction or selection.

To integrate these diverse requirements into a unified clustering approach, we began by adapting the AMA algorithm for convex clustering introduced by Chi and Lange (2015) to handle categorical data. Subsequently, inspired by works such as B. Wang et al. (2018) and M. Wang and Allen (2021), we incorporated a penalty term to regulate feature influence. Furthermore, we integrated a feature selection framework proposed by Witten and Tibshirani (2010) to enhance the robustness of feature selection in our clustering method.

Through extensive simulation experiments covering spherical and non-spherical cluster shapes (including the half-moon dataset) and applications to high-dimensional mixed gene expression datasets, our convex clustering method has demonstrated superior performance over other widely used clustering techniques. It effectively clusters both convex and non-convex shapes while accurately selecting relevant features in high-dimensional settings. Conversely, other methods struggle when datasets exhibit non-spherical shapes or contain substantial noise in features.

However, we acknowledge that the performance of our convex clustering method can be compromised by the presence of numerous noisy features. Our simulations and real-world dataset analyses indicate that our method's efficacy diminishes when the accuracy of our weighting scheme falls below 50%. In our convex clustering algorithm, positive weights are assigned between observations identified as neighbors, emphasizing proximity over exact nearest neighbors as detailed by Chi and Lange (2015). This necessitates a minimum accuracy threshold of 50% for consistent results.

Moving forward, avenues for future research could focus on refining the approach to weight selection. Given that the weighting scheme plays a pivotal role in convex clustering by determining which observations merge, its accuracy is critical, especially in the presence of noisy features. Additionally, efforts could concentrate on optimizing algorithm efficiency. Our current implementation achieves convergence within a minute for the largest dataset (36 samples, 14,258 features). Touw et al. (2023) and Sun et al. (2021) have demonstrated faster convergence rates compared to the AMA algorithm, suggesting potential areas for computational efficiency.

References

Arthur, D. & Vassilvitskii, S. (2006). *k-means++: The advantages of careful seeding* (Tech. Rep.). Stanford.

Bernardo, J., Bayarri, M., Berger, J., Dawid, A., Heckerman, D., Smith, A. & West, M. (2003). Bayesian clustering with variable and transformation selections. In *Bayesian statistics 7: proceedings of the seventh valencia international meeting* (Vol. 249).

Chang, W.-C. (1983). On using principal components before separating a mixture of two multivariate normal distributions. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, *32*(3), 267–275.

Chavent, M., Lacaille, J., Mourer, A. & Olteanu, M. (2020). Sparse k-means for mixed data via group-sparse clustering. In *Esann 2020-28th european symposium on artificial neural networks, computational intelligence and machine learning* (Vol. 978).

Chi, E. C. & Lange, K. (2015, October). Splitting methods for convex clustering. *Journal of Computational and Graphical Statistics*, *24*(4), 994–1013. Retrieved from `http://dx.doi.org/10.1080/10618600.2014.948181` doi: 10.1080/10618600.2014.948181

Combettes, P. L. & Wajs, V. R. (2005). Signal recovery by proximal forward-backward splitting. *Multiscale modeling & simulation*, *4*(4), 1168–1200.

Dempster, A. P., Laird, N. M. & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society: series B (methodological)*, *39*(1), 1–22.

Diebold, F. X. & Shin, M. (2019). Machine learning for regularized survey forecast combination: Partially-egalitarian lasso and its derivatives. *International Journal of Forecasting*, *35*(4), 1679–1691.

Drineas, P., Frieze, A., Kannan, R., Vempala, S. & Vinay, V. (2004). Clustering large graphs via the singular value decomposition. *Machine learning*, *56*, 9–33.

Ester, M., Kriegel, H.-P., Sander, J., Xu, X. et al. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd* (Vol. 96, pp. 226–231).

Foss, A., Markatou, M., Ray, B. & Heching, A. (2016). A semiparametric method for clustering mixed data. *Machine Learning*, *105*, 419–458.

Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., ... others (1999). Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *science*, *286*(5439), 531–537.

Gower, J. C. (1971). A general coefficient of similarity and some of its properties. *Biometrics*, 857–871.

Hocking, T., Vert, J.-P., Bach, F. & Joulin, A. (2011, 06). Clusterpath an algorithm for clustering using convex fusion penalties. , 745-752.

Huang, Z. (1997). Clustering large data sets with mixed numeric and categorical values. In *Proceedings of the 1st pacific-asia conference on knowledge discovery and data mining,(pakdd)* (pp. 21–34).

Huang, Z. (1998). Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Mining and Knowledge Discovery*, *2*, 283–304.

Jia, Z. & Song, L. (2020). Weighted k-prototypes clustering algorithm based on the hybrid dissimilarity coefficient. *Mathematical Problems in Engineering*, *2020*, 1–13.

Lemaire, V., Ismaili, O. A. & Cornuéjols, A. (2015). An initialization scheme for supervized k-means. In *2015 international joint conference on neural networks (ijcnn)* (pp. 1–8).

Lindsten, F., Ohlsson, H. & Ljung, L. (2011). Clustering using sum-of-norms regularization: With application to particle filter output computation. In *2011 ieee statistical signal processing workshop (ssp)* (pp. 201–204).

Liu, W.-m., Mei, R., Di, X., Ryder, T. B., Hubbell, E., Dee, S., ... others (2002). Analysis of high density expression microarrays with signed-rank call algorithms. *Bioinformatics*, *18*(12), 1593–1599.

MacQueen, J. et al. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth berkeley symposium on mathematical statistics and probability* (Vol. 1, pp. 281–297).

Panahi, A., Dubhashi, D., Johansson, F. D. & Bhattacharyya, C. (2017). Clustering by sum of norms: Stochastic incremental algorithm, convergence and cluster recovery. In *International conference on machine learning* (pp. 2769–2777).

Patel, S., Sihmar, S. & Jatain, A. (2015). A study of hierarchical clustering algorithms. In *2015 2nd international conference on computing for sustainable global development (indiacom)* (pp. 537–541).

Pelckmans, K., De Brabanter, J., Suykens, J. A. & De Moor, B. (2005). Convex clustering shrinkage. In *Pascal workshop on statistics and optimization of clustering workshop.*

Qian, W., Zhang, Y. & Chen, Y. (2021). Structures of spurious local minima in k-means. *IEEE Transactions on Information Theory*, *68*(1), 395–422.

R Core Team. (2023). R: A language and environment for statistical computing [Computer software manual]. Vienna, Austria. Retrieved from `https://www.R-project.org/`

Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, *66*(336), 846–850.

Su, T. & Dy, J. (2004). A deterministic method for initializing k-means clustering. In *16th ieee international conference on tools with artificial intelligence* (pp. 784–786).

Sun, D., Toh, K.-C. & Yuan, Y. (2021). Convex clustering: Model, theoretical guarantee and efficient algorithm. *Journal of Machine Learning Research*, *22*(9), 1–32.

Szepannek, G. (2018). clustmixtype: User-friendly clustering of mixed-type data in r. *The R Journal*, 200-208. Retrieved from `https://doi.org/10.32614/RJ-2018-048` doi: 10.32614/RJ-2018-048

Tamayo, P., Scanfeld, D., Ebert, B. L., Gillette, M. A., Roberts, C. W. & Mesirov, J. P. (2007). Metagene projection for cross-platform, cross-species characterization of global transcriptional states. *Proceedings of the National Academy of Sciences*, *104*(14), 5959–5964.

Tan, K. M. & Witten, D. (2015). Statistical properties of convex clustering. *Electronic journal of statistics*, *9*(2), 2324.

Touw, D. J. W., Groenen, P. J. F. & Terada, Y. (2023). *Convex clustering through mm: An efficient algorithm to perform hierarchical clustering.*

Tseng, P. (1991). Applications of a splitting algorithm to decomposition in convex programming and variational inequalities. *SIAM Journal on Control and Optimization*, *29*(1), 119–138.

Wang, B., Zhang, Y., Sun, W. W. & Fang, Y. (2018, April). Sparse convex clustering. *Journal of Computational and Graphical Statistics*, *27*(2), 393–403. Retrieved from `http://dx.doi.org/10.1080/10618600.2017.1377081` doi: 10.1080/10618600.2017.1377081

Wang, H. & Leng, C. (2008). A note on adaptive group lasso. *Computational statistics & data analysis*, *52*(12), 5277–5286.

Wang, M. & Allen, G. I. (2021). Integrative generalized convex clustering optimization and feature selection for mixed multi-view data. *Journal of Machine Learning Research*, *22*(55), 1–73.

Wang, S. & Zhu, J. (2008). Variable selection for model-based high-dimensional clustering and its application to microarray data. *Biometrics*, *64*(2), 440–448.

Wang, X.-D., Chen, R.-C. & Yan, F. (2019). High-dimensional data clustering using k-means subspace feature selection. *J. Netw. Intell.*, *4*(3), 80–87.

Weylandt, M., Nagorski, J. & Allen, G. I. (2020). Dynamic visualization and fast computation for convex clustering via algorithmic regularization. *Journal of Computational and Graphical Statistics*, *29*(1), 87–96.

Witten, D. M. & Tibshirani, R. (2010). A framework for feature selection in clustering. *Journal of the American Statistical Association*, *105*(490), 713–726.

Xie, B., Pan, W. & Shen, X. (2008). Penalized model-based clustering with cluster-specific diagonal covariance matrices and grouped variables. *Electronic journal of statistics*, *2*, 168.

Xu, J., Xu, B., Zhang, W., Zhang, W. & Hou, J. (2009). Stable initialization scheme for k-means clustering. *Wuhan University Journal of Natural Sciences*, *14*(1), 24–28.

Zhu, C., Xu, H., Leng, C. & Yan, S. (2014). Convex optimization procedure for clustering: Theoretical revisit. *Advances in Neural Information Processing Systems*, *27*.

Zou, H. (2006). The adaptive lasso and its oracle properties. *Journal of the American statistical association*, *101*(476), 1418–1429.

A Derivation AMA

In AMA, we set $\sigma = 0$ in the minimization for a^{t+1} , as such a term drops out which makes computation straightforward. This means that we minimize an ordinary Lagrangian function which is as follows:

$$l(\mathbf{a}, \mathbf{v}, \mathbf{\Lambda}) = \sum_{i=1}^{n} \mathcal{L}^{num}(\mathbf{x}_i^{num}, \mathbf{a}_i^{num}) + \mathcal{L}^{cat}(\mathbf{x}_i^{cat}, \mathbf{a}_i^{cat}) + \gamma \sum_{l \in \mathcal{E}} w_l ||\mathbf{v}_l||_q + \sum_{l \in \mathcal{E}} \langle \boldsymbol{\lambda}_l, \mathbf{v}_l - \mathbf{a}_{l_1} + \mathbf{a}_{l_2} \rangle \quad (40)$$

The first step in AMA minimizes over a in the Lagrangian specification 63

$$\mathbf{a}^{t+1} = \arg\min_{\mathbf{a}} \sum_{i=1}^{n} \mathcal{L}^{num}(\mathbf{x}_i^{num}, \mathbf{a}_i^{num}) + \mathcal{L}^{cat}(\mathbf{x}_i^{cat}, \mathbf{a}_i^{cat}) + \sum_{l \in \mathcal{E}} \langle \boldsymbol{\lambda}_l, \mathbf{v}_l - \mathbf{a}_{l_1} + \mathbf{a}_{l_2} \rangle \quad (41)$$

The objective function can be separated across all i, allowing us to compute the numerical and categorical centroids separately since they are independent of each other. For the numerical centroids we have

$$h^{num} := \frac{1}{2} \sum_{i=1}^{n} ||\mathbf{x}_{i\cdot}^{num} - \mathbf{a}_{i\cdot}^{num}||_2^2 + \sum_{l \in \mathcal{E}} \langle \boldsymbol{\lambda}_l^{num}, \mathbf{v}_l^{num} - \mathbf{a}_{l_1}^{num} + \mathbf{a}_{l_2}^{num} \rangle \quad (42)$$

As the squared Euclidean norm and dot product are convex functions, the overall objective is also convex. Thus taking the derivative and setting it equal to zero results in our update for a_i

$$\begin{aligned} \frac{dh^{num}}{d\mathbf{a}_i^{num}} &= -\mathbf{x}_i^{num} + \mathbf{a}_i^{num} - \sum_{l_1=i} \boldsymbol{\lambda}_l^{num} + \sum_{l_2=i} \boldsymbol{\lambda}_l^{num} = 0 \\ \Rightarrow \mathbf{a}_i^{num} &= \mathbf{x}_i^{num} + \sum_{l_1=i} \boldsymbol{\lambda}_l^{num} - \sum_{l_2=i} \boldsymbol{\lambda}_l^{num} \end{aligned} \tag{43}$$

In similar fashion, for the categorical features, we have the following function to minimize

$$h^{cat} := \sum_{j=m+1}^{p} \left\{ \sum_{k=1}^{K} -x_{ijk}\log(a_{ijk}) - (1-x_{ijk})\log(1-a_{ijk}) \right\} + \sum_{l\in\mathcal{E}} \langle \boldsymbol{\lambda}_l, v_l^{cat} - a_{l_1}^{cat} + a_{l_2}^{cat} \rangle, \quad i = 1, \ldots, n \tag{44}$$

Similarly, taking the derivative and setting it equal to zero gives

$$\begin{aligned} \frac{dh^{cat}}{d\mathbf{a}_i^{cat}} &= -\frac{\mathbf{x}_i^{cat}}{\mathbf{a}_i^{cat}} + \frac{\mathbf{1} - \mathbf{x}_i^{cat}}{\mathbf{1} - \mathbf{a}_i^{cat}} - \sum_{l_1=i} \boldsymbol{\lambda}_l^{cat} + \sum_{l_2=i} \boldsymbol{\lambda}_l^{cat} = 0 \\ \Rightarrow -\mathbf{x}_i^{cat} + \mathbf{a}_i^{cat} &= \sum_{l_1=i} \boldsymbol{\lambda}_l^{cat} - \sum_{l_2=i} \boldsymbol{\lambda}_l^{cat} \\ \Rightarrow \mathbf{a}_i^{cat} &= \mathbf{x}_i^{cat} + \sum_{l_1=i} \boldsymbol{\lambda}_l^{cat} - \sum_{l_2=i} \boldsymbol{\lambda}_l^{cat} \end{aligned} \tag{45}$$

Next, similar to Chi and Lange (2015), the update for $\boldsymbol{\lambda}_l^{t+1}$ is given by:

$$\boldsymbol{\lambda}_l^{t+1} = \boldsymbol{\lambda}_l^t + \sigma(\mathbf{v}_l^{t+1} - \mathbf{a}_{l_1}^{t+1} + \mathbf{a}_{l_2}^{t+1}) \tag{46}$$

Regarding the update for $\mathbf{v}^{t+1}$, we note that it is separable for the vectors $\mathbf{v}_l$.

$$\begin{aligned} \mathbf{v} &= \arg\min_{\mathbf{v}} \gamma w_l ||\mathbf{v}_l||_q + \langle \boldsymbol{\lambda}_l^t, \mathbf{v}_l^{t+1} \rangle + \frac{\sigma}{2} ||\mathbf{v}_l^{t+1} - \mathbf{a}_{l_1}^{t+1} + \mathbf{a}_{l_2}^{t+1}||_2^2 \\ &= \arg\min_{\mathbf{v}} \frac{1}{2} \left[||\mathbf{v}_l^{t+1} - (\mathbf{a}_{l1}^{t+1} - \mathbf{a}_{l2}^{t+1} - \sigma^{-1}\boldsymbol{\lambda}_l^t)||_2^2 + \frac{\gamma w_l}{\sigma} ||\mathbf{v}_l^{t+1}|| \right] \\ &= \text{prox}_{\text{c}||\cdot||}(\mathbf{a}_{l1}^{t+1} - \mathbf{a}_{l2}^{t+1} - \sigma^{-1}\boldsymbol{\lambda}_l^t) \end{aligned} \tag{47}$$

where the proximal operator is defined as

$$\text{prox}_{c\Omega}(\mathbf{x}) = \arg\min_{\mathbf{v}} \left[c\Omega(\mathbf{v}) + \frac{1}{2} ||\mathbf{x} - \mathbf{v}||_2^2 \right] \tag{48}$$

In this case, $\Omega = ||\cdot||_2$ and $c = \frac{\gamma w_l}{\sigma}$. This problem can be solved via block-wise soft-thresholding, hence the update for $\mathbf{v}^{t+1}$ looks as follows

$$\mathbf{v}_l^{t+1} = \left[1 - \frac{\frac{\gamma w_l}{\sigma}}{||\mathbf{a}_{l1}^{t+1} - \mathbf{a}_{l2}^{t+1} - \sigma^{-1}\boldsymbol{\lambda}_l^t||_2} \right] (\mathbf{a}_{l1}^{t+1} - \mathbf{a}_{l2}^{t+1} - \sigma^{-1}\boldsymbol{\lambda}_l^t) \tag{49}$$

Chi and Lange (2015) show that using Moreau's decomposition (Combettes & Wajs, 2005), one can rewrite the proximal operator with the norm as:

$$\operatorname{prox}_{c\Omega}(\mathbf{z}) = \mathbf{z} - \operatorname{proj}_{c\mathbf{B}}(\mathbf{z}), \tag{50}$$

where $\mathbf{B} = \{y : ||y||_\dagger\}$ is the unit ball of the dual norm $||\cdot||_\dagger$. In this case, the dual norm of $||\cdot||_2$ is simply $||\cdot||_2$. Moreover, we note that the solution of projection onto the unit vector $\operatorname{proj}_{\mathbf{B}}(\mathbf{z}) = \frac{\mathbf{z}}{\max\{1, ||\mathbf{z}||\}}$. Thus, when we plug the formula in Equation 47, we get

$$\begin{aligned} \mathbf{v}_l^{t+1} &= \operatorname{prox}_{\sigma_l||\cdot||}(\mathbf{a}_{l1}^{t+1} - \mathbf{a}_{l2}^{t+1} - \sigma^{-1}\boldsymbol{\lambda}_l^t) \\ &= \mathbf{a}_{l1} - \mathbf{a}_{l2} - \sigma^{-1}\boldsymbol{\lambda}_l - \operatorname{proj}_{c\mathbf{B}}(\mathbf{a}_{l1}^{t+1} - \mathbf{a}_{l2}^{t+1} - \sigma^{-1}\boldsymbol{\lambda}_l^t), \end{aligned} \tag{51}$$

where in this case $c = \frac{\gamma w_l}{\sigma}$. Plugging this in Equation 46 leads to

$$\begin{aligned} \boldsymbol{\lambda}_l^{t+1} &= \boldsymbol{\lambda}_l^t + \sigma(\mathbf{v}_l^{t+1} - \mathbf{a}_{l_1}^{t+1} - \mathbf{a}_{l_2}^{t+1}) \\ &= \boldsymbol{\lambda}_l^t + \sigma(\mathbf{a}_{l_1}^{t+1} - \mathbf{a}_{l_2}^{t+1} - \sigma^{-1}\boldsymbol{\lambda}_l^t - \operatorname{proj}_{c\mathbf{B}}(\mathbf{a}_{l_1}^{t+1} - \mathbf{a}_{l_2}^{t+1} - \sigma^{-1}\boldsymbol{\lambda}_l^t) - \mathbf{a}_{l_1}^{t+1} - \mathbf{a}_{l_2}^{t+1}) \\ &= \boldsymbol{\lambda}_l^t + \sigma(-\sigma^{-1}\boldsymbol{\lambda}_l^t - \operatorname{proj}_{c\mathbf{B}}(\mathbf{a}_{l_1}^{t+1} - \mathbf{a}_{l_2}^{t+1} - \sigma^{-1}\boldsymbol{\lambda}_l^t)) \\ &= -\sigma \operatorname{proj}_{c\mathbf{B}}(\mathbf{a}_{l_1}^{t+1} - \mathbf{a}_{l_2}^{t+1} - \sigma^{-1}\boldsymbol{\lambda}_l^t) \\ &= \operatorname{proj}_{C_l}(\boldsymbol{\lambda}_l^t - \sigma(\mathbf{a}_{l_1}^{t+1} - \mathbf{a}_{l_2}^{t+1})), \end{aligned} \tag{52}$$

where we use the identities $-\operatorname{proj}_{c\mathbf{B}}(\mathbf{z}) = \operatorname{proj}_{c\mathbf{B}}(-\mathbf{z})$ and $a\operatorname{proj}_{c\mathbf{B}}(\mathbf{z}) = \operatorname{proj}_{ac\mathbf{B}}(-a\mathbf{z})$ for $a > 0$. Moreover, $C_l = \{\boldsymbol{\lambda}_l : ||\boldsymbol{\lambda}||_\dagger \leq \gamma w_l\}$.

B AMA Dual derivation

Recall that minimizing the Lagrangian on primal variables gives the dual. For a general case, we want to minimize:

$$\begin{aligned} D(\lambda) &= \min_{\mathbf{u},\mathbf{v}} f(\mathbf{a}) + g(\mathbf{v}) + \langle \lambda, \mathbf{c} - \mathbf{A}\mathbf{u} - \mathbf{B}\mathbf{v}\rangle \\ &= \min_{\mathbf{u}} \{f(\mathbf{a}) - \langle \lambda, \mathbf{A}\mathbf{u}\rangle\} + \min_{\mathbf{v}} \{f(\mathbf{u}) - \langle \lambda, \mathbf{B}\mathbf{v}\rangle\} \\ &= \min_{\mathbf{u}} \left\{f(\mathbf{a}) - \langle \mathbf{A}^t\lambda, \mathbf{u}\rangle\right\} + \min_{\mathbf{v}} \left\{f(\mathbf{u}) - \langle \mathbf{B}^t\lambda, \mathbf{v}\rangle\right\} \\ &= -\max_{\mathbf{u}} \left\{\langle \mathbf{A}^t\lambda, \mathbf{a}\rangle - f(a)\right\} - \max_{\mathbf{v}} \left\{\langle \mathbf{B}^t\lambda, \mathbf{v}\rangle - g(\mathbf{v})\right\} \\ &= -f^*(\mathbf{A}^t\lambda) - g^*(\mathbf{B}^t\lambda), \end{aligned} \tag{53}$$

where in the last line we use the convex conjugate $f^*(\mathbf{v}) = \max_{\mathbf{x}\in\operatorname{dom} f} \{\langle\mathbf{x},\mathbf{v}\rangle - f(\mathbf{x})\}$. It now rests to find expressions for the Fenchel conjugates. The conjugate for g^* is the same in Chi and

Lange (2015).

$$\begin{aligned}
g^*(\lambda) &= \sup_{\mathbf{v}} \left[\langle \lambda, \mathbf{v} \rangle - \gamma \sum_{l \in \mathcal{E}} w_l ||\mathbf{v}_l|| \right] \\
&= \sup_{\mathbf{v}} \left[\langle \sum_{l \in \mathcal{E}} \lambda, \mathbf{v} \rangle - \gamma \sum_{l \in \mathcal{E}} w_l ||\mathbf{v}_l|| \right] \\
&= \sum_{l \in \mathcal{E}} \sup_{\mathbf{v}} \left[\langle \lambda, \mathbf{v} \rangle - \gamma w_l ||\mathbf{v}_l|| \right] \\
&= \delta_{C_l}(\lambda_l),
\end{aligned} \tag{54}$$

where in the last line we use the fact that the conjugate of the norm is the convex indicator function of the dual norm. However, in our case $f(a)$ is the essentially the sum of two convex functions. As they each operate on different part of the input variables, we can treat them separately. For the numerical data we have as loss function $\frac{1}{2}||\mathbf{x\text{-}a}||_2^2$. Using the definition of the convex conjugate f^*, we get the problem:

$$f^*(\mathbf{z}) = \sup_{\mathbf{a}} \langle \mathbf{z,a} \rangle - \frac{1}{2}||\mathbf{x\text{-}a}||_2^2 \tag{55}$$

Taking the derivative with respect to a, we get

$$\begin{aligned}
\frac{df^*(\mathbf{z})}{d\mathbf{a}} &= \mathbf{z} + \mathbf{x} - \mathbf{a} = 0 \\
&\Rightarrow \mathbf{a} = \mathbf{z} + \mathbf{x}
\end{aligned} \tag{56}$$

Plugging in the formula yields:

$$\begin{aligned}
f^*(\mathbf{z}) &= \langle \mathbf{z}, \mathbf{z} + \mathbf{x} \rangle - \frac{1}{2}||\mathbf{x} - \mathbf{z} - \mathbf{x}||_2^2 \\
&= ||\mathbf{z}||_2^2 + \langle \mathbf{z}, \mathbf{x} \rangle - \frac{1}{2}||\mathbf{z}||_2^2 \\
&= \frac{1}{2}||\mathbf{z}||_2^2 + \langle \mathbf{z}, \mathbf{x} \rangle
\end{aligned} \tag{57}$$

In similar fashion for $-\mathbf{x}\log(\mathbf{a}) - (1 - \mathbf{x})\log(\mathbf{1} - \mathbf{a})$, we have:

$$\begin{aligned}
f^*(\mathbf{z}) &= \sup_{\mathbf{a}} \langle \mathbf{z}, \mathbf{a} \rangle - (-\mathbf{x}\log(\mathbf{a}) - (\mathbf{1} - \mathbf{x})\log(\mathbf{1} - \mathbf{a})) \\
&= \sup_{\mathbf{a}} \langle \mathbf{z}, \mathbf{a} \rangle + \mathbf{x}\log(\mathbf{a}) + (\mathbf{1} - \mathbf{x})\log(\mathbf{1} - \mathbf{a})
\end{aligned} \tag{58}$$

Taking the derivative with respect to a_i result in:

$$\begin{aligned}
\frac{df^*(\mathbf{z})}{d\mathbf{a}_i} &= \mathbf{z} + \frac{\mathbf{x}}{\mathbf{a}} - \frac{\mathbf{1} - \mathbf{x}}{\mathbf{1} - \mathbf{a}} = 0 \\
&\Rightarrow (\mathbf{1} - \mathbf{a})\mathbf{x} - \mathbf{a}(\mathbf{1} - \mathbf{x}) = -\mathbf{z} \\
&\Rightarrow \mathbf{a} = \mathbf{z} + \mathbf{x}
\end{aligned} \tag{59}$$

Plugging in the equation yields:

$$\begin{aligned} f^*(\mathbf{z}) &= \langle \mathbf{z}, \mathbf{z}+\mathbf{x}\rangle + \mathbf{x}\log(\mathbf{z}+\mathbf{x}) + (\mathbf{1}-\mathbf{x})\log(\mathbf{1}-\mathbf{z}-\mathbf{x}) \\ &= ||\mathbf{z}||_2^2 + \langle \mathbf{z}, \mathbf{x}\rangle + \mathbf{x}\log(\mathbf{z}+\mathbf{x}) + (\mathbf{1}-\mathbf{x})\log(\mathbf{1}-\mathbf{z}-\mathbf{x}) \end{aligned} \tag{60}$$

Before we move on to the final term f^*, we clarify the notation further. Now, $\mathbf{a}$ denote the stacked such that $\mathbf{a} = [\mathbf{a}_1^t \dots \mathbf{a}_n^t]^t$ and similarily for $\mathbf{v} = [\mathbf{v}_1^t \dots \mathbf{v}_l^t]$ and $\mathbf{x} = [\mathbf{x}_1^t \dots \mathbf{x}_n^t]$. Recall that we have the condition: $\mathbf{a}_{l_1} - \mathbf{a}_{l_2} - \mathbf{v}_l = 0 \quad \forall l \in \mathcal{E}$. Hence, if we define $\mathbf{A}$ as $\mathbf{A}^t = [\mathbf{A}_1^t \dots \mathbf{A}_\varepsilon^t]$, where ε is the number of non-zero edges and $\mathbf{A}_l^t = [\mathbf{e}_{l_1} - \mathbf{e}_{l_2}] \otimes \mathbf{I}_p$ which is an $np \times p$ matrix. For example, $\mathbf{A}_l x = [\mathbf{e}_{1_1}^t - \mathbf{e}_{1_2}^t] \otimes \mathbf{I}_p * \mathbf{x} = \begin{bmatrix} x_{l_1 1} - x_{l_2 1} \\ \vdots \\ x_{l_1 p} - x_{l_2 p} \end{bmatrix}$. Moreover $\mathbf{A}^t\lambda$ looks as follows, remember $\mathbf{A}^t$ is a $np \times p\varepsilon$ matrix, and λ is a stacked matrix with dimensions $p\varepsilon \times 1$. For example, assume we take indexes $l_1 = 1$ and $l_2 = 2$, then

$$\mathbf{A}_l^t \lambda_l = \begin{bmatrix} \mathbf{I}_p \\ -\mathbf{I}_p \\ \vdots \\ 0_{(n-2)\times p} \end{bmatrix}_{np \times p} \times \begin{bmatrix} \lambda_{l_1} \\ \vdots \\ \lambda_{l_p} \end{bmatrix}_{p \times 1} = \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_p \\ -\lambda_1 \\ \vdots \\ -\lambda_p \\ 0_{(n-2)\times 1} \end{bmatrix}_{np \times 1} \tag{61}$$

Hence, $\mathbf{A}^t\lambda$ is essentially the sum of each $A_l^t\lambda_l$. Each block represents the sum (or subtraction depending on whether the index i is the first or second index) of the associated λ's. We can now finalize the formula for the dual, namely

$$\begin{aligned} f^*(\mathbf{A}^t\lambda) &= \frac{1}{2}||\mathbf{A}^t\lambda||_2^2 + \langle \mathbf{A}^t\lambda, \mathbf{x}\rangle + ||\mathbf{A}^t\lambda||_2^2 + \langle \mathbf{A}^t\lambda, \mathbf{x}\rangle + \mathbf{x}\log(A^t\lambda + \mathbf{x}) + (\mathbf{1}-\mathbf{x})\log(\mathbf{1} - \mathbf{A}^t\lambda - \mathbf{x}) \\ &= \frac{1}{2}\sum_{i=1}^{n} \left\{ ||\sum_{l_1 = i}\lambda_l - \sum_{l_2 = i}\lambda_l||_2^2 + \sum_{l \in \mathcal{E}} \langle \lambda_l, \mathbf{x}_{l_1} - \mathbf{x}_{l_2}\rangle \right\} + ||\mathbf{A}^t\lambda||_2^2 + \langle \mathbf{A}^t\lambda, \mathbf{x}\rangle + \mathbf{x}\log(\mathbf{A}^t\lambda + \mathbf{x}) \\ &\quad + (\mathbf{1}-\mathbf{x})\log(\mathbf{1} - \mathbf{A}^t\lambda - \mathbf{x}) \end{aligned} \tag{62}$$

C Derivation AMA with Ridge

The derivation is largely the same for the AMA objective with Ridge penalty.

$$\begin{aligned} l(\mathbf{a}, \mathbf{v}, \mathbf{\Lambda}) =& \sum_{i=1}^{n} \mathcal{L}^{num}(\mathbf{x}_i^{num}, \mathbf{a}_i^{num}) + \mathcal{L}^{cat}(\mathbf{x}_i^{cat}, \mathbf{a}_i^{cat}) + \gamma \sum_{l \in \mathcal{E}} w_l ||\mathbf{v}_l||_q + \sum_{l \in \mathcal{E}} \langle \mathbf{\Lambda}_l, \mathbf{v}_l - \mathbf{a}_{l_1} + \mathbf{a}_{l_2}\rangle \\ &+ \beta \sum_{j=1}^{p} ||\mathbf{a}_j - \tilde{\mathbf{a}}_j||_2 \end{aligned} \tag{63}$$

For the numerical centroids we have

$$h^{num} := \alpha\frac{1}{2}\sum_{i=1}^{n}||\mathbf{x}_{i\cdot}^{num} - \mathbf{a}_{i\cdot}^{num}||_2^2 + \sum_{l\in\mathcal{E}}\langle\boldsymbol{\lambda}_l^{num}, \mathbf{v}_l^{num} - \mathbf{a}_{l_1}^{num} + \mathbf{a}_{l_2}^{num}\rangle + \frac{1}{2}\beta\sum_{j=1}^{m}||\mathbf{a}_j^{num} - \tilde{\mathbf{a}}_j^{num}||_2^2 \tag{64}$$

$$\begin{aligned}\frac{dh^{num}}{d\mathbf{a}_i^{num}} &= -\alpha\mathbf{x}_i^{num} + \alpha\mathbf{a}_i^{num} - \sum_{l_1=i}\boldsymbol{\lambda}_l^{num} + \sum_{l_2=i}\boldsymbol{\lambda}_l^{num} + \beta(\mathbf{a}_i^{num} - \tilde{\mathbf{a}}_i^{num}) = 0\\ &\Rightarrow \alpha\mathbf{a}_i^{num} + \beta\mathbf{a}_i^{num} = \alpha\mathbf{x}_i^{num} + \sum_{l_1=i}\boldsymbol{\lambda}_l^{num} - \sum_{l_2=i}\boldsymbol{\lambda}_l^{num} + \beta\tilde{\mathbf{a}}_i^{num}\\ &\Rightarrow \mathbf{a}_i^{num} = \frac{\alpha\mathbf{x}_i^{num} + \sum_{l_1=i}\boldsymbol{\lambda}_l^{num} - \sum_{l_2=i}\boldsymbol{\lambda}_l^{num} + \beta\tilde{\mathbf{a}}_i^{num}}{(\alpha+\beta)}\end{aligned} \tag{65}$$

In similar fashion, for the categorical features, we have the following function to minimize

$$h^{cat} :=(1-\alpha)\sum_{j=m+1}^{p}\left\{\sum_{k=1}^{K} -x_{ijk}\log(a_{ijk}) - (1-x_{ijk})\log(1-a_{ijk})\right\} + \sum_{l\in\mathcal{E}}\langle\boldsymbol{\lambda}_l, v_l^{cat} - a_{l_1}^{cat} + a_{l_2}^{cat}\rangle, \quad i = 1,\dots,n \tag{66}$$

$$+\frac{1}{2}\beta\sum_{j=m+1}^{p}||\mathbf{a}_j^{cat} - \tilde{\mathbf{a}}_j^{cat}||_2^2 \tag{67}$$

Similarly, taking the derivative and setting it equal to zero gives

$$\begin{aligned}\frac{dh^{cat}}{d\mathbf{a}_i^{cat}} &= (1-\alpha)\left(-\frac{\mathbf{x}_i^{cat}}{\mathbf{a}_i^{cat}} + \frac{\mathbf{1}-\mathbf{x}_i^{cat}}{\mathbf{1}-\mathbf{a}_i^{cat}}\right) - \sum_{l_1=i}\boldsymbol{\lambda}_l^{cat} + \sum_{l_2=i}\boldsymbol{\lambda}_l^{cat} + \beta(\mathbf{a}_i^{cat} - \tilde{\mathbf{a}}_i^{cat}) = 0\\ &\Rightarrow -(1-\alpha)\mathbf{x}_i^{cat} + (1-\alpha)\mathbf{a}_i^{cat} + \beta\mathbf{a}_i^{cat} = \sum_{l_1=i}\boldsymbol{\lambda}_l^{cat} - \sum_{l_2=i}\boldsymbol{\lambda}_l^{cat} + +\beta\tilde{\mathbf{a}}_i^{cat}\\ &\Rightarrow \mathbf{a}_i^{cat} = \frac{(1-\alpha)\mathbf{x}_i^{cat} + \sum_{l_1=i}\boldsymbol{\lambda}_l^{cat} - \sum_{l_2=i}\boldsymbol{\lambda}_l^{cat} + \beta\tilde{\mathbf{a}}_i^{cat}}{1-\alpha+\beta}\end{aligned} \tag{68}$$

The dual also remains similar except for the fact that we add the ridge penalty's. Using the definition of the convex conjugate f^*, we get the problem:

$$f^*(\mathbf{z}) = \sup_{\mathbf{a}}\langle\mathbf{z},\mathbf{a}\rangle - \frac{1}{2}\alpha||\mathbf{x}\text{-}\mathbf{a}||_2^2 - \frac{1}{2}\beta||\mathbf{a} - \tilde{\mathbf{a}}||_2^2 \tag{69}$$

Taking the derivative with respect to a, we get

$$\begin{aligned}\frac{df^*(\mathbf{z})}{d\mathbf{a}} &= \mathbf{z} + \alpha\mathbf{x} - \alpha\mathbf{a} - \beta\mathbf{a} + \beta\tilde{\mathbf{a}} = 0\\ \Rightarrow \mathbf{a} &= \frac{\mathbf{z} + \alpha\mathbf{x} + \beta\tilde{\mathbf{a}}}{\alpha+\beta}\end{aligned} \tag{70}$$

Plugging in the formula yields:

$$\begin{aligned} f^*(\mathbf{z}) &= \langle \mathbf{z}, \frac{\mathbf{z} + \alpha\mathbf{x} + \beta\tilde{\mathbf{a}}}{\alpha + \beta} \rangle - \frac{1}{2}||\mathbf{x} - (\frac{\mathbf{z} + \alpha\mathbf{x} + \beta\tilde{\mathbf{a}}}{\alpha + \beta})||_2^2 \\ &= \frac{1}{1+\beta}\left(||\mathbf{z}||_2^2 + \langle \mathbf{z}, \mathbf{x} \rangle + \langle \mathbf{z}, \beta\tilde{\mathbf{a}} \rangle\right) - -\frac{1}{2}||\mathbf{x} - (\frac{\mathbf{z} + \mathbf{x} + \beta\tilde{\mathbf{a}}}{1 + \beta})||_2^2 \end{aligned} \tag{71}$$

In similar fashion for the categorical variables, we have:

$$\begin{aligned} f^*(\mathbf{z}) &= \sup_{\mathbf{a}} \langle \mathbf{z}, \mathbf{a} \rangle - (1-\alpha)(-\mathbf{x}\log(\mathbf{a}) - (\mathbf{1} - \mathbf{x})\log(\mathbf{1} - \mathbf{a})) + \frac{1}{2}\beta||\mathbf{a} - \tilde{\mathbf{a}}||_2^2) \\ &= \sup_{\mathbf{a}} \langle \mathbf{z}, \mathbf{a} \rangle + (1-\alpha)\mathbf{x}\log(\mathbf{a}) + (1-\alpha)(\mathbf{1} - \mathbf{x})\log(\mathbf{1} - \mathbf{a}) - \frac{1}{2}\beta||\mathbf{a} - \tilde{\mathbf{a}}||_2^2 \end{aligned} \tag{72}$$

Taking the derivative with respect to a_i result in:

$$\begin{aligned} \frac{df^*(\mathbf{z})}{d\mathbf{a}_i} &= \mathbf{z} + (1-\alpha)\frac{\mathbf{x}}{\mathbf{a}} - (1-\alpha)\frac{\mathbf{1} - \mathbf{x}}{\mathbf{1} - \mathbf{a}} - \beta\mathbf{a} + \beta\tilde{\mathbf{a}} = 0 \\ &\Rightarrow (1-\alpha)((\mathbf{1} - \mathbf{a})\mathbf{x} - \mathbf{a}(\mathbf{1} - \mathbf{x})) - \beta\mathbf{a} = -\mathbf{z} - \beta\tilde{\mathbf{a}} \\ &\Rightarrow \mathbf{a} = \frac{\mathbf{z} + (1-\alpha)\mathbf{x} + \beta\tilde{\mathbf{a}}}{1 - \alpha + \beta} \end{aligned} \tag{73}$$

Plugging in the equation yields:

$$f^*(\mathbf{z}) = \langle \mathbf{z}, \frac{\mathbf{z} + (1-\alpha)\mathbf{x} + \beta\tilde{\mathbf{a}}}{1 - \alpha + \beta} \rangle + \mathbf{x}\log(\frac{\mathbf{z} + (1-\alpha)\mathbf{x} + \beta\tilde{\mathbf{a}}}{1 - \alpha + \beta}) + (\mathbf{1} - \mathbf{x})\log(\mathbf{1} - (\frac{\mathbf{z} + (1-\alpha)\mathbf{x} + \beta\tilde{\mathbf{a}}}{1 - \alpha + \beta})) \tag{74}$$