Jestr

Journal of Engineering Science and Technology Review 10 (2) (2017) 51- 64

Research Article

JOURNAL OF
Engineering Science and Technology Review

www.jestr.org

A Study of Algorithm Selection in Data Mining using Meta-Learning

Murchhana Tripathy[1,*] and Anita Panda[2]

[1]Department of CSE, Institute of Technical Education and Research, Siksha 'O' Anusandhan University, Bhubaneswar, Odisha, India
[2]Department of Mathematics, Institute of Technical Education and Research, Siksha 'O' Anusandhan University, Bhubaneswar, Odisha, India

Received 24 November 2016; Accepted 15 March 2017

Abstract

This article discusses the algorithm selection problem in data mining with the help of meta-learning. We present the issue with the help of the classification and clustering problems. In this study, we have analyzed the working of a meta-learning system in connection with the classical algorithm selection problem. Various ranking combination methods available in the literature have been explored from the perspective of the measurement system. Discussion about two new ranking combination methods namely the relative ranking and the percentage ranking have been included. The study also identifies few potential challenges in relation to algorithm selection in data mining using meta-learning.

Keywords: Data mining, Meta-learning, Algorithm selection, Ranking

1. Introduction

The main aim of data mining is to extract knowledge and discover hidden patterns in large databases. Several algorithms are available to accomplish different data mining tasks. Given an application, the critical point is to select an algorithm which will perform better in comparison to others. Meta-learning helps to evaluate the performance of algorithms based on the meta-knowledge extracted from the algorithm learning process. Given a problem **p**, Meta-learning is about learning the behavior of the set of the data mining algorithms **A** to find the suitable algorithms for **p** [9]. The classical Algorithm Selection Problem (ASP) formulated by John Rice, proposes that there exists relationship between a problem's characteristics and the algorithm which can be used to find a solution for it [25]. Later, based on the concepts of the ASP, the No Free Lunch theorem (NFL) was formulated. NFL clearly expresses the fact that there exists no single algorithm, which will produce the optimum output for all the problems [32]. From the work of Rice and the NFL, it is very well understood that the performance of an algorithm varies from problem to problem or more precisely from dataset to dataset. Thus, algorithm selection is an interesting class of problem. This class of problem is considered to be NP-hard in the literature [10]. A decision problem p is said to be NP-hard, when every NP problem q can be reduced to p within a polynomial time [18]. Consider a problem q in the problem space A. Problem q may be very hard to solve, but can be transformed into a mirror problem p in space B. In that case, we can transform the input from space A to space B, solve by an algorithm in B and retransform the solution of B back to A. Now, we get the solution of the original problem q [18]. In other words, we are reducing the problem q to p and using the solution of p to find a solution for q.

2. The Simplified Algorithm Selection Model

John Rice proposed a basic algorithm selection problem model in his work [25]. A simplified version of the same is presented here. The model consists of the following main components.

The problem space P: It consists of a large and diverse collection of problems. It is high-dimensional in nature, because each problem has some independent characteristics which can influence the algorithm performance.
The feature space F: It is a m-dimensional vector space which consists of a number of feature vectors. The choice of the features depends on the problem domain and the selected algorithms [26].
The algorithm space A: It consists of large and diverse set of algorithms. Theoretically there may be millions of algorithms, but practically, one can choose the representatives from a group of algorithms, meant for the same type of task. This set is also high dimensional in nature.
The Performance measure space M: It consists of different measures of performance to evaluate the performance of an algorithm on a dataset. It is a n-dimensional vector space. For example, in case of classification, the most common measures are accuracy, precision, recall, F-score, AUC (Area Under the Curve) etc. and for clustering, internal validity indices are normally preferred.

The selection mapping is a function of features, which selects algorithms based on the feature values. The performance ||Y|| of an algorithm is measured by applying the algorithm in combination with each of the performance measures. Suppose, there are n performance measures, then we get an n-dimensional performance vector Y for one

*E-mail address: murchhanatripathy@gmail.com

algorithm. Taking its norm gives us the actual performance of the algorithm on a problem, where ||Y|| is given by

$$\| Y \| = \left(y_1^2 + y_2^2 + \ldots\ldots + y_n^2 \right)^{1/2}$$

After obtaining the performance of each of the algorithms, the one that maximizes the performance for a dataset can be chosen. As shown in figure 1, the output of the model is a tuple consisting of an algorithm α and its respective performance ||Y||.

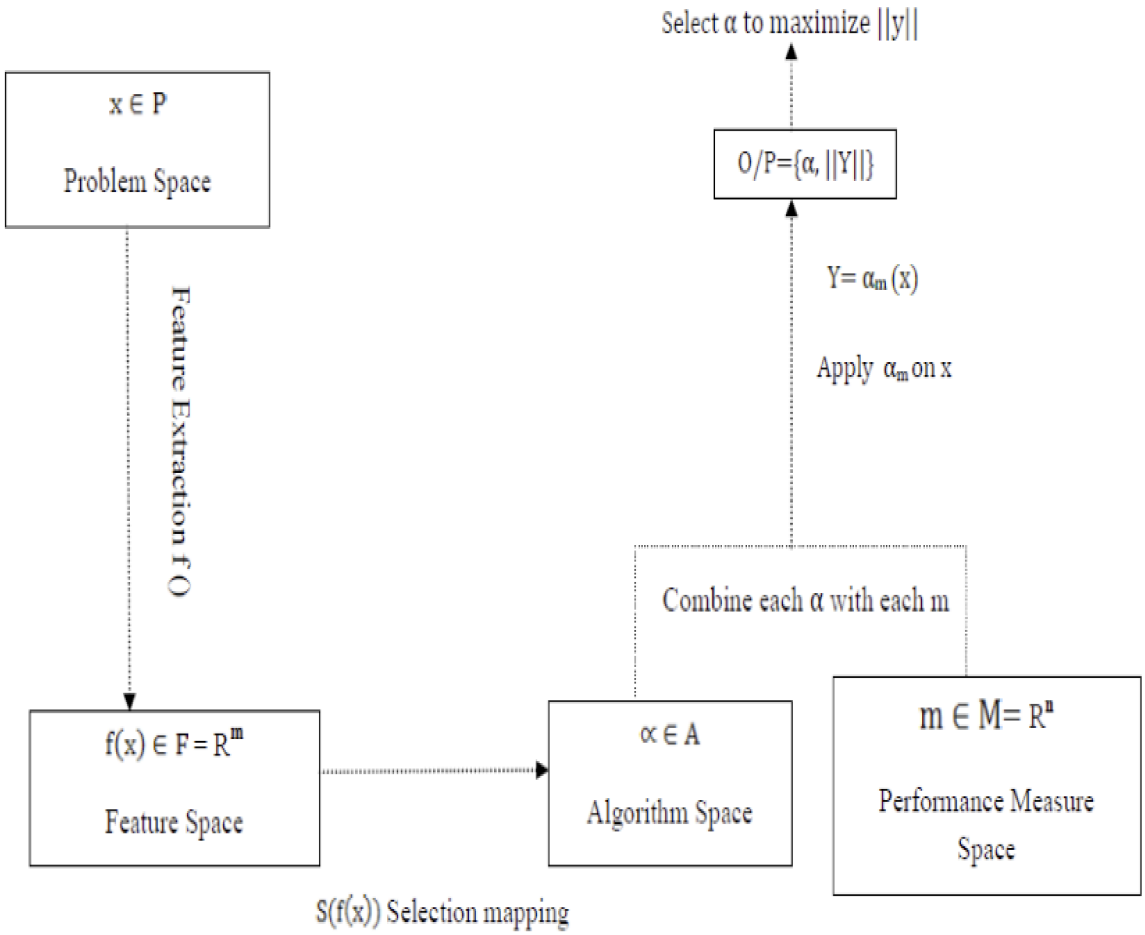

Fig. 1. The simplified ASP model

3. General approaches for Algorithm Selection

Approaches	Description
Brute-force	All the algorithms are run on a given dataset and based on some evaluation criteria, the one producing the optimum output is selected.
User's Choice	User's have preference for one algorithm over others based on certain criteria such as ease of Implementation.
Expert's Opinion	An expert in the area is consulted for selecting a Suitable algorithm. But such expert's advice is not Always available.
Automatic Method	When an user has a set of algorithm and a dataset, a System suggests a suitable algorithm. This leads to the Use of meta-learning in data-mining.

4. The Meta-learning System and its Operation in the light of ASP

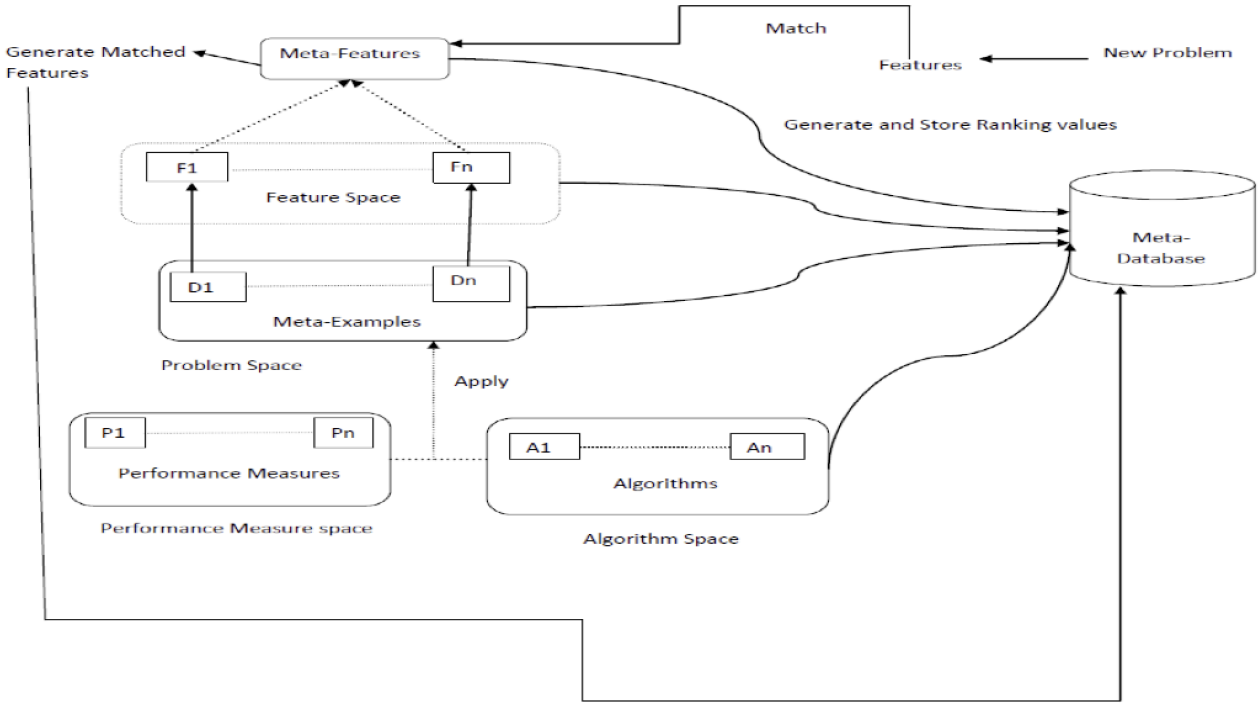

Fig. 2. Basic model of a meta-learning system for algorithm selection

Fig. 2 shows the basic model of a meta-learning system which can be used for algorithm selection. The main component of this system is a meta-database or a meta-knowledge base. It consists of a number of meta-examples, meta-features, a set of algorithms and the ranking of the algorithms obtained after applying them on the problems. D1, D2...Dn represent the meta-examples. In simple terms meta-examples are datasets or problems. So, the meta-examples constitute the problem space. F1, F2. Fn represent the feature sets of D1, D2...Dn respectively and together they represent the feature space. Features common to the meta-examples are known as meta-features [10]. The set of algorithms form the algorithm space and their rankings form the performance space. When a new problem comes, its features are matched with the meta-features in order to find the similar types of problem and their corresponding algorithm ranking. This process works like the 'selection mapping' process of the ASP model. In this case, the selection mapping is done by using a meta-algorithm. A meta-algorithm is a typical learning algorithm such as KNN. After finding the similar problems, the ranking of the algorithms for the new problem is generated by using any ranking aggregation technique.

4.1 Methods for obtaining meta-features

There are two ways in which meta-features can be obtained. They are direct problem characterization and indirect problem characterization. In direct problem characterization, common features of a set of problems are directly used as meta-features. In indirect problem characterization, meta-features are computed based on the values of the original features.

4.1.1 Direct problem characterization

In direct problem characterization, the following types of meta-features are obtained.

Simple: number of Objects, number of features, number of classes, number of categorical features etc..
Statistical: mean, standard deviation, skew, kurtosis, mean absolute correlation etc.
Information theoretic: entropy, noise-to-signal ratio, mutual information etc.

4.1.2 Indirect Problem Characterization

Indirect problem characterization consists of the following three methods.

Model based: In model based method, a model is induced from the dataset and based on that model, the meta-features are computed [2, 31]. e.g- Decision Tree.
Landmarking: Landmarkers are used to quickly estimate the performance of an algorithm on a dataset. There are two approaches for using a land marker.

i) Choose a diverse set of simple learners and estimate their performance on a dataset.
A dataset is characterized by the accuracy of a landmarker [31].

ii) Sub sampling landmarks: Choose a sample from the dataset and estimate the performance of the learner on the sample to find the characteristics of the data [31].

Distance based method: Very recently, in [10], the authors proposed a new indirect problem characterization method based on the Euclidean distance between two objects. In this method, a distance vector is constructed by considering the distance between an object with every other object. Meta-attributes are computed based on the components of the distance vector.

The first two approaches are best suited for classification problems, where the class labels are known. The distance based method is developed to generate meta-attributes for

clustering, where the class labels are unknown and by use of the distance concept retain the unsupervised nature of clustering.

5. Related Work

Kate A. and Smith Miles [26] have given an extensive review of the existing literature in both ASP and Meta-learning for classification. Also they have highlighted the few works done in other areas of data mining such as regression and time-series forecasting. Around 2009-10, the researchers of the meta-learning community started using meta-learning for clustering algorithm selection. In this work, we are going to discuss the algorithm selection issue in the areas of classification and clustering using meta-learning. Discussion about clustering algorithm selection will not be useful, unless classification algorithm selection is discussed, because the idea behind both is the same. Also, clustering tasks use the same set of meta-attributes as used for classification, because till now, no standard set of meta-attributes have been developed for clustering[10]. Another major focus is on the analysis of ranking schemes with respect to measurement systems, because ranking plays an important role in suggesting the suitable algorithms.

5.1 Classification

One of the earliest method for algorithm selection was proposed by Aha [1] which derived rules of the form "algorithm A outperforms B, C and D on the performance measures m_1, m_2,.....m_n for databases with characteristics C_1, C_2,.... C_n". Aha referred to the triplet {Algorithm, performance measures, database characters} as a case study. The method proposed by Aha consists of five steps. The first is to collect the case study details, second is to model the application database, third is to select the performance measures, fourth is to evaluate the selected algorithms on the dataset and the final step is to derive a new rue whenever a performance difference occurs. According to Aha, for simple databases, rules can be generated manually, but rule generating algorithms such as CN2 can be used for more complex studies. Aha conducted the experiments with seventy five instances from the Frey and Slate letter recognition database. He considered seven features such as number of training and test instances, number of target classes, number of prototype per class, number of relevant / irrelevant attributes, range of each attribute value, instance distribution range and prototype distribution range. Aha considered three types of classification algorithms such as IB1, a nearest neighbor classifier, CN2, a rule-based learner and C4, a decision tree. Aha used Classification accuracy as the performance measure, which is defined as the number of test instances correctly classified by a classifier [19]. Aha's experiment produced rules of the following form.

IF ((number of training instances < 737) AND
(number prototypes per class > 5.5) AND
(number of relevant attributes > 8.5) AND
(number of irrelevant attribute< 5.5))
THEN IB1 will be better than C4

StatLog (' The comparative testing of statistical and logical learning algorithms on large scale application to classification') was a project initiated by the European commission which compared the performances of three types of classification algorithms such as symbolic learning, statistical and neural network algorithms [16]. In the StatLog project, experiments were conducted using twenty two large datasets from the UCI machine learning repository, twenty three classification algorithms and sixteen meta-features using accuracy as the performance measures[26]. The experimental results of the project confirmed to the findings of the NFL theorem [32]. Further, the project gave many valuable findings to the meta-learning community, such as the meta-attributes developed during the project is used as a standard for most of the meta-learning tasks. The set of meta-attributes include the simple, statistical and information-theoretic attributes. Also, the idea of running simple algorithms on large datasets instead of putting a lot of effort and calculating the full set of features was first developed in StatLog. Later on, this idea gave rise to the concept of landmarking where the performances of simple algorithms were used to predict the performance of the original set of algorithms.

One of the major goals of applying meta-learning in algorithm selection is to enable end-users to select an appropriate algorithm for their application without running each and every algorithm on the dataests. This idea led to the development of many automated tools such as Mining Mart, Data Mining Advisor (DMA), METALA and Intelligent Discovery Assistant [11]. The description about each one of them is found in [12]. Here, we are going to discuss about the DMA, which first implemented the concept of ranking in algorithm selection. DMA is a web-based system which gives support for algorithm/model (e.g C 5.0 tree algorithm/ Decision Tree model) selection to data mining and machine learning users. It was one of the major deliverable of the METAL project. After Statlog, METAL was the second ESPRIT project. In developing DMA, a direct problem characterization method was adapted and all the statistical and information theoretic attributes were considered. DMA expected all datasets to be in C4.5 decision tree format. In DMA, the algorithm space composed of 10 algorithms and K-NN was used as the meta-learner due to its simplicity and ease of implementation. The developers of DMA considered both accuracy and time as the performance measures. They designed a trade-off for accuracy and time and named it AccD, which had three options in the initial version of DMA.

AccD= 0.1 : Emphasis is on accuracy
AccD= 10 : Emphasis is on time
AccD=1 : A compromise between accuracy and time

DMA implemented two ranking methods, one based on the ratio of accuracy and time and the other based on data envelopment analysis [11]. DMA operated in three steps. First, the input was given and then the dataset was characterized using the K-NN learner. Then the user was prompted to enter the ranking method and the AccD value and after that the ranking of the algorithms was generated. DMA provided three possible treatments for data. At the lowest level, data could be publically accessed by all users and in the highest level only data owner could look at the data, generate ranking and run the algorithms on the data. However, at the mid level, base level data was private and meta-data was public. Also, all users were able to generate ranking, but only data owners could look at the data and run algorithms on it. The use of ranking method in DMA

encouraged the development of many types of ranking schemes and comparison between their results by the meta-learning researcher community.

Kalousis and Theoharis [14] proposed an intelligent assistant for classifier selection named as NOEMON. Their work is based on the idea that choosing a classifier depends on a lot of background information on datasets, algorithms and models and if this job can be delegated to NOEMON which can use the background information and can suggest a classifier, then the analyst's job will be easy. NOEMON has two components, an assistant and a problem solver. The assistant suggests the appropriate classifier upon getting a dataset as input and required instructions from the analyst. For this, the assistant has to access the large knowledge base which is similar to a meta-database. The solver solves the problem and returns the result. Initially, Kalousis et.al. considered only accuracy as the performance measure and conducted the experiments with seventy seven datasets from the university of Irvine repository. They considered forty five features and three algorithms such as IBL(Nearest neighbor , NB(Naïve Bayse) and ID3(decision tree) for their experiments. They implemented a prototype in the following steps.

1. Consider three algorithms: IBL, NB, ID3
2. Create three pairs: IBL-NB, NB-ID3, IBL-ID3
3. Apply the nearest-neighbor classifier to each pair on a dataset
4. Repeat the same for all the 77 datasets
5. Store the inductive models showing the class distribution in the KB and use it for future prediction.

Later on, they proposed a method to consider multiple performance measures such as accuracy, training time, execution time and resource demand and use the DEA(Data Envelopment Analysis) approach which helps to map multiple multidimensional performance values to an 1-dimensional range [0,1]. Also, they claimed that more number of algorithms have to be considered in future. Further they proposed that a correlation between the continuous and the discrete attributes have to be established in their work.

Brazdil and Soares [3] have given a comparison of three ranking methods: average ranks (AR), success rate ratios (SRR) and significant wins (SW), which are defined and explained in section 6.1. They conducted the experiments using six algorithms and sixteen datasets using accuracy as the evaluation measure. In this process, the three ranking methods produced the similar ranking results. Next, to measure the quality of the ranking results, they were compared with the ideal ranking. An ideal ranking is based on the actual performances of the algorithms on a test dataset. A ranking method R1 is said to be better than another method R2, if the difference between R1 and the ideal ranking is less than the difference between R2 and the ideal ranking and this comparison is done by using the SRC (Spearman's rank correlation coefficient) and a n-fold cross-validation. In order to determine the difference in performance between ranking methods, Brazdil and Soares used the Friedman's hypothesis test and Dunn's multiple comparison test. This phase of their experiment showed that SRR and AR were better than SW. Followed by the experimental results, they also presented a theoretical analysis of why the results produced by the three ranking methods were different. They showed that the method that exploited more information about the magnitude of the difference between algorithm performances produced better result. They also claimed that there is a necessity to conduct significance test and tests for analyzing the robustness of the methods, where datasets with large outliers are involved. Followed by their previous work, Brazdil and Soares proposed the zoomed ranking method [4]. In zoomed ranking, the algorithm selection problem is divided into two phases. The first one is zooming, which identifies a subset of relevant datasets by applying KNN to the problem space. The second is ranking, which is built by using the Adjusted Ratio of Ratios (ARR). The ARR is defined as the ratio of success rate ratio and adjusted time ratio of two algorithms on a dataset.

Peng et al. [20] used a decision tree based problem-characterization method. They used a C5.0 decision tree classifier to generate fifteen meta-attributes. The meta-attributes were tree width, tree height, number of nodes, number of leaves, maximum and minimum number of nodes at one level, mean and standard deviation of nodes and leaves at one level, the length of the longest and the shortest branch, mean and standard deviation of the length of the branch, maximum and minimum occurrence of attributes and mean and standard deviation of the occurrences of attributes. A meta database was constructed by using the meta-attributes generated by the DCT (Data Characterization Tool) method and previously analyzed performances of the algorithms using accuracy and time as performance measures. In their experiment, they used three sets of meta-attributes to produce the ranking of the algorithms with a zoomed ranking approach. The decision tree based method is an indirect problem characterization method. They compared the performance of this method with a direct problem characterization method, the DCT and with another indirect method, landmarking. The DCT method used 25 meta-attributes and the landmarking approach used 5 meta-attributes. Peng et.al. used forty seven datasets from the UCI machine learning repository and 10 learning algorithms for experiment. They considered the error rate and time as the performance measures and used a 10-fold cross validation method. Also they applied a manual feature selection to all the three methods and compared their performances. The result showed that the decision tree based method outperformed the other two techniques in both the cases i.e with feature selection and without feature selection.

In [30], Vilalta and Drissi have discussed about a self-adaptive learning algorithm which improves the learning bias dynamically. Learning bias or inductive bias is defined as the set of assumptions that the learning algorithm makes to predict the output of an unseen example. For example, in the K-NN algorithm, the inductive bias is the assumption that a new instance will belong to a class of instances which are at a minimum Euclidean distance from it. Given a space of tasks, due to the presence of the inductive bias any base learning algorithm prefers to learn only on those tasks which support the bias and thus they never learn over the entire space. Basically, this idea led to the development of a self-adaptive learning algorithm by Vilalta and Drissi. They divided the space of all tasks into two regions: random and structured, according to a measure of complexity K. The definition and explanation for K can be found in [30]. They defined a task as a structured task, which show regular patterns over the data and lead to valuable knowledge representation. A task was defined as random if it had many irregularities and long representations were required for the retrieval of the original data. They focused only on the

structured tasks because failing to learn over random tasks did not affect the outcome of learning. The universe of all structured task was divided into a number of regions R_{L1}, R_{L2}.....$R_{Ln.}$ Then, they proposed that meta-learning could be used to determine the properties of a task in a region R_{Li} that made L suitable for such a region and also the properties of an algorithm L_i, which caused its dominance over R_{Li}. They used these ideas to formulate the model of a self-adaptive learner which changed its bias according to the characteristics of a task. According to their proposal, the algorithm initially started with a fixed bias to predict the target variable and as more tasks were added the bias was adjusted according to the characteristics of the tasks and this process continued till the algorithm covered the space of the structured tasks completely. One potential drawback of the self-adaptive learner was that the meta-learner used a fixed bias and to make the meta-learner self-adaptive, a meta-meta-learner could be used. But, this meta-learner also exhibited a fixed bias and to make it self-adaptive, another meta-learner needs to be used and thus the chain continued. As identified by Vilalta and Drissi, a challenge in meta-learning tasks is to "achieve a complete flexibility in the selection of bias".

In [31], Vilalta et.al. have given an extensive review of different techniques required to build a meta-learning system and proposed a meta-learning architecture. They discussed about many potential techniques such as problem characterization methods, methods to map the input datasets to an output model, techniques to combine base classifiers to improve classification performance, inductive transfer which deals with sharing of meta-knowledge across domains and dynamic bias selection. They gave a vivid explanation of all these techniques and their use to realize a meta-learning system. Further, they discussed about the meta-learning tools required for data mining and few applications. Overall, their contribution gives a lot of information to the beginners in the meta-learning field.

The survey article by Bhatt et al. [2] gave an overview of different techniques used for algorithm selection such as zoomed ranking, landmarking, active meta-learning, clustering based meta-learning, resampling based ensemble methods etc. to name a few. They presented a meta-learning architecture based on the architecture proposed by Villta et.al [31]. They also conducted an experiment considering seven datasets, seven classification algorithms and two evaluation measures such as accuracy and mean absolute error. Basically, they tried to study the impact of dataset characteristics on the performance of a given classifier on a particular dataset and for that they considered a set of simple and statistical attributes along with attributes such as presence of noise and outliers. Based on the experimental results, they generated a ranking of the algorithms using the ARR (Adjusted Ratio of Ratios). Their approach gives an idea about how impact of dataset characteristics can be studied on classifier performance. Although the experimental set up used in the study was small, the extension of the same with considerably large number of algorithms and more number of datasets can help in building useful meta-knowledge which may lead to the investigation of the most and the least influential characteristics. Further Bhatt et. al have highlighted about some of the research challenges associated with meta-learning.

Kim and Hong [15] used the meta-learning approach to create a multi classifier system in order to improve classifier's performance. In order to apply meta-learning, they generated the base classifiers by executing the mean-field genetic algorithm multiple times on the training dataset. In their model, a base classifier was formed with two components, a general classifier and a meta-classifier. The general classifier performed the normal classification task and the meta-classifier evaluated the classification result of the general classifier. First, a general classifier (GC) was learned from a training dataset denoted by TGC. Each example in the TGC was formed with a set of attributes and a class label. Next, a meta-classifier (MC) was trained for a training dataset denoted as TMC. To construct TMC, first the training examples in TGC were classified with GC. For each example, the prediction of GC was classified either as 0 or 1, where 0 represented an incorrect decision and 1 represented a correct decision. Then the TMC was built in which an element was a triplet (Attribute value of an example in TGC, PGC, and CPGC). PGC represented the prediction of GC and CPGC represented the classification of PGC. For any given unknown problem, if MC classified prediction of GC as correct, the base classifier participated in the final decision making process. Experiments were performed using the Glass dataset and the Soybean dataset. Kim and Hong indicated an improvement in performance with their approach, but further analysis of the work shows that building a separate meta-classifier training dataset or TMC for each dataset is a time consuming and complicated task. In meta-learning, we want to avoid the execution of the algorithms on the new datasets and wish to find a suitable algorithm for it. If we stick to this idea and try to use a multi-classifier approach, then we may obtain improved performance with less complexity.

Reif et al. [24] used a regression approach for computing the performances of the classifiers independently. They built the meta-knowledge with the meta-features computed on a number of datasets and the actual performance of the target classifier on the training data. Using this meta-knowledge a regression model was learned for each classifier. To find a suitable classifier for a new dataset, the meta-features of the dataset were first computed and the previously learnt regression models were applied on these feature values. They identified that a possible drawback of the regression approach was the use of a regression model for each target algorithm. However, this was accompanied with several benefits such as the ease of the addition and deletion of classifiers and the training and evaluation of the individual classifiers. Reif et.al. also experimented with the wrapper feature selection approach and the performance of the selected feature set was determined by the leave-one-out cross-validation of the resulting regression model. They evaluated the performance of the meta-learner using root mean squared error and pearson product moment correlation coefficient. For experimental analysis, they considered fifty four datasets from the UCI machine learning repository and the Statlog. They chose forty three meta-features from all the five categories and selected nine classifiers including few like SVM which were investigated rarely in previous works. They contributed in two main aspects of meta-learning for algorithm selection. First, they considered all the five types of state-of-the-art met features namely simple, statistical, information theoretic, model-based and landmarking to find their applicability in measuring the classification accuracy. Second, they integrated the proposed model as an open source software wizard with RapidMiner, a data mining tool. The aim was to help non-expert users in classifier selection.

Prudˆencio et. al.[22] took the 'application of meta-learning in the field of data mining algorithm selection' one step further by incorporating the idea of active learning in it.

Active learning is a paradigm of machine learning in which the learner decides which data to be included in the training set. The motivation to apply active learning is to reduce the number of training examples without compromising with the performance of the algorithm applied on the dataset or even try to improve the algorithm performance. We know that, meta-learning requires sufficient number of meta-examples which are generated by executing the candidate algorithms on the datasets and this is a very time consuming task. In such a scenario, if active meta-learning (meta-learning with active-learning) could be used, then the performance of the meta-learning system would improve because it would require less number of meta-examples and thereby less number of evaluations of the algorithms. Prudˆencio et. al. provided an architecture for active meta-learning which consists of three phases, the meta-example generation phase, the training phase and the use phase. In the meta-examples generation phase, the active learning module selects the most relevant problems for the meta-learning task based on a predefined criteria which is further based on the meta-features and the present knowledge of the meta-learner. Then, the candidate algorithms are executed on the selected datasets to generate meta-examples. In the training phase, the meta-learner acquires knowledge from the meta-database and the associated meta-features with the performance of the algorithms. In the use-phase, the performance of the algorithms, for a new problem is found out. Prudˆencio et. al considered K-NN as the meta-learner and implemented a prototype by considering two uncertainty based active-learning methods. The first method is based on the classification uncertainty of the K-NN algorithm which is defined as the ratio of the distance between the unlabeled examples and the nearest labeled examples to the sum of the distances between the unlabeled example and its nearest labeled neighbors of different classes. After computing the uncertainty of each of the problems in the problem space, the one with the highest uncertainty is chosen as the meta-example. The second method uses the concept of entropy to define classification uncertainty. Here, it was assumed that the K-NN could predict a probability distribution over all possible class values for a given problem. In this method also, the active learning module selected the problem with highest uncertainty. They evaluated the prototype for the algorithm selection problem in time-series forecasting and for predicting the performance of multi layer perceptron networks for regression problems. In both the experiments, they found their method to be a better method than any random method which would generate meta-examples. Further to extend this proposal, they used the concepts of this work along with datasetoids to generate meta-examples, which is presented next.

The success of meta-learning depends on the availability of sufficient number of meta-examples which are normally generated by running the candidate algorithms available in the algorithm space on the datasets. As described by [12], a meta-examples is a tuple of the form < f(x); t(x) >, where f(x) represents the feature vector for the dataset x and t(x) represents a target variable for x. If we are considering ranking of algorithms, then t(x) represents the specific rank of an algorithm on x. Generating meta-examples is computationally very expensive [12] and also we need sufficient number of datasets for it. Prudˆencio et.al. [23] proposed an effective method for generating meta-examples . They used the active meta-learning technique on datasetoids for the same**.** A datasetoid is a new dataset generated from the original dataset by replacing one of its independent variable with the target variable and vice-versa [23]. Normally nominal attributes which are also referred to as symbolic attributes are chosen for replacement. Following this method of attribute replacement, a sufficient number of datasetoids were generated and then active meta-learning was applied. The input for the active meta-learning module was a set of labeled examples and a set of unlabeled examples. The module selected unlabeled examples one after the other on the basis of the meta-features of the labeled problems and the current set of labeled examples. Then the candidate algorithms were evaluated on the selected unlabeled example and the best performing algorithm became the label of the unlabeled meta-example. The process continued till some stopping criterion was met. For experiment, Prudˆencio et.al selected 64 algorithms from the UCI machine learning repository and generated 983 datasetoids. Further, they selected only two meta-attributes i.e the class entropy and average entropy of the attributes. This was a very novel approach in the meta-learning research field, but there are certain issues on which focus can be given. The first is the data characterization aspect which is very important in any meta-learning applications. Instead of going for only two meta-attributes, more number of meta-attributes could have been considered covering a wide variety of structural properties of the data. Secondly, Prudˆencio et.al considered accuracy as the only performance measure. Consideration of other measures along with could have produced more promising results. The third issue was, out of the 64 original datasets, few were chosen as the test set and the ones those were chosen for the same, their datasetoid were not chosen as the training examples, because an original dataset and its corresponding datasetoid did not generate a different meta-example. This significantly reduced the number of meta-examples and finally a leave-one-out cross-validation was needed like any other traditional meta-learning approach in data-mining. However, their approach addressed two of the major issues of meta-learning for algorithm selection and they are: generation of meta-examples and reducing the cost of producing meta-data. Focusing on the above mentioned issues may further enhance the result.

5.2 Clustering

In real world clustering problems, the labels of the objects are not known. Hence it is really important to recommend clustering algorithms in advance for new unseen datasets. Also, the significance as well as the difficulty in applying meta-learning techniques for clustering is quite higher than that of classification problems due to its unsupervised nature. The study of literature for data mining algorithm selection using meta-learning reveals that there has been a lot of work in the field and eventually different directions of the problem has been explored. But we find the reference to a small number of works in clustering. In this section, briefly we are going to discuss few of them.

De Souto et al. [7] proposed a very simple architecture to apply meta-learning for clustering algorithm selection which resembles to that of the general architecture for classification algorithms. Further, they highlighted the importance of the choice of the dataset, the selection of the clustering algorithms, the choice of the meta-attributes and the choice of the meta-learner for clustering algorithm selection. In their experiment, they used the cancer microarray gene expression dataset, seven clustering algorithms, eight meta-attributes and the regression Support Vector Machine (SVM) as the meta-learner. In order to generate the ranking of n

number of clustering algorithms, they used n regressors. Each regressor predicted the performance of a specific algorithm for the input dataset. Upon arrival of a new dataset, a supervised learning algorithm was applied to each of the n regressors, which associated the dataset to a ranking value. The corrected rand index (cR) was used to measure the similarity between the partitions produced by the clustering algorithms and the actual number of classes present in the data. The range of cR is [-1, 1] with 1 indicating exactly similar partitions and value near 0 or -1 indicating cluster agreement found by chance. They used the Spearman's rank correlation coefficient to measure the similarity between the ideal ranking and the recommended ranking. The larger is the value of SRC; the grater is the similarity between the ideal ranking and the recommended ranking. This was one of the first attempts to adapt a meta-learning approach for clustering. De Souto et.al proposed that other meta-learners could be tried for the same experiment. We find that the meta-attributes used were from the set of standard meta-attributes for classification task because no standard set for clustering is available as of now. Since, classification and clustering differ completely by their nature, an attempt to devise a new set of meta-attributes to characterize the clustering problems may enhance the performance.

Soares et al. [27] proposed a clustering algorithm recommendation technique for artificial datasets. By using the Gaussian cluster generator and the Ellipsoidal cluster generator, they generated hundred and sixty datasets [13]. In order to produce the recommended ranking, they evaluated nine clustering algorithms using the global error rate measure. Global error corresponds to the proportion of examples that falls outside the cluster corresponding to the actual class. Two algorithms were considered to perform in a similar manner if the difference between their global error rates was less than 0.01. Two meta-learners namely the MLP and the SVR were used and two sets of meta-features are used for the experiments. One set of meta-features were based on the statistics extracted directly from the data and another set was calculated from an one-dimensional array obtained by applying Hotelling's T^2 vector statistics. The algorithm ranking was formulated by ascending order of global error rate using both MLP and SVR and was compared with the ideal ranking using Spearman's rank correlation coefficient. One important finding of this work is the generation of artificial datasets to build meta-examples. The success of the meta-learning system depends on the availability of sufficient number of meta-examples and this work explains the methods to achieve that.

Ferrari and de Castro [8] experimented with thirty problems from the UCI machine learning repository to evaluate the performance of five base level clustering algorithms such as K-Means, Single-Linkage (SL), Complete-Linkage (CL), Medium Linkage (ML) and Self Organizing Map (SOM). They selected 10 attributes from the standard meta-attributes set used for classification problems and their specialty was that they were all independent of the class labels. Attributes of unlabeled objects were chosen, because in clustering object labels are not known initially. They used an external evaluation measure named FBCubed to evaluate the quality of the clusters. An external measure evaluates the generated clusters with respect to known labels. FBCubed is based on the F-measure and the BCubed metric. The Bcubed metric is based on precision and recall values of the retrieved information. The meta-attributes were normalized in the range [0, 1]. Ferrari and de Castro chose three meta-learners, the K-Nearest Neighbor (an instance based learner), the Multi-layer Perception (a neural network) and the CART (a decision tree) and estimated their performances i.e the quality of clusters produced by them by using the Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE). Their experiment showed K-NN to be a better learner. This work investigated the feasibility of applying meta-learning to clustering and claimed that the use of internal evaluation measures may improve the performance of the system. However, only three varity of algorithms were tested, one partitional algorithm (K-Means), one artificial neural network (SOM) and three hierarchical clustering algorithms (SL, CL, ML) and later Ferrari and De Castro extended this work, where they considered a larger varity of algorithms [10].

Further, Ferrari and de Castro [9] extended this work to propose a ranking scheme for clustering algorithms. The experimental set-up was exactly same as that of [8] except that they considered another meta-algorithm the naïve bayse. They constructed the ranking of the five base-level algorithms as used in [8] with the help of each individual meta-learner and by considering one dataset at a time. The final ranking was built for individual dataset, by considering the FBC values of the five algorithms in a descending order. The ranking result produced by them was evaluated by using the Spearman's rank correlation coefficient (SRC). The SRC compared the ideal ranking with the recommended ranking. From the comparison, it was found that K-Nearest Neighbor produced better result. A general observation in this work is that there does not exist any significant difference between the FBC values of two algorithms and still they are assigned with different ranking. This idea was further extended by Ferrari and de Castro to generate the ranking of a variety of clustering algorithms with the help of internal evaluation measures [10].

Ferrari and de Castro [10] proposed an algorithm selection method for seven clustering algorithms from different categories such as partitional (K-means and K-harmonic means), one graph-based (minimum spanning tree), one density-based (DBSCAN), one hierarchical (SL) and two bio-inspired clustering algorithms (Evolutionary algorithm for clustering and Particle swarm clustering) using ten internal evaluation measures. Internal measures evaluate the clusters by considering information intrinsic to the data [8] and the aim of using internal measures was to preserve the unsupervised nature of clustering. For experiment, they selected eighty four problems from the UCI machine learning repository covering a wide variety of domains. In this work, they proposed the distance based ;problem characterization method and claimed that it would be most suitable for clustering, because this distance measure only relied upon the Euclidean distance between objects and did not need any class label information. However, they experimented with three sets of meta-attributes such as object based meta-attributes, distance-based meta-attributes and a mixture of both. Given a new problem, to identify the problems similar to it they used the K-NN classifier because K-NN proved to be suitable in their earlier experiments [8,9]. To provide a ranking of the algorithms for a particular dataset, they used three ranking combination methods such as average ranking, score ranking and winner ranking and compared their outputs with the output of the ideal ranking method with the help of SRC. The three ranking methods and the ideal ranking method used in this work are mostly used by meta-learning researchers and thus we provide the

working and explanation about them with the help of an example in section 6**.** This was the first major work done in the field of clustering for automatic algorithm selection and Ferrari and de Castro indicated that a number of future research directions such as extraction of other meta-attributes from the distance vector or the use of other ranking scheme may enhance the performance.

Lee and Olafsson [17] proposed a new cluster evaluation measure called disconnectivity. They used the concepts of cluster compactness and disconnectivity to find the natural clusters in a dataset which was independent of the data distribution. Cluster compactness is based on the intra-cluster distance which has to be minimized and the inter-cluster distance which has to be maximized. Disconnectivity is based on two concepts namely cluster K-nearest neighbor consistency and cluster K-Mutual Nearest neighbor consistency. The cluster K-NN consistency states that, for any data object in a cluster, its K nearest neighbor should also be in the same cluster. The cluster K mutual nearest neighbor states that if the nearest neighbor of object a is object b and the nearest neighbor of object b is object c, then a and c are mutual nearest neighbors. That means, there exists a transitive relationship between objects a , b and c. Extending this concept to a set of objects, if a is in the set of m nearest neighbor of object b and b is in the set of n nearest neighbor of object a and k=max{m, n}, then a is in the set of k mutual nearest neighbor of b and vice versa. The appropriate number of clusters is found from the disconnectivity plot. The experiment was conducted with thirteen artificially generated dataset with varying range of cluster density and they compared the outcome with the output of other eight internal evaluation measures. They found that their proposed internal evaluation measure was able to predict the actual number of clusters in most of the cases, though it was not the best every time. To show the applicability of their method, they also experimented with two real datasets, the Wine dataset and the Wisconsin breast cancer dataset and were able to discover the appropriate number of clusters. This study aimed to design a new improved cluster evaluation measure which could be applied in meta-learning and this was done successfully. However, there exists a number of cluster evaluation measures other than the eight considered in this work. Further comparative study can be done with more number of internal measures and with more number of real datasets which may lead to a robust internal measure. Further, the applicability of the new evaluation measure 'disconnectivity' in the clustering algorithm selection problem can be studied.

6. Techniques for algorithm recommendation

Basically there are four different types of algorithm recommendation methods as discussed in [5] and they are as follows.

1. The best algorithm in a set
2. A subset of algorithms
3. Ranking of algorithms
    1. Linear and complete ranking
    2. Weak and complete ranking
    3. Linear and incomplete ranking
    4. Nonlinear and complete ranking
4. Performance estimation of algorithms

The best algorithm in a set: Given a set of algorithms, 'The best algorithm in a set' technique recommends a single algorithm that is expected to perform the best. The disadvantage of this kind of recommendation scheme is that when the recommended algorithm fails, the user is left with no other choice. Also there is no guarantee that the algorithm recommended is truly the best one.

A subset of algorithms: Given a set of algorithms, this method recommends a subset of algorithms which perform better than other algorithms on a dataset. The notion of 'performing better' on a given dataset can be defined with the help of a threshold value or a statistical test can be carried out to compare the performances of algorithms. This recommendation technique provides users with more than one choice of the algorithms, but it is provided as an unordered subset of the original set of alternatives. Therefore there is no guidance on which algorithm to try first, which to next and so on.

Ranking of algorithms: Given a set of algorithms, the user is provided with an ordered set of algorithms. There are different ranking schemes as explained below.

1. Linear and complete ranking: Linear ranking means ranks are different for all algorithms. Complete ranking means all algorithms have their rankings defined. Ties are not allowed. But in real life, possibility of a tie exists, so this kind of ranking scheme is not suitable for all algorithm recommendation applications.

2. Weak and complete ranking: Weak ranking is used when the performances of two or more algorithms is not significantly different. The definition of complete ranking remains the same as defined above. In weak ranking, ties are allowed.

3. Linear and incomplete ranking: When the metalearning method is unable to provide a recommendation for all the algorithms, then that refers to incomplete ranking. The definition for linear ranking remains the same.

4. Nonlinear and complete ranking: In nonlinear and complete ranking, ties in algorithm ranking are allowed and ranks are defined for all algorithms.

Performance estimation of algorithms: The performance of an algorithm can be estimated based on some fixed target such as higher accuracy or less running time in predicting the correct class label in case of classification problems or accuracy in predicting the actual number of natural clusters in case of a clustering problem.

Determining the rank of Learning Algorithms for a new problem

The rank of an algorithm can be determined by following steps.

1. A new problem is presented to the meta-learning system.
2. Meta-attributes of the new problem are extracted.
3. A classification algorithm is applied to find the most similar problems to the new problem.
4. For each of the similar problems, the identified algorithms are applied and their performances are

measured with the help of the chosen evaluation measures.

5. The outcome is an (m×n) matrix which is quite large. Choosing the suitable algorithm(s) for a new problem from such large data is again difficult.
6. So a ranking combination method is applied to determine the ranking of one problem.
7. Step 1-6 is repeated for all the similar problems.
8. To determine a ranking of algorithms for the new problem, a ranking aggregation-technique is applied to the ranking obtained for all the problems.

The advantage of using ranking methods is that, it provides users with an ordered set of algorithms and the users can follow that order in the experiment. In this section, we are going to discuss about various ranking combination methods used in the meta-learning literature for algorithm selection from the perspective of a measurement system. The term 'measurement' has been defined in various ways in the literature. The Mathematical theories of measurement [29] view measurement as the mapping of qualitative empirical relations to relations among numbers or other mathematical entities. A measurement system formulates rules to establish this mapping and a measurement scale is a way to measure the variables. There are four types of measurement scales namely nominal, ordinal, interval and ratio [6]. These scales are formulated based on the following four properties of measurement system.

1. Unique Identity: Each object being measured is identified by a unique class. A class is a collection of objects having some common characteristics and each such class is a mutually exclusive and mutually exhaustive group.
2. Magnitude: Values on the measurement scale has an order relationship to one another. The values are represented by real numbers and real numbers satisfy the order completeness property.
3. Equal interval: The equal interval property is based on the isometry or distance preserving property between two points in two metric spaces. Let U and V be two metric spaces with metrics d_u and d_v. A map f: U→V is called an isometry if for any $x, y \in U$, $d_v(f(x), f(y)) = d_u(x, y)$.
4. Existence of a true zero: The scale has a true zero, which is the minimum value on the scale and below that no value exists.

A nominal scale satisfies the first property only. An ordinal scale satisfies the first two properties but does not show any further mathematical structure such as equal interval or the ratio property. An interval scale satisfies the first three properties. Using an interval scale we can find whether there exists equal or unequal interval among objects being measured on the scale. A ratio scale satisfies all the four properties and due to the existence of a true zero comparison between two objects is possible.

6.1 Ranking combination methods used in meta-learning literature in the context of algorithm selection

We find different ranking combination methods such as average ranking, score ranking and winner ranking. Throughout the literature, the results obtained by them are compared with ideal ranking in order to choose the best possible ranking for a set of algorithms. Along with these existing methods, we are also going to discuss two more ranking methods namely relative ranking and percentage ranking which are based on the concepts of percentile and percentage. Table 1 gives a sample calculation of all the ranking methods mentioned above, where we have assigned ranking to seven algorithms A_1, A_2... A_7 using ten evaluation measures M_1, M_2... M10. In score ranking, unique scores are assigned for each rank position. Table 2 shows the assignment of score values as done in [10].

Table 1. Sample calculation for ranking combination methods

Evaluation Measures	Algorithms						
	A_1	A_2	A_3	A_4	A_5	A_6	A_7
M_1	6	5	3	1.5	4	1.5	7
M_2	6	5	4	1.5	3	1.5	7
M_3	7	5	6	1.5	4	1.5	3
M_4	4	3	1	5.5	2	5.5	7
M_5	4	2	1	5.5	3	5.5	7
M_6	6	5	4	1.5	3	1.5	7
M_7	4	2	3	5.5	1	5.5	7
M_8	6	5	3	1.5	4	1.5	7
M_9	6	3	1	4.5	2	4.5	7
M_{10}	4	2	3	5.5	1	5.5	7
Sum of rankings	53	37	29	34	27	34	66
Mean rank position value	5.3	3.7	2.9	3.4	2.7	3.4	6.6
Average Ranking	6	5	2	3.5	1	3.5	7
Total Score	27	51	64	58.5	66	58.5	15
Score Ranking	6	5	2	3.5	1	3.5	7
Number of victories	17	33	41	36	43	36	4
Winner Ranking	6	5	2	3.5	1	3.5	7
Deviation	17	33	41	36	43	36	4
Deviation Ranking	6	5	2	3.5	1	3.5	7
Relative Rank percentage values	85.71	71.42	28.57	57.14	14.28	57.14	100
Relative Ranking	6	5	2	3.5	1	3.5	7

Table 2, Number of points for each rank position in score ranking

Rank Positions	1	2	3	4	5	6	7
scores assigned	10	8	6	4	3	2	1

6.1.1 Average Ranking:
Average ranking is defined by the following formula.

$$\bar{r}_i = \frac{1}{k}\sum_{j=1}^{k} r_{ji} \quad (1)$$

Where $\bar{r}_i$ is the mean rank position value and r_{ji} represents the rank for the i-th algorithm with respect to the j-th evaluation measure.

Average ranking satisfies the i) identity property because each algorithm is identified by one and only one class and an unique class is identified by an unique mean rank position value. A tie occurs when the sum of rankings obtained by two algorithms is same. In this case, the ranks get distributed. It satisfies the ii) magnitude property, because there exists an ordered relationship among the rankings assigned to different algorithms. Average ranking does not satisfy the iii) equal interval property because it can be observed from table 1 that, the distance between the x-th rank position and the y-th rank position is not preserved in the distance or difference between the 'sum of ranking' obtained by the algorithms for all the seven examples. Obviously, it does not satisfy the iv) ratio property. Because in order to satisfy the ratio property, one has to satisfy the 'equal interval' property.

6.1.2 Score Ranking
In score ranking, each algorithm receives some scores according to their rank position as shown in table 2. The scores obtained by an algorithm, for each of the evaluation measure are totaled. This total score is considered in a descending order for building the final ranking of the algorithms. Score ranking satisfies the i) identity property because each algorithm is identified by one and only one class and an unique class is identified by an unique total score. In case of a tie, the ranks between two algorithms get distributed and they belong to the same class. It satisfies the ii) magnitude property, because based on the scores obtained, the algorithms are ranked and there exists an ordered relationship among them. Score ranking does not satisfy the iii) equal interval property. This can be seen from the table 1, where we find that the distance between the x-th rank position and the y-th rank position is not preserved in the distance between the 'scores' obtained by the algorithms. It obviously does not satisfy the iv) ratio property as an implication of not satisfying the 'equal interval' property. Another drawback of score ranking is that scores are assigned to rank positions (table 2) according to user's assumption. There is no standard rule defined for that.

6.1.3 Winner Ranking
The winner ranking method is based on the number of victories obtained by an algorithm in a pair wise competition between algorithms with respect to all the evaluation measures. It is based on the following formula.

$$v_i = \sum_{j=1}^{m}\left(|r| - p_{ij}\right) \quad (2)$$

Where v_i is the number of victories of the ith algorithm, m is the total number of evaluation measures, |r| is the number of algorithms in the ranking and p_{ij} is the rank position for the i-th algorithm with respect to the the j-th evaluation measure. The final ranking is built based on the descending order of the number of victories.

Winner ranking satisfies the i) identity property because each algorithm is identified by one and only one class and each unique class is identified by an unique 'number of victories' value. A tie occurs when two algorithms obtain the same number of victories. In that case, their ranks get distributed and they belong to the same class. It satisfies the ii) magnitude property, because their exists an ordered relationship between the winning positions or ranks of the algorithms. Winner ranking does not satisfy the iii) equal interval property. This can be seen from the example shown in table 1, where we find that distance between the x-th rank position and the y-th rank position is not preserved in the difference between the 'number of victories' obtained by the algorithms. It does not either satisfy the iv) ratio property because of failing to satisfy the equal interval property.

6.1.4 Ideal ranking
To compare the results obtained by different ranking methods, the ideal ranking method is used. Ideal ranking is based on the concept of average ranking [5]. The only difference between them is, average ranking uses the ranking of the algorithms on a single problem whereas ideal ranking uses the ranking of algorithms for all problems in the database [10]. Table 3 illustrates the computation of ideal ranking for five datasets D_1... D5 and seven algorithms A1... A7.

Table 3. Calculation for ideal ranking

Datasets	**Algorithms**						
	A_1	A_2	A_3	A_4	A_5	A_6	A_7
D_1	2	1	4	3	6	5	7
D_2	1	2	4	3	5	7	6
D_3	3	1	2	4	5	6	7
D_4	2	1	3	4	5	6	7
D_5	1	2	3	4	5	6	7
Mean rank position	1.8	1.4	3.2	3.7	5.7	6	6.8
Final ranking	2	1	3	4	5	6	7

6.1.5 Relative Ranking

Relative ranking is based on the concept of percentiles and refers to the percentage of scores that fall below a given score [28]. A discussion about percentile ranking is found in [28] and the same can be used for algorithm selection. Since we are working with ranks, we know that a low index refers to a higher rank and vice versa. So higher sum of ranking and their corresponding percentile value will refer to lower ranks.

e.g- In table 1, when an algorithm scores 85.71 percentile , 85.71% of scores are above it.

Relative ranking satisfies the i) identity property because each algorithm is uniquely identified by one and only one class. One percentile score represents one rank position and therefore one unique class. In case of a tie, which happens when two algorithms obtain the same percentile values and therefore the same ranking, they belong to the same class. It satisfies the ii) magnitude property, because there exists an ordered relationship between the percentile scores and the ranks obtained by the algorithms. Relative ranking does not satisfy the iii) equal interval property. This can be seen from the example shown in table 1, where we find that distance between the x-th rank position and the y-th rank position is not preserved in the distance between the 'percentile scores' obtained by the algorithms and thus it does not satisfy the iv) ratio property also. A disadvantage associated with relative ranking is that, if the percentile scores are distributed normally, they underestimate large differences in the tails of the distribution and overestimates small differences in the middle of the distribution [28].

6.1.6 Percentage ranking

Percentage ranking is based on the concept of percentage. The percentage value of a score is its value out of hundred. Percentage ranking satisfies the i) identity property because each algorithm is identified by one and only one class and an unique class is identified by an unique percentage score. In case of a tie, which happens when two algorithms obtain the same percentage score and therefore the same ranking, they belong to the same class. It satisfies the ii) magnitude property, because there exists an ordered relationship among the percentage values and the corresponding assigned ranks. Percentage ranking satisfies the iii) equal interval property. This can be seen from the example shown in table 1, where we find that the distance between the x-th rank position and the y-th rank position is preserved in the distance between the 'percentage value of scores' obtained by the algorithms. Here, we can claim this because all values are computed taking hundred as the standard. It also satisfies the iv) ratio property because first it satisfies the ‘equal interval’ property and second since all values are computed out of hundred, a comparison among them is possible.

The use of measurement scale gives an authentic comparison between the results obtained by a system. There are many standard books of statistics, where we can find a lot of details about each individual scale and the statistics those can be computed using them. In this article, our aim is to only discuss their applicability in the context of algorithm ranking. To begin with, let us recall the sample calculation presented in table 2, which shows that all the ranking evaluation methods produced the same ranking, but let us also remember that the calculation is based on simple arithmetic only. In a real life situation, where actually we have to apply the meta-learning technique for algorithm selection, we have to choose one of the most accurate methods. Throughout the literature, we find the use of SRC to compare the recommended ranking with ideal ranking and then choose the better method. A theoretical foundation of the comparison of ranking combination methods is also found in [10]. All these studies encouraged us to look into the ranking combination methods with respect to measurement systems and to consider the case of relative ranking and percentage ranking. Both these cases can be considered for future experiments. Regarding Percentage ranking, we have a point to make here. When we are working with ranking combination methods, the final ranking is built by comparing the mean rank position values of the algorithms on all the datasets and the comparison of means is meaningful only when a measurement system satisfies the ratio property. Out of all the methods, the percentage ranking method only satisfies this property and we expect that the use of the same in experiments may produce more accurate results.

7. Challenges associated with algorithm selection using meta-learning

Prior to Meta-learning, researchers designed a number of new algorithms to overcome the limitations of the known algorithms. There were two reasons for this. The first was to enhance the performance of existing algorithms and the second was the unavailability of suitable algorithms for a new problem. In the second case, the process was considered to violate the principle of machine learning and data mining [12] and also it was extremely time-consuming. In such a situation, meta-learning was used as an alternative. But applying the same principle of the 'No Free-Lunch theorem, we can state that 'No technique is superior to no other technique', when it is considered over the universe of all tasks and this is applicable for meta-learning as well. There are a number of challenges associated with meta-learning when applied for algorithm selection and next we are going to discuss about them.

1. Choice of the base-level algorithms: In the algorithm selection problem model, the algorithm space consists of the base-level algorithms. While designing a meta-learning system, initially one would like to consider those algorithms, which are perceived to be representatives of their respective areas. But, one important consideration is to choose base-learners with complementary expertise areas with an aim to maximize coverage and minimize the size of the algorithm space [11,12]. Maximizing coverage indicates maximizing the application domain of the algorithm which increases the effectiveness of the algorithm [11]. Minimizing the size of the algorithm space comparatively reduces the burden of computing the performances of the algorithms and thus increases the efficiency of the system [11]. These two seem to be two conflicting interests and hence it is challenging to choose the base-level algorithms satisfying these constraints.

2. Choice of the meta-learner: The meta-learner itself is a learning algorithm which selects the most similar problems of a new problem by matching the meta-features of the new problem with the existing meta-features. Thus efficiency of the meta-learner determines the efficiency of the meta-learning system. Most of the literature in meta-learning shows the use of the K-Nearest Neighbor (K-NN) algorithm as the meta-learner due to its simplicity. There has been

very few attempts to use other meta-learners such as MLP, SVM, Naive Bayes etc. through empirical study. So far, no analytical study has been done on this issue.

3. The bias-variance trade-off for the meta-learner: The literature of meta-learning shows the use of supervised machine learning algorithms as meta-learners and every supervised algorithm has a bias and a variance. Bias refers to the set of assumptions made by an algorithm to learn the target function easily. Variance refers to the change in the estimated target output which is supposed to occur with the use of different sets of training data. Ideally, any learning algorithm should have low bias and low variance. But, there exists a reciprocal relationship between them. The K-NN algorithm, which is often used as a meta-learner has a low bias and high variance. The point is whether one uses the K-NN or any other algorithm as a meta-learner, the trick lies in selecting the appropriate parameters of the algorithm.
e.g- selecting the initial value of K (The number of nearest neighbor) .

Selection of the suitable parameters so as to balance the bias and the variance of the meta-learner and achieve good prediction from it is a challenging task.

4. Availability of sufficient data to generate meta-examples: The success of meta-learning depends on the availability of large amount of datasets. The UCI machine learning repository is considered to be a standard for this purpose. However, if we consider a data-intensive task like text-mining, we get only 30 datasets and suppose we consider an application like Game, then we have only 9 datasets at our disposal. In such situations, we have to generate more number of datasets in order to build sufficient number of meta-examples. This still remains as a challenge in the meta-learning community, because only a very few attempts like use of datasetoids and use of the combination of active learning with datasetoids have been made in this direction.

5. Computational cost involved in generating meta-examples: A meta-example is a tuple of the form <f(x), t(x)>[12], where f(x) refers to the feature vector of the problem x and t(x) refers to the desired target output of x which is also a column vector. Generation of one meta-example requires the computation of all meta-features of a problem and also all the algorithms in the algorithm space has to be run over the dataset and their performance has to be stored in the target vector t(x). Both these operations involve some cost and increase in the number of meta-examples increase this cost significantly. Hence, generation of valuable and non-redundant meta-examples is of great importance and particularly challenging for algorithm selection using meta-learning.

6. Cost involved in feature extraction: The cost of extracting features from a new problem must be significantly lower than running the best algorithm on the problem [21]. Otherwise, even if the meta-learning model is very accurate, it will not be appropriate to use it.

7. Choice of problem-characterization method: Although a number of problem characterization methods are available, no study clearly explains which method is suitable for which task. In [12], Carrier explained the advantage of using a model based characterization method. According to him, this is the only method where the dataset is interpreted in the form of a data structure which retains the complexity and performance of the hypothesis function and is not confined to specific examples. In most of the literature, we found the use of a decision tree to induce the model. Investigation of this aspect of meta-learning and attempts to develop more number of suitable model based methods will surely enhance the performance of the meta-learning system.

8. Choice of performance-measures: In most of the data-mining applications, accuracy is used as a performance measure and same in the case of algorithm selection using meta-learning. Few researchers have also considered running time with accuracy. Attempts can be made to consider some of the other qualitative measures such as simplicity, expressiveness, computational complexity (both time and space) and comprehensibility etc.[12].

9. Dealing with the cases of ties: A tie refers to a situation where two algorithms have the same performances on a dataset. Presence of ties worsens the predictive capability of a meta-learning system [21]. But, in a real life situation ties are a part of life and there has to be some mechanism to handle them.

10. Scalability of the meta-learning system: The number of base-label algorithms to build the meta-learning system depends on the availability of computational resources [5]. So scalability of the ML system is a real challenge.

11. Development of automated tools to support algorithm selection: Using meta-learning, a number of automated tools have been developed to support users in algorithm selection and they are Mining Mart, DMA and METALA etc. We have given a short description about DMA in this article. Interested learners may refer to [12] for further study and attempts can be taken to devise more advanced types of tools to support automatic algorithm selection.

8. Conclusion and Future direction

In this work, we have discussed about the state-of-the-art in algorithm selection using meta-learning and have analyzed the aspects of classification and clustering algorithm selections. Having realized the importance of ranking methods in algorithm selection, we have given a detailed study of various ranking combination methods with respect to the measurement system. Also, we have identified ten major challenges in this area, though the list is not restricted to the same. We believe that, the need of the time is to analyze these challenges in greater detail. In literature, we found the attempts to use feature selection techniques for performance improvement. However, we did not find any reference to the use of feature extraction techniques for dimensionality reduction. Further study can be made in that direction.

References

1. Aha, D.W.: Generalizing from case studies: A case study. In Proceedings of the 9th International Conference on Machine Learning. 1-10. (1992)
2. Bhatt, N., Thakkar, A., Ganatra, A.: A survey & current research challenges in meta learning approaches based on dataset characteristics. International Journal of soft computing and Engineering, Vol. 2, No. 10, 234-247. (2012)
3. Brazdil, P. B., Soares, C.: A comparison of ranking methods for classification algorithm selection. In European Conference on Machine Learning. Springer Berlin Heidelberg, 63-75. (2000)
4. Brazdil, P. B., Soares, C.: Zoomed ranking: Selection of classification algorithms based on relevant performance information. In European Conference on Principles of Data Mining and Knowledge Discovery. Springer Berlin Heidelberg, 126-135. (2000)
5. Brazdil, P., Carrier, C. G., Soares, C., Vilalta, R.: Metalearning: Applications to data mining. Springer Science & Business Media. (2008)
6. Chawla, D., Sodhi, N.: Research methodology: concepts and cases. Vikas Publishing House. (2011)
7. De Souto, M. C., Prudencio, R. B., Soares, R. G., De Araujo, D. S., Costa, I. G., Ludermir, T. B., Schliep, A.: Ranking and selecting clustering algorithms using a meta-learning approach. In 2008 IEEE International Joint Conference on Neural Networks. IEEE World Congress on Computational Intelligence, 3729-3735. (2008)
8. Ferrari, D. G., de Castro, L. N.: Performance estimation for clustering algorithms with meta-learning techniques. In 19th Brazilian Conference on Automation. 2380-2386. (2012)
9. Ferrari, D. G., de Castro, L. N.: Clustering algorithm recommendation: a meta-learning approach. In International Conference on Swarm, Evolutionary, and Memetic Computing. Springer Berlin Heidelberg, 143-150. (2012)
10. Ferrari, D. G., De Castro, L. N.: Clustering algorithm selection by meta-learning systems: A new distance-based problem characterization and ranking combination methods. Information Sciences, Vol. 301, 181-194. (2015)
11. Giraud-Carrier, C.: The data mining advisor: meta-learning at the service of practitioners. In Fourth International Conference on Machine Learning and Applications. IEEE. (2005) December).
12. Giraud-Carrier, C: Metalearning-a tutorial. In Tutorial at the 7th international conference on machine learning and applications. San Diego, California, USA. (2008)
13. Handl, J., Knowles, J.: Cluster generators for large high-dimensional data sets with large numbers of clusters. (2005)
14. Kalousis, A., Theoharis, T.: Noemon: Design, implementation and performance results of an intelligent assistant for classifier selection. Intelligent Data Analysis, Vol.3, No.5, 319-337. (1999)
15. Kim, Y., & Hong, C.: A meta-learning approach for combining multiple classifiers. Adv. Sci. Technol. Lett. 29, 50-54. (2013)
16. King, R. D., Feng, C., Sutherland, A.: Statlog: comparison of classification algorithms on large real-world problems. Applied Artificial Intelligence an International Journal, Vol. 9, No. 3, 289-333. (1995)
17. Lee, J. S., Olafsson, S.: A meta-learning approach for determining the number of clusters with consideration of nearest neighbors. Information Sciences, Vol. 232, 208-224. (2013)
18. Neri, F.: Linear Algebra for Computational Sciences and Engineering. (2016)
19. Pang-Ning, T., Steinbach, M., Kumar, V.: Introduction to data mining. Pearson Education India. (2006)
20. Peng, Y., Flach, P. A., Brazdil, P., Soares, C.: Decision tree-based data characterization for meta-learning. (2002)
21. Pfahringer, B., Bensusan, H., Giraud-Carrier, C: Tell me who can learn you and i can tell you who you are: Landmarking various learning algorithms. In Proceedings of the 17th international conference on machine learning. 743-750. (2000)
22. Prudêncio, R. B., Ludermir, T. B.: Selective generation of training examples in active meta-learning. International Journal of Hybrid Intelligent Systems, Vol.5, No. 2, 59-70. (2008)
23. Prudêncio, R. B., Soares, C., Ludermir, T. B.: Combining meta-learning and active selection of datasetoids for algorithm selection. In International Conference on Hybrid Artificial Intelligence Systems. Springer Berlin Heidelberg, 164-171. (2011)
24. Reif, M., Shafait, F., Goldstein, M., Breuel, T., Dengel, A.: Automatic classifier selection for non-experts. Pattern Analysis and Applications, Vol. 17, No. 1, 83-96. (2014)
25. Rice, J. R.: The algorithm selection problem. Advances in computers, Vol.15, 65-118. (1976)
26. Smith-Miles, K. A.: Cross-disciplinary perspectives on meta-learning for algorithm selection. ACM Computing Surveys, Vol. 41, No.1. (2009)
27. Soares, R. G., Ludermir, T. B., De Carvalho, F. A. : An analysis of meta-learning techniques for ranking clustering algorithms applied to artificial data. In International Conference on Artificial Neural Networks. Springer Berlin Heidelberg, 131-140(2009)
28. Stockburger, D. W.: Introductory statistics: concepts, models, and applications. Missouri State University, (2016). [online]. Available: http://www.psychstat.missouristate.edu/IntroBook3/sbk.htm
29. Tal, E.: Measurement in Science. *The Stanford Encyclopedia of Philosophy, (*2015). [online]. Available: http://plato.stanford.edu/archives/sum2015/entries/measurement-science.
30. Vilalta, R., Drissi, Y.: A perspective view and survey of meta-learning. Artificial Intelligence Review, Vol. 18, No. 2, 77-95. (2002)
31. Vilalta, R., Giraud-Carrier, C. G., Brazdil, P., Soares, C.: Using Meta-Learning to Support Data Mining. International Journal on Computer Science and Applications, Vol. 1, No. 1, 31-45. (2004)
32. Wolpert, D. H., Macready, W. G., No free lunch theorems for optimization. IEEE transactions on evolutionary computation, Vol. 1, No. 1, 67-82. (1997)