UNIVERSITY OF TARTU
Faculty of Science and Technology
Institute of Computer Science
Computer Science Curriculum

Hasan Mohammed Tanvir

Meta-Learning Based Approach for Automated Pre-processing for Clustering

Master's Thesis (30 ECTS)

Supervisor: Dr. Radwa Elshawi
University of Tartu

Tartu 2022

Meta-Learning Based Approach for Automated Pre-processing for Clustering

Abstract:
Data pre-processing is an integral part of any data analysis project. There are wide range of Data pre-processing methods, such as replacing missing values, scaling, and data reduction. The aim of this project is to automate data pre-processing by leveraging Automated Machine Learning (AutoML). While supervised learning has been in core focus of AutoML research, unsupervised learning remained comparatively less unexplored. Therefore, the thesis focuses on suggesting a data pre-processing pipeline for an unsupervised clustering task by exploiting meta-learning space and meta learners in a domain-agnostic manner for users who does not have in depth knowledge of the machine learning algorithms. The thesis explores the potential of integrating data preprocessing approach to a meta-learning-Based framework for automated algorithm selection and hyperparameter tuning for clustering, named CsmartML built on scikit-learn with 8 clustering algorithm. The proposed methodology applies meta-learning and creates a knowledge space on each of the 112 benchmark datasets. We show that the performance of cSmartML when integrating the automated preprocessing component is often much better than the original clustering result. The comparison with cSmartML showed that the proposed data preprocessing improved the clustering result 0.3% to 27% in 7 out of 10 real datasets and 4% to 44% in 3 out of 6 artificial datasets. In addition, experimentation reveals that the proposed approach takes advantage of the defined objective functions on multi-objective functions framework. This shows that data preprocessing for unsupervised clustering task is as important as supervised learning. Additionally based on the meta-learning space, the project also proposes user a pipeline of data preprocessing, algorithm selection including hyperparameter tuning for further clustering.

Keywords: Automated machine learning, Data pre-processing, Clustering, AutoML

CERCS: P176 - Artificial Intelligence

Metaõppel põhinev lähenemine klastrite automaatseks eeltöötluseks

Lühikokkuvõte:
Andmete eeltöötlemine on iga andmeanalüüsi projekti lahutamatu osa. Andmete eeltöötlusmeetodeid on mitmeid, näiteks puuduvate väärtuste asendamine, skaleerimine ja andmete puhastamine. Käesoleva projekti eesmärgiks on automatiseerida andmete eeltöötlemist automatiseeritud masinaõppe (AutoML) abil. Kuigi juhendatud õpe on olnud kesksel kohal AutoML meetodi arendamisel, siis juhendamata õpet on senini suhteliselt väheuuritud. Seetõttu pakub käesolev töö välja andmekonveieri (data pipeline), mis võimaldaks piiratud masinõppe algoritme puudutavate eelteadmistega kasutajatel teostada klastrite juhendamata eeltöötlust, kasutades selleks metaõppe ruumi ja metaõppijaid domeen-agnostilisel viisil. Töö uurib võimalust integreerida andmete eeltöötlemist metõppe-põhisesse raamistikku (cSmartML), võimaldamaks automatiseeritud algoritmi valikut ning klastrite hüperparameetrite häälestamist. Kavandatav metoodika hõlmab metaõpet ja loob uut teadmist iga 112 võrdlusandmekogumi lõikes. Ilmneb, et cSmartML jõudlus on automatiseeritud eeltöötluskomponendi integreerimisel sageli palju parem kui algne klastrite töötlus. cSmartML-l põhinev analüüs näitas, et välja pakutud andmete eeltöötlemise lähenemine parandas päris andmestike puhul klastrite tulemust 0.3%-27% 7 juhul 10-st ning kunstlike andmestike puhul 4%-44% 3 juhul 6st. Lisaks ilmneb katsetest, et töös välja pakutud lähenemine kasutab määratletud objektiivseid funktsioone multi-objektiivsete funktsioonide raamistikus, mis tõestab, et andmete eeltöötlus juhendamata klastrite töötlemisel on sama oluline kui juhendatud õpe. Lisaks pakub antud projekt metaõppe ruumile tuginedes andmete eeltöötluse ja algoritmi valiku (sealhulgas hüperparameetrite häälestamine) konveieri, mida saab kasutada edasiseks klastrite töötlemiseks.

Võtmesõnad: Automatiseeritud masinaõpe, Andmete eeltöötlemine, Klasterdamine, AutoML

CERCS: P176 - Tehisintellekt

Contents

1 Introduction

As the abundance of data quadrupled over the years by the inventions of data collection and storage capacities, so did the pressure to analyze those data and extract knowledge out of the binary bits. The knowledge discovery, known as the data analysis process, consists of several steps, such as data selection, data pre-processing, data mining and evaluation [3]. The analysis requires individuals with specific knowledge and expertise to prepare the data in order to analyze them. During the process majority of the time, 50-80% of the time is spent after data pre-processing that makes this part of the process to be pivotal to any data analysis project [13]. Data professionals decide on the application of data pre-processing techniques or combinations of techniques to be used in order to yield the best outcome. Such decisions require years of experience and in-depth knowledge in the domain of statistics, data science, and Machine Learning (ML) to understand the nuances of the data, including machine learning algorithms and various ways those algorithms are impacted by the modification of the data. One can apply the best learning algorithm yet receive a sub-standard result because of unprocessed data. It is a challenging and time-consuming task of any project to determine the best performing data pre-processing techniques, even for a seasoned data professional. The growing demand of extracting knowledge out of the ever-growing data sphere requires more data professionals, but there is a lack of individuals with that skill-sets [8]. Therefore, the demand for off-the-shelf data pre-processing solutions for non-experts is growing. Such solutions may speed up research in other fields, such as psychology, biology, sociology, etc., and extract information out of those data. Such solutions would also help small and mid-scale business ventures who cannot afford data scientists to explore and make the best use of their resources in ways that would maximize socio-economic profits.

1.1 Motivation

Traditional machine learning models cannot solve the aforementioned challenge; therefore, the emergence of AutoML that provides an automated machine learning pipeline for the user-specific task, such as regression or classification. Although the research in AutoML gained traction in recent years, specifically in the domain of supervised learning, studies and research are incommensurate with unsupervised learning [20]. The reason behind such disproportion is generally due to the lack of information in unsupervised problems required for validation purposes referred to as "ground truth"; the true label of the clusters. This leads to the complications of cluster evaluation during the AutoML process. Although few of the recent research endeavors addressed the problem and derived a solution from the perspective of

algorithm and hyper parameter tuning, none of the research, to the current state of knowledge, addressed the problem as a combined problem of three elements; data preprocessing, algorithm selection, and hyperparameter tuning. The motivation behind this thesis stemmed from this very problem statement that aims to address the challenges.

1.2 Problem Statement

The objective of this thesis is to combine data preprocessing with the existing problem of combining algorithm selection and hyperparameter tuning, namely CASH problem, with data preprocessing for unsupervised clustering. The CASH problem is denoted as follows,

$$\mathcal{CS} = \mathcal{A} \times \mathcal{H}$$

Where $\mathcal{CS}$ is the configuration space, $\mathcal{A}$ is the problem of algorithm selection and $\mathcal{H}$ is the problem of hyperparameter tuning. The thesis combines this problem with data preprocessing, denoted as transformation $\mathcal{T}$ in the paper. Therefore, the problem statement stands at;

$$\mathcal{CS} = \mathcal{T} \times \mathcal{A} \times \mathcal{H}$$

Let us consider $\mathcal{D}$ denote a set of n data sets $\mathcal{D} = D_1, D_2, \ldots, D_n$, $F(D) = f_1, \ldots, f_k$ denote a set of k meta-features extracted from data set D. Also $\mathcal{T} = T_1, T_2, \ldots, T_n$ denote a set of transformations applied on the datasets.

Let us consider $\mathcal{A} = A_1, \ldots, A_m$ is a set of clustering algorithms, and Λ_i denote the domain of hyper-parameters of algorithm A_i. Finally, let $L(A_i\lambda), D)$ denote the loss of A_i with hyperparameters $\lambda \in \Lambda_i$ on D. Then, the problem statement comes to combining data preprocessing with existing CASH problem and to find the joint algorithm and hyperparameter setting that minimizes the loss [18]:

$$A^*, \lambda^* = \underset{A_i \in \mathcal{A}, \lambda \in \Lambda_i, T_i \in \mathcal{T}}{\arg\min} L(A_i(\lambda), T_i D)$$

1.3 Contribution

The contribution of the thesis can be summarized in the following points,

- Leveraging the concept of meta-learning to provide a data preprocessing pipeline for unsupervised clustering tasks.
- Selecting best performing algorithm, and hyperparameter tuning for an unsupervised clustering tasks.

- Comparing the results against cSmartML results to check if data preprocessing improved the results and if such a solution could be adopted.
- As a whole, combining data preprocessing with the existing CASH problem and address the lack of research in unsupervised AutoML tasks and provide the users a solution to automated data preprocessing pipeline, especially for clustering tasks.

1.4 Thesis Organization

The organization of the thesis is as follows; first, there is a discussion about the theoretical background that covers the basics of ML, including unsupervised learning and clustering algorithms that were considered for this research. This section also includes the cluster validity indices (CVI) and the basics of hyperparameter tuning. The next section focuses on the related work in the domain on AutoML and their comparative analysis that leads to the preliminaries, which comprises of the basic understanding of the datasets, details of the created metadata, data preprocessing techniques. After that, the proposed methodology section describes the details of the working principle in two subsections, offline and online. The result section shows the result after the applied process. The result section also discusses the result in two parts, offline and online. The offline phase shows the impact of data preprocessing against the unprocessed data on clustering, and the online phase shows the result of the suggested pipeline on real-life datasets which are absent from the meta-learning knowledge space. The result section focuses on a comparative analysis with cSmartML tool and its impact on processed data leading to a conclusion focusing on the task itself, outcome, challenges, and future work.

2 Theoretical Background

The scope of this section is to provide information about the theoretical background required to perform the tasks mentioned in the thesis. The sub-sections provide the necessary theoretical background to ML, Unsupervised Learning & Clustering, Cluster Validity Indices (CVI), and Hyper-parameter tuning, respectively.

2.1 Machine Learning

There has been a significant effort, advancement, and breakthrough success in the research of ML. By the invention and application of deep neural networks led to solving complex problems in the field of Computer Vision, Natural Language, and Speech Processing that were unthinkable before. We experience the benefit of such advancements in our day-to-day life, from movies, music, or book recommendation system to autonomous driving vehicles. Constant research and development in data pre-processing to neural network architectures have invented a plethora of powerful toolkits for data professionals, and each of these toolkits comes with its own specificity, such as parameters, dimensionality reduction, data scaling. All these methods are part of data pre-processing and are required for an ML pipeline. Data professionals have to spend hours after configuring the settings to yield the best result through trial and error. This process is quite manual and time-consuming, not only for large datasets but also for the effective application of a simple machine learning algorithm. One simple example is imputing missing value with a relevant value. The process of choosing the relevant value can be difficult. If the number of such missing values is in larger quantity, imputation can skew the final data in various ways, and removing them might lead to a shortage of data. Both of the cases should be avoided. A data scientist requires a profound knowledge of the dataset in order to initiate any pre-processing on the dataset. For non-experts, this algorithm-specific tuning may require more time compared to a seasoned data professional.

Hence the challenge emerges; how to use all the available tools collectively and effectively in an automated manner that would help non-experts to focus more on their respective research fields by leveraging the power of machine learning which leads us to the field of AutoML. Details are discussed in the related work section.

2.2 Unsupervised Learning & Clustering

There are two types of machine learning, supervised and unsupervised learning. Supervised learning is based on a model trained to learn mappings to labels, referred to as ground truth, exploring the pattern of data. Unsupervised learning, on the

contrary, learns without having a ground truth or label that makes the task a bit difficult. Unsupervised learning is applied in anomaly detection, association mining, dimensionality reduction, and clustering. In clustering, the data points are split into groups based on their homogeneity, meaning; the measure of similarity based on their distance.
There are two types of clustering; hard clustering and soft clustering.

- In hard clustering, each data point is assigned to one group or cluster. A data point cannot be present in two cluster at the same time.
- Whereas, soft clustering assigns the probability of a data point belonging to a specific cluster.

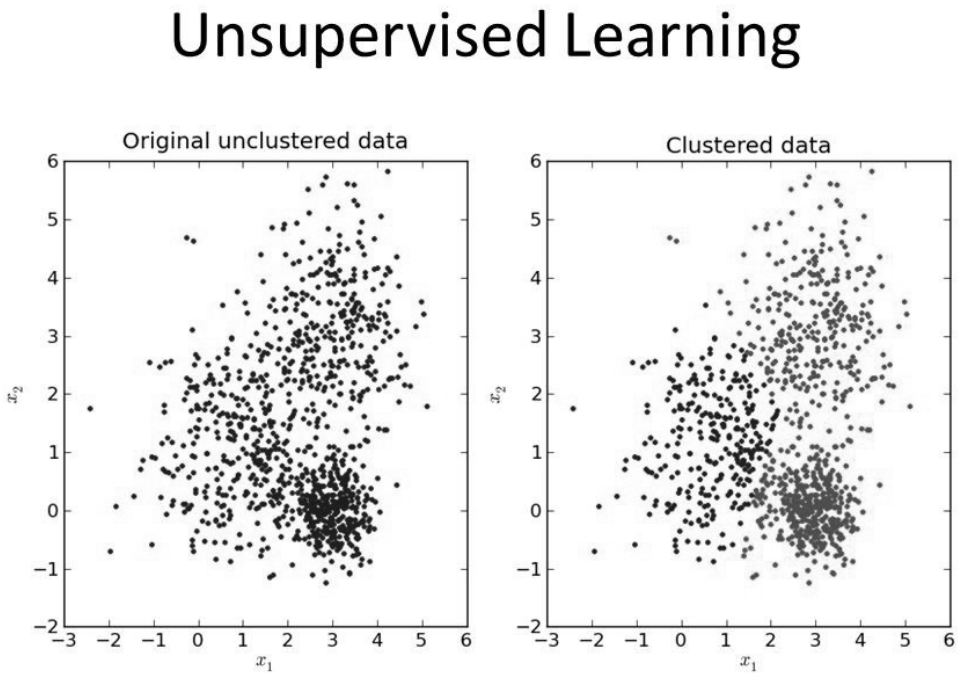

Figure 1. Unsupervised learning, unclustered vs clustered data[1]

Figure 1 shows unclustered vs clustered, hard cluster, data. And hard clustering could be divided into 9 categories[22].
Table 1 shows various clustering algorithm based on various features.

Clustering Algorithm Based on	Algorithms
Partition	K-means, K-medoids, PAM, CLARA, CLARANS
Hierarchy	Agglomerative, BIRCH, CURE, ROCK, Chameleon
Fuzzy Theory	FCM, FCS, MM
Distribution	DBCLASD, GMM
Density	DBSCAN, OPTICS, Mean-shift
Graph Theory	CLICK, MST
Grid	STING, CLIQUE
Fractal Theory	FC
Model	COBWEB, GMM, SOM, ART

Table 1. Clustering Algorithms based on different clustering techniques [22]

We will discuss the most used ones that were applied in this thesis for meta-learning purpose; K-means, Birch, and Agglomerative.

2.2.1 K-Means

K-means is a partition based clustering algorithm that splits the whole data samples into similar groups based on their similarity measure. Among all the similarity measure, Euclidean distance is most commonly used technique [15]. Following are the steps to cluster data using K-means;

Step 1: Randomly choose the initial cluster centroids C_1, C_2 ... C_k, where k is the number of clusters from the given data set X_1, X_2, X_3 ... X_n.

Step 2: For each point X_i, find the nearest centroid C_j

$$\arg\max_{j} D(x_i, c_j)$$

Where D is the distance measure between the data point x_i and cluster center c_j. Assign the point X_i to the cluster j.

Step 3: Compute new cluster center as

[1]https://medium.com/the-21st-century/machine-learning-a-strategy-to-learn-and-understand-chapter-3-9daaad4afc55

$$C_j^* = \frac{1}{n_j} \sum_{x_i \to C_j} x_i$$

Step 4: If $C_j^* = C_j$ then the algorithm converges otherwise repeat from Step 2.

Step 5: It indicates that the algorithm has executed the maximum number of iterations if it does not converge in Step 4 and the algorithm stops.

Euclidean Distance Euclidean distance between two points p_1 and p_2 having n dimensions can be mathematically expressed as;

$$D_{Eucl.} = \sqrt{\sum_{i=1}^{n} (p_{1_i} - p_{2_i})}$$

2.2.2 BIRCH

Balanced Iterative Reducing and Clustering using hierarchies or BIRCH is an effective and fast clustering algorithm, introduced by Zhang et al [23]. Although BIRCH outperforms most of the other clustering algorithms by up to two orders of magnitude [23], BIRCH requires cluster count as input to. be able to cluster effectively. However, we used BIRCH algorithm in our experiment as it is one of the most important clustering algorithm from the hierarchy group. Table 1.

BIRCH has three parameters; branching factor, threshold, and the cluster count.

Step 1: Data loaded and a height-balanced tree is built where each node represents a cluster in the cluster hierarchy. Intermediate nodes are considered as superclusters and the leaf nodes are actual clusters. The branching factor is a global parameter and denoted by the maximum number of children of a node.

Step 2: Condense data by building a smaller CF tree.

Step 3: Global clustering; cluster radii is computed for each cluster. Every new point starts at the root and recursively walks down the tree, entering the subcluster with the nearest center till the leaf node.if the radius is not increased beyond the threshold, the new point is added to the leaf cluster.

$$Cluster\ center,\ C_i = \frac{1}{n_i}\sum_{j}^{n} x_{ij},\ where\{x_{ij}\}_{j=1}^{1}$$

$$Cluster\ radii,\ R_i = \sqrt{\frac{1}{n_i}\sum_{j}^{n}(x_{ij} - C_i)^2}$$

Step 4: Otherwise a new cluster is created with the new point as its only member. Thus, the threshold controls the size of the clusters.

2.2.3 Agglomerative

Agglomerative clustering is also a hierarchical clustering method. It starts with n groups, each containing initially one data point. Two of such most similar data point keeps merging at each stage until there is a single group containing all the data. Later a typical heuristic for large n runs k-means and then apply hierarchical clustering to the estimated cluster centers. A binary tree, dendrogram represents the merging process [7].

2.2.4 Model Selection

After the models are applied and added to the knowledge base, the proposed methodology performs model selection based on the multi CVI correlation score. Given a set of learning algorithms $\mathcal{A}$ and a limited amount of training data $\mathcal{D} = (x_1, y_1), ..., (x_n, y_n)$, the goal of model selection is to determine the algorithm $\mathcal{A}^* \in \mathcal{A}$ with optimal generalization performance. Generalization performance is estimated by the maximum evaluation score by applying the learning functions f_i of all the algorithms, $\mathcal{A}$ on the dataset, $\mathcal{D}$. This can be written as a following;

$$A^* \in \underset{A \in \mathcal{A}}{argmax}\ \mathcal{E}(A, \mathcal{D}^{(i)})$$

Where, $\mathcal{E}(A, \mathcal{D}^{(i)})$ is the evaluation score, in our case, the multi CVI correlation score achieved by applying A on the dataset $\mathcal{D}^{(i)}$.

2.3 Cluster Evaluation Metrics

The quality of clusters requires evaluation through defined metrics. Such evaluation metrics for finding the best clustering solutions are known as cluster validity indices or CVI. The CVIs can be classified into two sections, such as external and internal cluster validation [2, 16]. Such validation of internal and external measures are not

always specific and no single such CVI can evaluate the quality of the clusters [19]. In the following section, internal and external validity indices are explained.

2.3.1 Internal Indices

Internal indices provide information about the internal quality of the clusters by relying on the intrinsic structure of the data to quantify how good a partitioning is in terms of the compactness and separation between clusters. If the ground truth is available, there are a few techniques to validate the clusters, but in case there is not; there are several approaches to validate a cluster.

Some of the internal indices used commonly are Dunn, Davies–Bouldin, Calinski–Harabas [1]. There are also other internal indices for the same measure of compactness and separation of clusters, such as Root-mean-square standard deviation, R-squared, Modified Hubert T, I-index, Silhouette index, Xie-Beni, SDbw etc [11]. Some of such internal indices are described in the following sections.

2.3.2 Banfeld Referty

Banfeld Referty is the weighted sum of the logarithms of the traces of the variance covariance matrix of each cluster. The index can be denoted as;

$$\mathcal{C} = \sum_{k=1}^{K} n_k \log \frac{Tr(WG^k)}{n_k}$$

Trace of within-group scatter matrix is denoted as $Tr(WG)$, number of clusters is denoted as K, and n_k as the sample size.

2.3.3 Davies Bouldin

Davies Bouldin is another internal cluster validity index. It estimates the cohesion based distance from the points in a cluster to its centroid and the separation based on the distance between centroids. The following is the equation for the index.

$$\mathcal{C} = \frac{1}{K} \sum_{k=1}^{K} M_k = \frac{1}{K} \sum_{k=1}^{K} \max_{k\prime \neq k} \frac{\delta_k + \delta_{k\prime}}{\Delta_{kk\prime}}$$

Where, Davies Bouldin is the mean value among the the clusters of quantities M_k where, M_k is the maximum quotients $\frac{\delta_k + \delta_{k\prime}}{\Delta_{kk\prime}}$ for each cluster, k where $k\prime \neq k$.

And δ_k is the mean distance of the points belonging to cluster C_k to their barycenter $G^{\{k\}}$.

$$\delta_k = \frac{1}{n_k} \sum_{i \in I_k} \|M_i^{\{k\}} - G^{\{k\}}\|$$

$$\Delta_{kk\prime} = d(G^{\{k\}}, G^{\{k\prime\}}) = \|G^{\{k\prime\}} - G^{\{k\}}\|$$

Where, $\Delta_{k,k\prime}$ denotes the distance between the barycenters $G^{\{k\}}$ and $G^{\{k\prime\}}$ of clusters C_k and $C_{k\prime}$.

2.3.4 SDbw

SDbw calculates the density of the points belonging to two clusters. The SDbw index is defined as the sum of the mean dispersion in the clusters $\mathcal{S}$ and of the between-cluster density $\mathcal{G}$ [2].

$$\mathcal{C} = \mathcal{S} + \mathcal{G}$$

Where, the quantity $\mathcal{S}$ is the mean of the norms of the vectors $\mathcal{V}^{\{k\}}$ divided by the norm of vector $\mathcal{V}$:

$$\mathcal{S} = \frac{\frac{1}{K} \sum_{k=1}^{K} \|\mathcal{V}^{\{k\}}\|}{\|\mathcal{V}\|}$$

Where, the $\mathcal{V}$ is the vector of variances for each variable in the data set.

$$\mathcal{V} = Var(V_1), ..., Var(V_p)$$

Similarly, variance vectors $\mathcal{V}^{\{k\}}$ for each cluster C_k:

$$\mathcal{V}^{\{k\}} = Var(V_1^{\{k\}}), ..., Var(V_p^{\{k\}})$$

and cluster density $\mathcal{G}$ is defined as;

$$\mathcal{G} = \frac{2}{K(K-1)} \sum_{k<k\prime} R_{kk\prime}$$

Where, the quotient $R_{kk\prime}$ between the density at the midpoint and the largest density at the two barycenters;

$$R_{kk\prime} = \frac{\gamma_{kk\prime}(H_{kk\prime})}{max(\gamma_{kk\prime}(G^{\{k\}}), \gamma_{kk\prime}(G^{\{k\prime\}}))}$$

where the densities, $\gamma_{kk\prime}$, for the barycenters $G^{\{k\}}$ and $G^{\{k\prime\}}$ of the clusters and for their midpoint $H_{kk\prime}$.

[2] https://cran.r-project.org/web/packages/clusterCrit/vignettes/clusterCrit.pdf

The density $\gamma_{kk\prime}$ for a given point, relative to two clusters C_k and $C_k\prime$, is equal to the number of points in these two clusters whose distance to this point is less than σ. Where σ is equal to the square root of the sum of the norms of the variance vectors $V^{\{k\}}$ divided by the number of clusters;

$$\sigma = \frac{1}{K}\sqrt{\sum_{k=1}^{K} \|\mathcal{V}^{\{k\}}\|}$$

2.3.5 External Indices

External validity indices measure similarity of clustering to a-priori known class labels or ground truth. They take into account only the distribution of the points in the different clusters and do not measure the quality of the distribution. But for external indices, the class labels are required and often labels are not available. Adjusted Rand Index (ARI), Accuracy, and Normalized Mutual Information (NMI) are few of the external indices. NMI was used during the multi CVI ranking correlation calculation described in the following section

2.3.6 Normalized Mutual Information (NMI)

Normalized Mutual Information (NMI) is an information for theoretic measure of how close the predicted cluster labels $\hat{y}$ are to the ground-truth labels y. It is defined as,

$$NMI = \frac{I(\hat{y}; y)}{max\{H(\hat{y}), H(y)\}}$$

where $I(\hat{y}; y)$ is the mutual information between the ground-truth labels y and the predicted cluster labels $\hat{y}$, and $H(\cdot)$ denotes their entropies. NMI is in the range of $[0, 1]$, with 0 meaning no correlation and 1 exhibiting perfect correlation [10].

Another approach is the stability based validation. It is is not model dependant and free from any assumption of compactness. This method does not directly validate a partition, but it relies on the stability of the clustering algorithm over different samples of the input dataset [1].

2.3.7 Multi-CVI Ranking

Using a single cluster validity index reflects a single measure of goodness of a partitioning. Therefore, it is important to simultaneously optimize several cluster quality measures that can capture different data characteristics. For the scope of the thesis,

Index	Interval	Objective
Banfeld Raferty	$[-\infty, \infty]$	Min
Davies Bouldin	$[0, \infty]$	Min
SDbw	$[0, \infty]$	Min
Dunns Index	$[0, \infty]$	Max
McClain Rao	$[0, \infty]$	Min
PBM Index	$[0, \infty]$	Max
Ratkowsky-Lance	$[0, \infty]$	Max
Ray-Turi	$[0, \infty]$	Min
Scott Symons	$[-\infty, \infty]$	Min
I-index	$[0, \infty]$	Max
Modified Hubert T	[-1, 1]	Max

Table 2. Cluster validity indices with their intervals and objectives

three such internal indices are considered, but in a combined manner as suggested by Elshawi et al.,[19]. As propose, each cluster is evaluated using NSGA II based on every possible three clustering objective functions. Two clustering validity indices are used for building Pareto fronts and the third validity index is used independently for sorting each front that creates a ranking for possible three validity indices and dataset. The accuracy of the ranking is evaluated by comparing two rankings: the rankings obtained from the supervised Normalized Mutual Information (NMI) [9], and the ranking obtained from every possible three validity indices. To evaluate the similarity of the two rankings, Spearman's rank correlation is used which gives a one dimensional value, 0 to 1, higher value indicates better cluster quality. The correlation value makes it easier to decide and understand about the quality of the clusters.

Table 2 shows the intervals of the internal indices and their objective function.

2.4 Hyperparameter Tuning

Parameters based on which the model is designed known as hyperparameters. Hyperparameter tuning requires intensive experimentation to figure out which value works best as, unlike model parameters, there is no loss function to optimize for hyperparameters. Hyperparameter is crucial to any machine learning model for superior performance yet very experimental. Therefore, the whole tuning process is an experiment heavy and the best result, as of now, could be achieved through empirical studies. For the scope of this research, part of the hyperparameter tuning process is included during the meta-learning space creation process.

For this research, only two hyperparameters were considered for tuning to save time on the creation of the meta-learning space. One is number of clusters for all the three clustering algorithms applied on the datasets and maximum iteration for K-means algorithm. Details are discussed in the proposed methodology section.

3 Related Work

This section of the thesis covers relevant studies in the same or related domain and explains the competitiveness of those methods. It is divided into two sections, AutoML for classification and AutoML for clustering. Both the section describe researches in their respective fields from the perspective of combining algorithm selection and hyperparameter tuning, known as CASH problem, which can be defined as

$$\mathcal{CS} = \mathcal{A} \times \mathcal{H}$$

where $\mathcal{CS}$ is the configuration space, $\mathcal{A}$ is the problem of algorithm selection and $\mathcal{H}$ is the problem of hyperparameter tuning. The majority of the research only deals with one of the problems without combining both. The CASH problem could be further formalized as follows;

let $\mathcal{D}$ denote a set of n data sets $\mathcal{D} = D_1, D_2, \ldots, D_n$. Also, let $F(D) = f_1, \ldots, f_k$ denote a set of k meta-features extracted from data set D. Let $\mathcal{A} = A_1, \ldots, A_m$ is a set of clustering algorithms, and let Λ_i denote the domain of hyper-parameters of algorithm A_i. Finally, let $L(A_i\lambda), D)$ denote the loss of A_i with hyperparameters $\lambda \in \Lambda_i$ on D. Then, the Combined Algorithm Selection and Hyperparameter optimization (CASH) problem is to find the joint algorithm and hyperparameter setting that minimizes this loss [18]:

$$A^*, \lambda^* = \underset{A_i \in \mathcal{A}, \lambda \in \Lambda_i}{\arg\min}\, L(A_i(\lambda), D)$$

The following section describes the approaches from the CASH problem perspective.

3.1 AutoML for Classification

As machine learning becomes popular, a variety of solutions are developed to support the AutoML domain. The domain of such development is diverse, and packages target both academia and industry focusing on minimizing the time required for a data analysis project and providing the off-the-shelf solution to non-experts. AutoML pipelines also optimize the application of algorithms by tuning the parameters to obtain a better outcome. As a whole, AutoML combines the theory of algorithm selection and hyper-parameter tuning in a manner that provides a wholesome solution speeding up the whole data analysis project.

A brief review of the familiar tools for classification is provided below.

3.1.1 Auto-Weka

Auto-Weka is the first such solution that combined algorithm selection and hyperparameter tuning as a joint problem. Auto-Weka is named after the open-source data analysis tool WEKA. WEKA is written in JAVA and able to apply machine learning models and feature selectors. The challenge for machine learning given a dataset is to automatically choose a learning algorithm and set its hyperparameters to optimize the result. Auto-Weka combines algorithm selection and hyperparameter - CASH; optimization problem [20].

Auto-Weka approaches the problem through Sequential Model Algorithm Configuration – SMAC, a Bayesian Optimization method with Random Forests. Some of the key features of SMAC are the following:

- SMAC handles conditional parameters by instantiating inactive ones to default values for model training and prediction. This allows individual decision trees to create splits which check if a hyperparameter is active.
- The Expected Improvement Criterion is used to choose the next configuration setting for evaluation.
- CASH problem optimizes the function as the mean of a set of losses as calculated by Cross Validation for different pairs of D_{train}, D_{test}. SMAC progressively better estimates by evaluating one points at a time. A new configuration outperforms the previously best resulting one only if it shows better result in 1-fold to n-fold. Provided the new configuration is better, the number of folds for the evaluation is increased.
- SMAC selects a random configuration to evaluate every two evaluations in order to make the procedure robust.

Auto-Weka conducted the experiment on 21 different datasets including MNIST, CIFAR-10, three different Grid Search methods, Random Search and using the default parameters to the Weka algorithms. Results are presented for a 10-fold Cross Validation [20].

3.1.2 Auto-Sklearn

Auto-Sklearn is a familiar AutoML package that is written in Python. Data preprocessors and learners of SciKit-Learn, including Classification Algorithms and data preprocessing methods, are available in Auto-Sklearn with a focus on supervised classification. Auto-Sklearn also implements SMAC for hyperparameter tuning

following the study of Auto-Weka.

Auto-Sklearn leverages meta-learning to warm-start the Bayesian optimization procedure and an ensemble step that ensures more than one configuration in the optimization procedure is used. The Bayesian optimization warm-starts by initialization from configurations that produced the best results in a similar dataset. Then SMAC is applied on 140 data sets and the best resulting configurations are stored according to the mean accuracy of a 10-fold Cross-Validation. Similarities among datasets are defined by the L_1 distance among 38 Meta Features extracted from each data set. Furthermore, a post-processing method is suggested to utilize models trained during the course of Bayesian Optimization and later discarded. A selection of 50 of those models may provide near-optimal results. Then it constructs an ensemble with those ensemble selections as the method for calculating weights [5].

Later as a comparison, Feurer et al [5] used the same setting presented in Auto-Weka evaluation. Finally, the results were tested for the vanilla Auto-Sklearn with and without Meta-Learning and ensemble building, on a collection of 140 datasets from OpenML according to balanced classification error rate (BER). Both of the methods proved to improve performance over vanilla Auto- Sklearn. Meta-Learning showed improvements to all of the configuration evaluations in the Bayesian optimization procedure and ensemble construction proved to benefit from Meta-Learning as better performing models were chosen to build the ensemble stack [5].

3.1.3 Auto-Net

Auto-Net is an AutoML framework for tuning deep learning Neural Networks. Between two versions of Auto-Net, the first one is considered as an extension to Auto-SKlearn focusing only on hyperparameter optimization for fully connected feed-forward Neural Networks [12]. Auto-Net is based on Lasagne[3], a Deep Learning library in Python that is built on top of the deep learning framework Theano[4]. Same as Auto-Weka and Auto-Sklearn, Auto-Net applies SMAC for hyperparameter optimization in a configuration space of 63 hyperparameters. The number of layers varies from one to six, so that the training time of a single configuration is low.

The second version is built on PyTorch[5]. It is also known as Auto-PyTorch. Auto-PyTorch supports a variety of different deep learning modules such as network type, learning rate, scheduler, etc. At the time of writing, Auto-PyTorch supports

[3]https://github.com/Lasagne/Lasagne
[4]https://github.com/Theano/Theano
[5]https://github.com/pytorch/pytorch

four different network types: Multi-Layer Perceptrons, Residual Neural Networks, Shaped Multi-Layer Perceptrons, and Shaped Residual Networks, tuning them for a set of 112 hyperparameters.

3.1.4 TPOT

TPOT, short for Tree-based Pipeline Optimization Tool, is an open-source Python project. Instead of Bayesian optimization methods, it applies genetic programming for creating machine learning pipelines [14]. TPOT focuses on supervised learning, specifically classification applying 150 SciKit-Learn algorithms, including preprocessing methods. One hundred tree-based pipelines are generated in the initial generation and further optimized according to the Python package DEAP. According to cross-validation accuracy and minimized number of processes, 20 pipelines are selected to mutate and produce a new generation of pipelines. Each pipeline produces five more pipelines through cross-over. The algorithm executes for 100 generations where a Pareto front of the non-dominated solutions being updated in each generation.

The authors used a variety of benchmark datasets that are subject to 30 replicates of the procedure with different random generator seeds to evaluate the method. TPOT showed improvements over median accuracy varying from 10% to 60% while only degrading on the scale of 2%-5% when performing worse. Furthermore, it allowed for the discovery of useful preprocessors such as RandomizedPCA for a benchmark dataset leading to near-perfect accuracy. In another scenario, RandomSearch produced very competitive results compared to those of TPOT, but it did not take into account the number of pipeline operations, which led to complex and computationally intensive solutions [14].

3.1.5 Google Vizier

Google Vizier is state-of-the-art research that focuses on Google's Cloud Machine Learning subsystem HyperTune[6]. It is implemented in Python, C++, and Golang. Google Vizier can dynamically select optimization algorithms; for instance, among a variety of optimization algorithms, the default is Batched Gaussian Process Bandits for short scale studies and recommends proprietary local-search algorithms for large scale ones.

It also includes automated early stopping, a technique that aims to terminate parameter exploration if the next configuration is not predicted to be any better than the last configuration. The technique mentioned is implemented by two

[6]https://github.com/GoogleCloudPlatform/cloudml-hypertune

	HPO	**Implementation Platform**	**Key Features**
Auto-Sklearn	Supervised	Bayesian-Random Forests	Python
Autto-Weka	Supervised	Bayesian-Random Forests	Weka
TPOT	Supervised	Genetic Opaimization	Python
Auto-Net	Supervised	SMAC	Python
Google Vizier		Variety of Algorithms – Batched Gaussian Process	Python, C#, Golang

Table 3. Comparative analysis of the AutoML packages for classification

methods; the performance curve stopping rule, where a Gaussian process regressor is trained on a partial performance curve and parameters of the optimization procedure to predict whether the optimal value of the objective function is found until any given point is sufficiently low to terminate the process. The second one is the median stopping rule, a rule that terminates the optimization process if, at any given step, the objective found is strictly worse than the running average of the previous ones. The approach used for transfer learning is to build a stack of Gaussian Process Regressors where each of the regressors is associated with an optimization procedure that has already taken place and regresses on the residual of its objective relative to the prediction of the regressor below. Each component of the stack is placed according to chronological order, with the top stack representing the most recent studies [6].

The result is demonstrated against random search and four implemented algorithms in Google Vizier. All four algorithms showed improvements over the random search, while Gaussian Processes regressors and the probabilistic model showed the best performance. Between both the early stopping criteria, Median Automated Stopping Rule showed the best results. It sped up the random search process by a factor of two [6].

Table 3 shows the comparative analysis in a table for better understanding.

3.2 AutoML for Clustering

There is significantly less research in the field of AutoML for unsupervised clustering, mainly due to the challenge of evaluation criteria for clustering. However, for unsupervised learning tasks, a few approaches were developed in recent years to support novice analysts with the selection of the best performing clustering algorithm, where configuration space is $\mathcal{CS} = \mathcal{A}$ for certain problems [4, 17]. These methods are based on meta-learning, solely focused on the clustering algorithm $\mathcal{A}$, and completely ignoring the corresponding hyperparameters $\mathcal{H}$.

The following subsections describe a few of such solutions that addressed the CASH problem.

3.2.1 AutoML4Clust

AutoML4Clust approaches the problem by combining both the algorithm and hyperparameter tuning problem. It is implemented on the top of scikit-learn and supports only k-center clustering algorithm, including K-means, MiniBatch K-means, k-Medoids, and GMM. For hyperparameter tuning, AutoML4Clust only tunes the number of clusters k over a fixed search space, such that the maximum k value is set in relation to the number of instances in the dataset.
For evaluation, the user should choose between three internal metrics, including Calinski-Harabasz, Davies-Bouldin Index, and Silhouette. AutoML4Clust uses Bayes optimizer, Hyperban, or BOHB to find well-performing configurations efficiently [21].

According to Tschechlov et al., it is possible to select clustering algorithms of any kind by defining $\mathcal{CS}$ in a hierarchical way, similar to existing AutoML systems for supervised learning tasks. According to the paper, AutoML4Clust shows results clearly that it is orders of magnitude faster than the time-consuming exhaustive search (ES). It achieves the fastest results in 57 seconds, while the fastest results for the ES require roughly 6 hours [21].

3.2.2 AutoClust

AutoClust is an end-to-end framework for automatic clustering algorithm selection based on meta-learning and cluster validity indices which approaches the problem by applying a combination of meta-learning and Bayesian optimization techniques. The framework proposes a method for hyperparameter tuning of clustering algorithms, which capitalizes on a new optimization criterion, namely regression of cluster validity indices. The author claims that empirical evaluation of the approach using various real-life data sets demonstrates the framework's advantages against

state-of-the-art methods.

The framework considers eight clustering algorithms, such as K-means, DBSCAN, OPTICS, Birch, Spectral, Agglomerated, Affinity Propagation, and MeanShift with a limited hyperparameter space.

AutoClust was compared against a baseline approach, namely ARI. The best-performing clustering algorithm is determined by the Silhouette coefficient, a metric that the authors choose due to its popularity for evaluating clustering results. For 24 data sets, AutoClust outperforms the baseline in half of the cases, and in several cases, the difference is substantial (up to 0.62). The baseline is better in 7 cases, but the difference is much smaller (at most 0.20). Essentially, this experiment demonstrates that AutoClust usually outperforms the common practice of running multiple clustering algorithms and keeping the best [18].

3.2.3 cSmartML

cSmartML is another such framework that addresses the challenge of CASH problem. It provides a solution that takes care of algorithm selection and hyperparameter tuning at the same time as a single problem.
The framework consists of four main phases, including the input phase, algorithm and metric selection, hyper-parameter optimization, and computing the output and updating the knowledge base. In the input phase, the user uploads the dataset and specifies the time budget constraint for metric selection, algorithm selection, and hyper-parameter tuning process. The algorithm and metric selection phase are partitioned into two main components; meta-feature extraction and meta-learning recommendation.

cSmartML exploits a set of meta-features described and eight clustering techniques, including, K-means, DBSCAN, OPTICS, Birch, Spectral, Agglomerated, Affinity Propagation, and MeanShift. It assesses the quality of the clustering solutions using 12 internal indices. Each dataset is clustered using each of the eight algorithms with various hyper-parameter settings and evaluated with each possible combination of three objective clustering functions to obtain rankings of the best configurations. It computes the correlation to configuration rankings obtained from external validation with Normalized Mutual Information (NMI) using Spearman's rank correlation in order to evaluate the correctness of the multi-objective rankings. For each dataset, cSmartML stores the clustering algorithm along with the three validity indices, which exhibited the highest correlation coefficient for the meta-learning purpose. cSmartML considers main hyperparameters, such as n-clusters, max-iter, n-init MeanShift bandwidth, and conditional hyperparameters, such as

xi, eps, gamma etc [19].
cSmartML performs a comparison of the average correlation between the NMI and each of the best single objective ranking and the best multi-objective ranking of 300 clustering on 15 datasets. The correlation between the expected NMI ranking and the multi-objective rankings is above 0.71. The result from cSmartML was also compared against default baseline, grid-search baseline, random-search baseline, and AutoML4Clust. In the majority of cases, cSmartML outperforms the mentioned baseline frameworks [19].

4 Preliminaries

This section describes the specifics related to the proposed methodology. The subsections describe datasets used for the project, metadata, the transformations applied on the dataset.

4.1 Dataset

The first step is about preparing the meta-learning space that requires various datasets. For the scope of this work, we have used 112 artificial datasets from clustering benchmark[7]. Few of the datasets have been omitted during the meta-learning phase to avoid further pre-processing. The dataset used for the meta-learning phase is synthetic data representing structured and unstructured shapes when visualized. Figure 2 shows various shapes present in the dataset used for creating the meta-learning space. The datasets have a range of features, holding numeric data. There are multiple classes ranging from 2 to 41. The dataset also comprises several shapes and sizes.

The metadata of these datasets was extracted for creating the metaspace. Details of the metadata can be found in the metadata subsection of this chapter.

4.2 Metadata

The fundamental task of this project is to create the meta-learning space. The meta-learning space consists of metadata of datasets mentioned in the previous section. Metadata provides information about the dataset regardless of the content of the data. In general, metadata represents the nature of the dataset through structural characteristics such as mean, median of the dataset, number of instances, variance, standard deviation, the number of instances, the number of attributes, predictive accuracies, multi CVI ranking correlation in this case, of the algorithms applied on datasets, etc.—that jointly represent the relationships of algorithms with datasets. Different meta-learning systems may use different characteristics of datasets and different performance measures. Figure 3 shows a high-level metadata creation scenario.

There are hundreds of such metadata, and there is no defined methodology to find the set that will yield the best results. But these extractions might become costly, and a trade-off must be made between the amount of metadata and the purpose.

In order to determine the list of metadata to be used in the thesis, a previous study by Elshawi et al., [19] was followed since the topic of both the studies is

[7]https://github.com/deric/clustering-benchmark

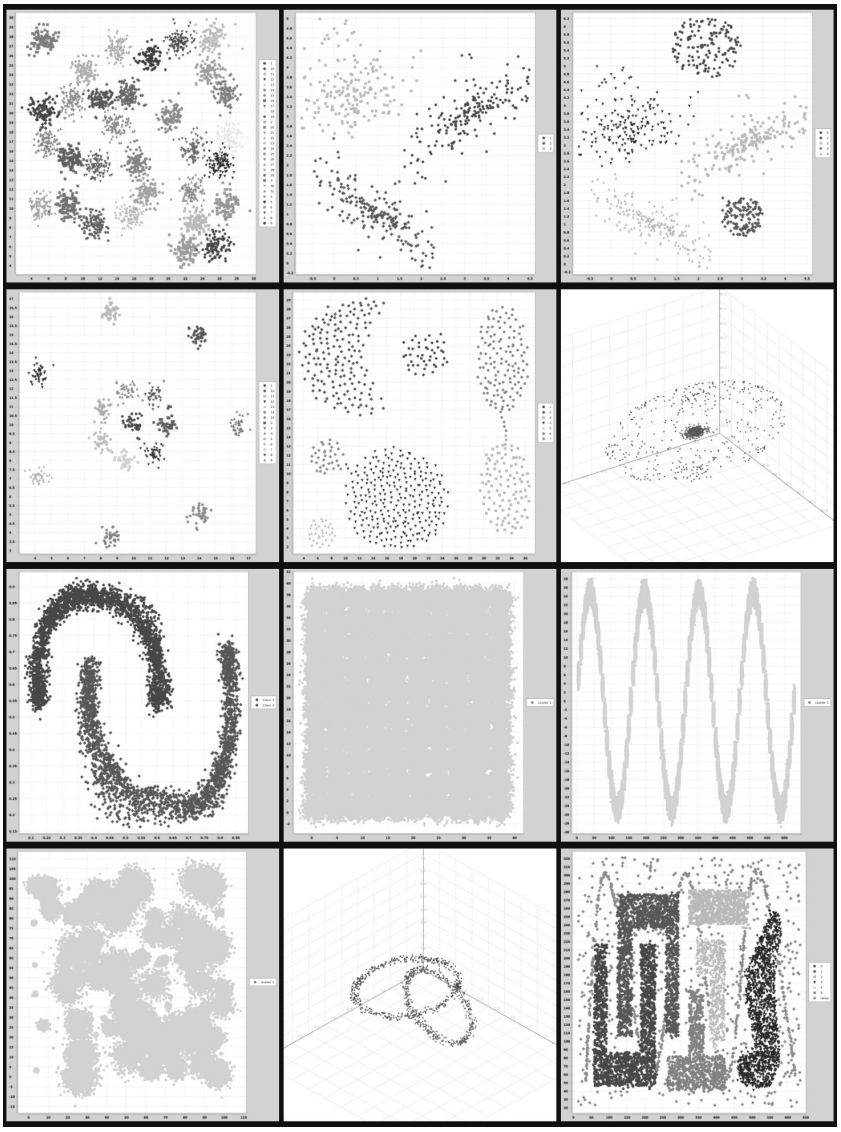

Figure 2. Various shapes of artificial dataset used for meta-learning[8]

about unsupervised clustering. In addition to those set of metadata, a few more metadata is included in the whole list, such as number of instances, number of categorical and continuous features, number of classes. The metadata used in the scope of the thesis is shown in Table 4.

[8]https://github.com/deric/clustering-benchmark

metadata (MD)	**Description**
MD1	Mean of distances vector
MD2	Variance of distances vector
MD3	Standard deviation of distances vector
MD4	Skewness of distances vector
MD5	Kurtosis of distances vector
MD6	Percentage of distance values in each of ten intervals that equally comprise range [0,1]
MD7	Percentage of distance values in each of ten intervals that equally comprise range [0,1]
MD16	Percentage of distance values with absolute z-score in four intervals of range $[0, \infty)$
MD17	Percentage of distance values with absolute z-score in four intervals of range $[0, \infty)$
MD18	Percentage of distance values with absolute z-score in four intervals of range $[0, \infty)$
MD19	Number of categorical values
MD20	Number of continuous values
MD21	Number of classes
MD22	Number of instances
MD23	Number of features

Table 4. Description of metadata

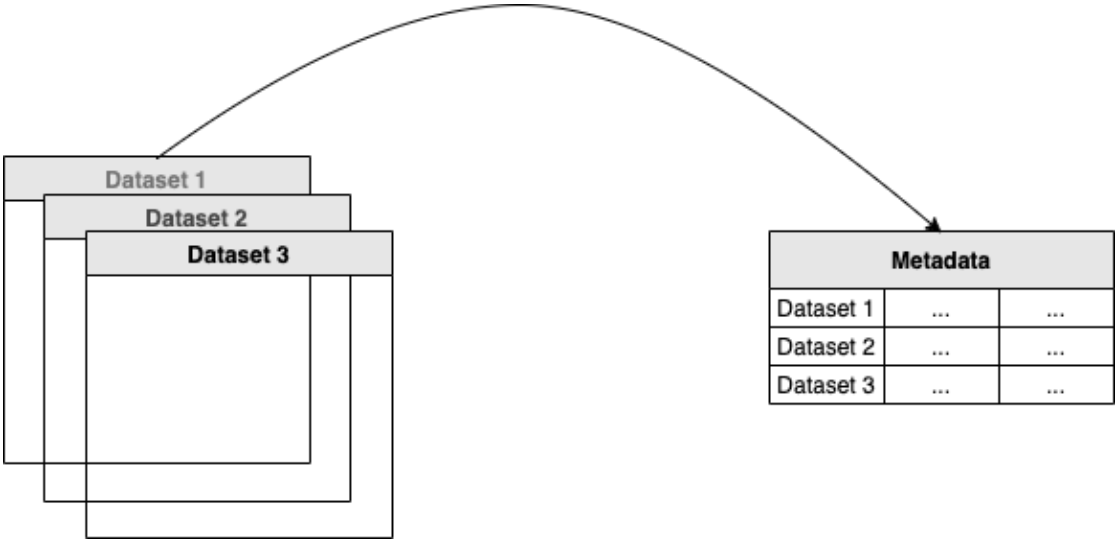

Figure 3. Dataset to metadata

4.3 Transformations

In order to create the meta-learning space and find out the impact of transformations, various data preprocessing methods, several transformations were applied to the datasets. After the transformations of the dataset, they were clustered again, and the multi CVI score was calculated. For the scope of this thesis, the transformations applied can be found in Table 5. The table also shows how the nature of the data changes after the application of the transformations. The technique column shows which specific methods are used, while input and output column shows the change of status of the data.

4.3.1 Discretization

Discretization is a technique to convert continuous data to categorical data. The method is also known as binning. With a given range of values, the data are put into the specific bin it falls. As a result, the data loses its continuous attributes and becomes a categorical value. Sometimes, discretization helps to ML algorithm to perform better. For this project, KBin discretizer was applied from SciKit-Learn [9] python package with default parameters.

[9]https://scikit-learn.org/

4.3.2 Normalization

As a second step to preprocess the data, normalization was applied on the continuous data, which sustains the continuity of the dataset while normalizing the data. Normalization can refer to an array of techniques to scale data and bring to a probability distribution of adjusted values into the range 0 to 1. L1 and L2 normalization from SciKit-Learn was applied in this project.

4.3.3 Scaling

Another step is data scaling which again brings the data to a given scale for the model to learn better. MinMaxScaler from SciKit-Learn library was applied on the features scale them to a range of minimum and maximum value.

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Where, x_{scaled} =scaled value, x = observed value, x_{min} = minimum value of the data, x_{max} = maximum value of the data.

4.3.4 Standardization

Data standardization is another process of scaling where the values are centered around the mean with a unit standard deviation. The Z-score standardization method from the SciKit-Learn library is used for the standardization of data.

$$Z = \frac{x - \mu}{\sigma}$$

Where Z= standard score, x= observed value, μ = mean of the sample, σ = standard deviation of the sample.

4.3.5 Imputation

Imputation refers to replacing missing values in the dataset. There are a few ways to deal with missing values. One way is to eliminate the missing rows, but if the number of missing rows is in the majority, it becomes a costly preprocessing since the majority of the data would be lost. One way is to replace the missing values using the mean, median, or mode of the dataset. While mean and median are common for continuous values, the mode could be used for categorical values. SimpleImputer with the mean and median strategy were used from the SciKit-Learn library. Since all of the features of all the datasets have continuous data, mode was not used in the process.

Transformations	Techniques	Input	Output
Discretization	KBin	Continuous	Categorical
Normalization	L1	Continuous	Continuous
Normalization	L2	Continuous	Continuous
Scaling	MinMax	Continuous	Continuous
Standardization	Z score	Continuous	Continuous
Imputation	Mean	Continuous	Continuous
Imputation	Median	Continuous	Continuous

Table 5. List of Transformations applied

5 Proposed Methodology

There are two steps of the proposed set of processes. The first step is about preparing the meta-learning space considering hyper-parameter tuning, applied transformations, and a combination of Multi-CVI, which would be used as an evaluation metric; this step is considered as the offline phase. The second step consists of finding the closest dataset from the meta-learning space leveraging the metadata, and based on that suggesting the pipeline of transformations and clustering model for optimum result and finally applying the suggestions and receiving the result; this step is known as the online phase. In order to find the closest dataset, the nearest neighbor algorithm was applied in the online phase. Later, we perform the transformations on the dataset and calculate the multi CVI correlation score on four test datasets. Details of each of the phases are described in the latter subsection.

Figure 4 shows the proposed working principle of the thesis.

5.1 Offline Phase

The offline phase consists of extracting the metadata from the dataset and creating the bigger meta-learning space, which includes the extracted metadata of a particular dataset, best-performing clustering algorithms including hyperparameter settings in the given range, applied transformation, or no transformation and corresponding multi-CVI score.

5.1.1 Extracting Metadata

Referring to the Metadata subsection mentioned earlier, in this stage, the metadata of the datasets are extracted, and metadata is created. The dataset holds information table 4 for each of those 112 datasets. The table 6 shows a sample of

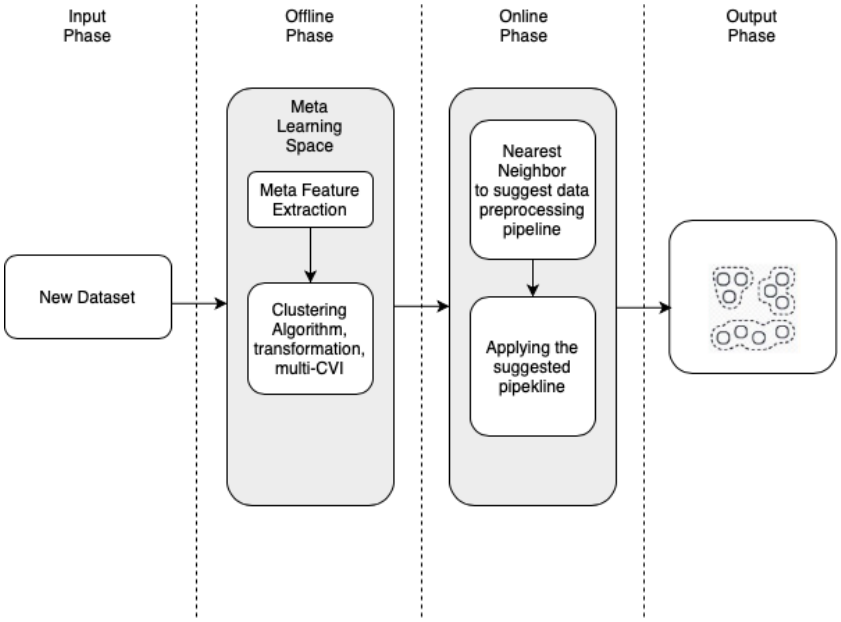

Figure 4. Proposed working principle

Metadata		
Dataset 1	...	...
Dataset 2	...	...
Dataset 3	...	...

Transformations & Clustering algorithm applied on each dataset and Multi CVI caluclated

Meta Learning Space				
Name	Metadata	Transformations	Algorithm	Multi_CVI
Dataset 1	...	No_Tr	KMeans	0.67
Dataset 1	...	Tr_1	Birch	0.75
Dataset 1	...	Tr_1_2	Birch	0.70
Dataset 1	...	No_Tr	Birch	0.60
...	...	...	...	...

Figure 5. Metadata to establishing meta-learning Space

those metadata extracted in this stage. For creating the metadata, package from cSmartML [19] is used.

5.1.2 Cluster Dataset & Hyperparameter Tuning

Provided the metadata is extracted, clustering algorithms were applied on the datasets with a range of hyperparameters. Initially, no transformations were applied to the dataset. The multi-CVI ranking correlation score is calculated after clustering the original data. While calculating the multi-CVI, the python package DEAP was used to pick the pareto optimal front from the ranking of the clusters. Each dataset produces three rows with three best-performing clustering algorithms

name	MD22	MD23	MD21	MD19	MD20	MD1	MD3	MD2
disk-4000n	4000	2	2	0	2	0.01	0.01	75e-6
disk-5000n	5000	2	2	0	2	0.01	0.01	6e-5
zelnik4	622	2	4	0	2	0.03	0.02	48e-5
chainlink	1000	3	2	0	3	0.03	0.02	3e-4

name	MD5	MD4	MD6	MD16	MD17	MD18	MD3	MD2
disk-4000n	-1.11	0.246	100	26.49	20.09	0.66	0.01	75e-6
disk-5000n	-1.11	0.246	100	26.49	20.09	0.66	0.01	6e-5
zelnik4	-1.11	0.246	100	26.50	20.09	0.66	0.02	48e-5
chainlink	-1.11	0.246	100	26.50	20.09	0.66	0.02	3e-4

Table 6. Metadata after extraction

on the original dataset. This step is important because, in some cases, the clustering algorithm with hyperparameter tuning may perform best without even further preprocessing. Having this information in meta-learning space will ensure that we do not exclude the scenario where such settings may outperform scenarios where a transformation or preprocessing method is applied.

After the initial clustering, the transformations mentioned in the following sections were applied to the datasets, and the clustering was performed, including the hyperparameter range. Based on the clustering, the multi-CVI is again calculated and added to the meta-learning space.

5.1.3 Transformations Applied

Transformations refer to various data preprocessing methods that were applied to the datasets. Before applying the clustering algorithms, each dataset goes through the transformations twice. First, each transformations methods from table 5 are applied to the datasets exactly once, and the clustering algorithms are applied. Secondly, a combination of two transformation methods is applied to the dataset, and the clustering was performed again on the dataset.

For instance, we can denote the process following way, We apply $tr_1, ..., tr_7$ is first applied once on the datasets, $D_1, D_2, ..., D_n$. Then we apply the clustering algorithm on the transformed datasets, $tr_1D_1, tr_1D_2, ..., tr_1D_n, ..., tr_7D_1, tr_7D_2, ..., tr_7D_n$ and calculate the multi-CVI score.

For the second phase, we apply $tr_1, ..., tr_7$ twice on the datasets, $D_1, D_2, ..., D_n$. Then we apply the clustering algorithm on the transformed datasets, $tr_1tr_2D_1$,

name	disk-4000n
MD22	4000
MD23	2
MD21	2
MD19	0
MD20	2
...	...
best_cluster_setting	[AgglomerativeClustering(n_clusters=14)]
transformation	SimpleImputer()
multi_cvi	Banfeld_Raferty, davies_bouldin_score, SDbw
multi_cvi_correlation_score	0.62022

Table 7. Sample of meta-learning space

$tr_1tr_3D_2, ..., tr_2tr_1D_n, ..., tr_7tr_1D_1, tr_7tr_2D_2, ..., tr_7tr_6D_n$ and calculate the multi-CVI score. All the multi-CVi score creates a row in the meta-learning space.

For imputation transformation, 20% of the data were randomly removed and replaced by both mean and median imputation strategy as the dataset did not have missing values.

5.2 Online Phase

After the meta-learning space is ready, given a new dataset, the online phase calculates the nearest neighbor distance between the datasets of meta-learning space and the new dataset. For calculating the closest dataset, the nearest neighbor algorithm from sklearn is used with the euclidean distance metric. Then it suggests the closest dataset from the knowledge base. And based on the knowledge base, the system suggests which transformation method could be applied for the best possible clustering result, including the algorithm and hyperparameter sets.

Later the exact method suggested by the system is applied to the new test dataset, and the multi-CVI ranking correlation score is calculated.

For both the offline and online phase, a laptop with an intel i5 7th generation processor and 16 GB of memory was used. It took around four days to create the meta-learning space with the given hyperparameter range. The time increases significantly when the number of hyperparameter range is increased. Therefore, the hyperparameter range was kept at a level so that the time complexity is reduced and the initial result is achieved.

6 Result

This section covers the result achieved from the implementation of the proposed methodology. Here we demonstrate the offline phase performance that shows the impact of transformations and clustering algorithms on the overall clustering result: the online phase covers, implementation of the method on the test dataset, and a comparison with existing cSmartML. For the online phase study, four datasets were chosen randomly, which is absent from the knowledge base so that the proposed system can have a nonbiased outcome on the given dataset.

6.1 Offline Phase

The result and statistics from the offline phase is presented in this section. Figure

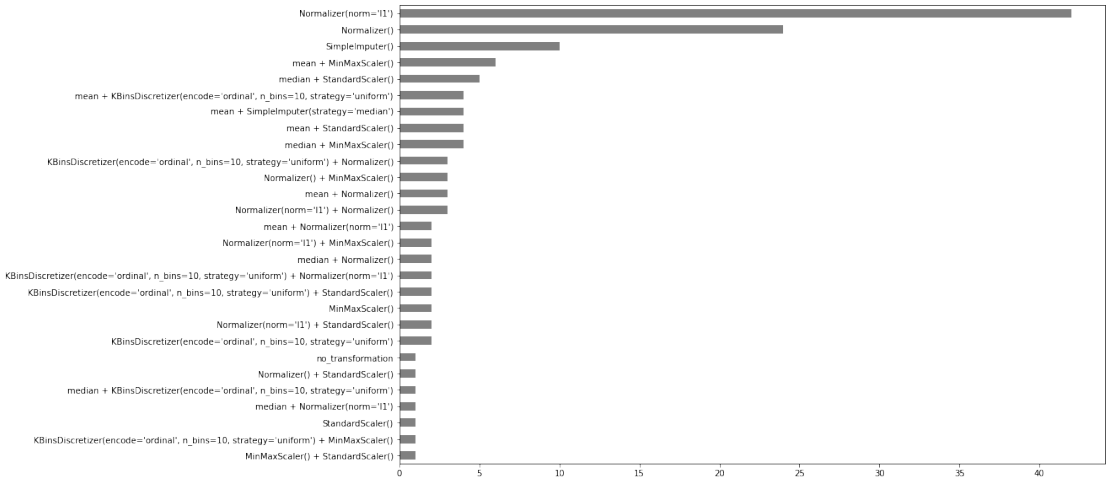

Figure 6. Different transformation's performance

6 shows that among 112 datasets, only one dataset showed to have better performance without any transformations. On the contrary, l1 normalizer showed better clustering results over 40 datasets, followed by the default l2 normalizer showing improved performance on 25 datasets. SimpleImputer with default mean imputing strategy is on the third transformation that showed a better result on over 10 datasets.

In the fourth and onwards, we can see that the combination of two transformations showed a significantly higher multi-CVI correlation score than performing one

transformation. Overall, transforming or data preprocessing yields significantly better results than clustering original data without any preprocessing. This outcome from the knowledge base shows that data preprocessing can improve the quality of the clustering as opposed to unprocessed data.

Figure 7 shows which clustering algorithm performed better in the majority of the datasets. From the figure we can see that Birch and Agglomerative clustering algorithm performed better than K-means in most of the 112 datasets. Birch performed better in 64 datasets while Agglomerative performed better in 50, and K-means performed better in the rest of the 24 datasets. From the offline phase

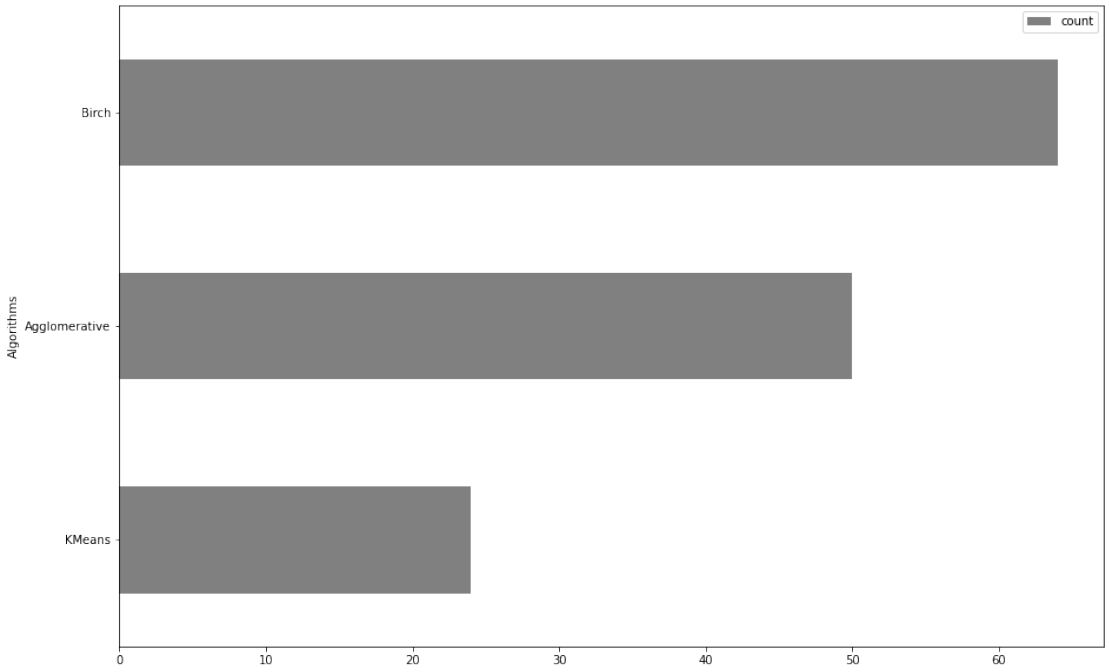

Figure 7. Different transformation's performance

study, it is visible that transformations applied on the dataset lead to better clustering results than the nontransformed ones, which means data preprocessing is an important step for clustering algorithm that leads to significantly better clustering results. Then new test datasets were used to show how the proposed system performs.

Dataset	Algo	Transformation	multiCVI	orgCluster
balance-scale	Agglomerative (n_clusters=4)	MinMaxScaler()	0.80	3
vowel	K-means (n_clusters=16)	SimpleImputer()	0.81	11
arrhythmia	K-means (n_clusters=17)	SimpleImputer()	0.86	13
insect	Agglomerative (n_clusters=3)	SimpleImputer (strategy ='median') +StandardScaler()	0.99	3

Table 8. Proposed method applied on 4 test dataset

6.2 Online Phase

For the online phase study, the proposed methodology was applied on four real-world datasets named balance-scale, vowel, arrhythmia, and insect. These datasets come from the same project from where the datasets for the knowledge base were used. In order to conduct the study, at first, the metadata from the test dataset is extracted, and the three closest datasets are found from the knowledge base for each of the 4 test datasets by applying the nearest neighbor algorithm. Later the clustering method, hyperparameter setting, and transformations are applied to the dataset; based on the suggested pipeline. A rule-based approach is applied for this specific task. After three closest neighbors are selected, the specific cluster setting, including hyperparameters and transformations from the row that has the highest multi_CVI_correlation_score are applied.

Finally, the transformations are applied to the dataset, and the exact clustering configuration setting is applied. Then the multi-CVI correlation score is calculated. Table 8 shows the result of those 4 datasets. The last column in the table, org_cluster, shows the original number of clusters available in the datasets, and the Algo column shows the clustering algorithm applied. The transformations column shows the data preprocessing methods applied to the dataset. The CVI_score column shows the multi CVI correlation score. Although the values it shows are quite close, if not exact.

6.3 Comparison with cSmartML

Finally, we applied cSmartML tool from Elshawi et al., [19] in order to compare if the multi CVI correlation score for clustering improves after data preprocessing.

The datasets went through the data preprocessing methods similar to the offline phase, and the same clustering setting for each dataset was used, including the identical multi CVI combination. This experiment will pave the way for similar automated data preprocessing technique integration with the existing cSmartML tool. The results are shown in two sections for real and synthetic datasets similar to Elshawi et al., [19].

6.3.1 Real Dataset

In order to perform the experiment on the real datasets, first, the datasets went through the mentioned transformations in previous sections. And then, the datasets were clustered using cSmartML tool. The only addition is the prior data preprocessing.
Table 9 shows the comparison of clustering quality after data preprocessing on the real datasets. Where 7 out of 10 dataset shows improved cluster quality. And 3 did not show better result after data preprocessing. But the margin of those 3 datasets where original data performed better is highest at 9% and lowest at 2%.

6.3.2 Artificial Dataset

In order to perform the experiment on the artificial/ synthetic datasets, first, the datasets went through the mentioned transformations in previous sections. And then, the datasets were clustered using cSmartML tool. The only addition is the prior data preprocessing.

And table 10 shows the same comparison against the artificial datasets. And for the artifical dataset, the proposed method shows better results in 3 datasets out of 6 datasets, and for the rest of the three datasets, the margin of difference is highest at 14% and lowest at 1%. Usually, in cases where the original already yields a satisfactory result, data preprocessing does not perform quite well in those cases.
In earlier research, cSmartML showed better results against the other four benchmarks, including default baseline, random search baseline, grid search baseline, and AutoML4Clust. This shows that data preprocessing has a positive impact on unsupervised clustering, which again proves that data preprocessing is an integral part of any data science project, particularly unsupervised clustering. Such techniques could be integrated into the cSmartML tool for an end-to-end pipeline for the user to use.

Dataset	Default	Random	Grid	AutoML 4Clust	cSmartML	Preprocessing +cSmartML
iris	0.77	0.72	0.771	0.74	**0.81**	0.72
glass	0.47	0.45	0.45	0.28	0.48	**0.483**
ecoli	0.62	0.50	0.63	0.63	**0.65**	0.63
iono	0.25	0.23	0.12	0.14	0.30	**0.40**
arrythmia	0.30	0.33	0.30	0.03	**0.43**	0.39
tae	0.11	0.19	0.11	0.01	0.32	**0.59**
thy	0.58	0.50	0.44	0.37	0.58	**0.77**
sonar	0.22	0.32	0.46	0.01	0.39	**0.40**
segment	0.51	0.35	0.59	0.58	0.60	**0.80**
haberman	0.06	0.09	0.08	0.04	0.18	**0.20**

Table 9. Real Dataset comparison with cSmartML[19]

Dataset	Default	Random	Grid	AutoML 4Clust	cSmartML	Preprocessing +cSmartML
aggregation	0.94	0.87	0.55	0.67	**0.97**	0.83
compound	0.81	0.72	0.86	0.59	0.89	**0.93**
jain	0.51	0.41	1.00	0.34	0.46	**0.90**
pathbased	0.75	0.55	0.49	0.46	0.78	**0.84**
3-spiral	0.70	0.41	0.53	0.49	**0.96**	0.90
R15	0.88	0.94	0.95	0.99	**0.99**	0.98

Table 10. Artifical Dataset comparison with [19]

7 Conclusion

The focus of this study was to apply a meta-learning based approach on unsupervised clustering task to suggest the pipeline of the clustering algorithm, hyperparameter setting, and most importantly, the preprocessing of the dataset. This suggestion would help end-user, irrespective of their background, to be able to select the better performing algorithm and preprocessing pipeline. The proposed method leverages meta-learning space to create a knowledge base from which it makes the suggestions. Given the limited study and result, it seems that data preprocessing is important not only for supervised regression tasks it is also important in unsupervised clustering tasks. The result section shows data preprocessing can increase the overall clustering performance. Later, the suggestion from the pipeline was applied to new datasets, and satisfactory results were achieved.

There were a few challenges while conducting this research. First of all, there was a lack of similar studies, especially on the unsupervised learning domain, to extract useful ideas and settings of the research. Another prominent challenge was about the evaluation metrics as it is quite difficult to evaluate cluster quality, and there is no single evaluation metric to rely on. For the thesis, the multi CVI ranking correlation score was considered as an evaluation metric that leverages three cluster validity indices and combines all of the information gained from the three CVIs, and calculates the correlation with the ground truth, which gives a score where maximization becomes the priority. The implementation of the multi CVI ranking is complex and performed with the help of NSGA II sorting algorithm; hence part of the code for this specific case was reused from the earlier study of Elshawi et al., [19]. The goal of this research was to explore a meta-learning method for suggesting automated data preprocessing pipeline. Another challenge was collecting the datasets and dealing with their various structure. Preparation of those 112 datasets required enormous amount of time as each dataset required different processing to bring all of them on an identical structure and data type, to begin with the meta data extraction. The biggest challenge was during the meta-learning space creation. The trade-off between larger hyperparameter space and the significantly increased time to evaluate each of the settings. Therefore, a very limited hyperparameter range was selected in order to deal with all the datasets and algorithms in a limited amount of time.

The scope of the research was limited to reduce the time required to create the knowledge base. In the future, the range of the hyperparameter search could be increased in order to incorporate bigger range, including more datasets which will significantly increase the size of the knowledge base that will later lead to an even better suggestion of a more accurate pipeline. Another future work is to

dynamically choose the multi-CVI combination, which was kept static with the combination of three CVI metrics, Banfeld Raferty score, Davies Bouldin score, and SDbw. But from Elshawi et al., [19] it has been proved that different combinations of those CVI can perform differently on a specific task. In order to avoid complexity, and since the focus of the research was on creating a data preprocessing pipeline, the static combination was used. This specific combination showed better results in the majority of Elshawi et al., [19] dataset result. The list of transformation or data preprocessing techniques could be increased as well, and the combination of more of those transformations could be applied to create the knowledge base. Again, in order to complete the task in the given time, the search space of transformations was also reduced.

The goal of this thesis was to incorporate data preprocessing techniques in the complete AutoML pipeline in a task-specific scenario. For realizing the objective, the thesis investigates the applicability of meta-learning method to create that pipeline on a relatively new domain of unsupervised clustering. From the initial results, it seems promising to apply meta-learning for the data preprocessing pipeline. It will give people from different backgrounds without much knowledge of data preprocessing and ML algorithms to explore their data for new knowledge in terms of clustering. Such data preprocessing components could be also be integrated with existing AutoML tools, such as cSmartML.

References

[1] Olatz Arbelaitz, Ibai Gurrutxaga, Javier Muguerza, Jesús M. Pérez, and Iñigo Perona. An extensive comparative study of cluster validity indices. *Pattern Recognition*, 46(1):243–256, 2013.

[2] Marcel Brun, Chao Sima, Jianping Hua, James Lowey, Brent Carroll, Edward Suh, and Edward R. Dougherty. Model-based evaluation of clustering validation measures. *Pattern Recognition*, 40(3):807–824, 2007.

[3] Usama Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. From data mining to knowledge discovery in databases. *AI Magazine*, 17(3):37, Mar. 1996.

[4] Daniel G Ferrari and Leandro Nunede Castro. Clustering algorithm selection by meta-learning systems: A new distance-based problem characterization and ranking combination methods. In *Information Sciences*, 2015.

[5] Matthias Feurer, Aaron Klein, Jost Eggensperger, Katharina Springenberg, Manuel Blum, and Frank Hutter. Efficient and robust automated machine learning. In *Advances in Neural Information Processing Systems 28 (2015)*, pages 2962–2970, 2015.

[6] Daniel Golovin, Benjamin Solnik, Subhodeep Moitra, Greg Kochanski, John Karro, and D. Sculley. Google vizier: A service for black-box optimization. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, page 1487–1495, New York, NY, USA, 2017. Association for Computing Machinery.

[7] Ron Johnston. Regionalization and classification. In Kimberly Kempf-Leonard, editor, *Encyclopedia of Social Measurement*, pages 337–350. Elsevier, New York, 2005.

[8] Gang-Hoon Kim, Silvana Trimi, and Ji-Hyong Chung. Big-data applications in the government sector. *Commun. ACM*, 57(3):78–85, March 2014.

[9] Tarald O. Kvalseth. Entropy and correlation: Some comments. *IEEE Transactions on Systems, Man, and Cybernetics*, 17(3):517–519, 1987.

[10] Stan Z. Li, Lirong Wu, and Zelin Zang. Consistent representation learning for high dimensional data analysis. *CoRR*, abs/2012.00481, 2020.

[11] Yanchi Liu, Zhongmou Li, Hui Xiong, Xuedong Gao, and Junjie Wu. Understanding of internal clustering validation measures. In *2010 IEEE International Conference on Data Mining*, pages 911–916, 2010.

[12] Hector Mendoza, Aaron Klein, Matthias Feurer, Jost Tobias Springenberg, Matthias Urban, Michael Burkart, Maximilian Dippel, Marius Lindauer, and Frank Hutter. *Towards Automatically-Tuned Deep Neural Networks*, pages 135–149. Springer International Publishing, Cham, 2019.

[13] M. Arthur Munson. A study on the importance of and time spent on different modeling steps. *SIGKDD Explor. Newsl.*, 13(2):65–71, May 2012.

[14] Randal S. Olson and Jason H. Moore. *TPOT: A Tree-Based Pipeline Optimization Tool for Automating Machine Learning*, pages 151–160. Springer International Publishing, Cham, 2019.

[15] Surendra Singh Patel, Navjot Kumar, J. Aswathy, Sai Krishna Vaddadi, S. A. Akbar, and P. C. Panchariya. K-means algorithm: An unsupervised clustering approach using various similarity/dissimilarity measures. In Jennifer S. Raj, Ram Palanisamy, Isidoros Perikos, and Yong Shi, editors, *Intelligent Sustainable Systems*, pages 805–813, Singapore, 2022. Springer Singapore.

[16] Darius Pfitzner, Richard Leibbrandt, and David Powers. Characterization and evaluation of similarity measures for pairs of clusterings. *Knowledge and Information Systems*, 19(3):361, Jul 2008.

[17] Bruno Almeida Pimentel and André C.P.L.F. de Carvalho. A new data characterization for selecting clustering algorithms using meta-learning. *Information Sciences*, 477:203–219, 2019.

[18] Yannis Poulakis, Christos Doulkeridis, and Dimosthenis Kyriazis. Autoclust: A framework for automated clustering based on cluster validity indices. In *2020 IEEE International Conference on Data Mining (ICDM)*, pages 1220–1225, 2020.

[19] Radwa El Shawi. csmartml: A meta learning-based framework for automated selection and hyperparameter tuning for clustering. In *IEEE BigData 2021*, 2021.

[20] C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown. Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms. In *Proc. of KDD-2013*, pages 847–855, 2013.

[21] Dennis Tschechlov, Manuel Fritz, and Holger Schwarz. Automl4clust: Efficient automl for clustering analyses. In Yannis Velegrakis, Demetris Zeinalipour-Yazti, Panos K. Chrysanthis, and Francesco Guerra, editors, *Proceedings of the 24th International Conference on Extending Database Technology, EDBT 2021,*

Nicosia, Cyprus, March 23 - 26, 2021, pages 343–348. OpenProceedings.org, 2021.

[22] Dongkuan Xu and Yingjie Tian. A comprehensive survey of clustering algorithms. *Annals of Data Science*, 2(2):165–193, Jun 2015.

[23] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: A new data clustering algorithm and its applications. *Data Mining and Knowledge Discovery*, 1(2):141–182, Jun 1997.

I. Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, **Hasan Mohammed Tanvir**,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to

   reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   Meta-Learning Based Approach for Automated Pre-processing for Clustering,

   supervised by Dr. Radwa Elshawi.

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Hasan Mohammed Tanvir
22/12/2021