# Automated Data Cleansing through Meta-learning

**Ian Gemp**[*]
College of Information and Computer Sciences
University of Massachusetts Amherst
Amherst, MA 01003
imgemp@cs.umass.edu

**Georgios Theocharous** and **Mohammad Ghavamzadeh**
Adobe Research
San Jose, CA 95113
{theochar,ghavamza}@adobe.com

## Abstract

Data preprocessing or cleansing is one of the biggest hurdles in industry for developing successful machine learning applications. The process of data cleansing includes data imputation, feature normalization & selection, dimensionality reduction, and data balancing applications. Currently such preprocessing is manual. One approach for automating this process is *meta*-learning. In this paper we experiment with state of the art meta-learning methodologies and identify the inadequacies and research challenges for solving such a problem.

## Introduction

*Data cleansing* which includes data imputation (filling in missing values), feature normalization & selection, dimensionality reduction, and data balancing can be a painfully tedious, yet crucial process. Unfortunately, "dirty" data is generally the rule in real-world industrial settings and so automated cleansing is highly valued. Meta-learning presents a viable solution to this problem.

Introductory machine learning courses teach that models can be applied to data intelligently by understanding their behavior in relation to data (e.g., # of features or samples). Meta-learning extends this paradigm by encoding datasets with a vector representation (metafeatures) containing informative statistics of the dataset. Assuming datasets lie on a manifold, datasets nearby in metafeature space might behave similarly when consumed by similar models; in other words, the best cleansing approach for dataset A might work well for nearby dataset B (see Figure 1). Current meta-learning techniques generally employ the same metafeatures and distance metric ($L_1$), however, it is not clear if these ad hoc choices describe an accurate metafeature manifold.

Discovering a more accurate representation could help close a large performance gap that we illuminate in our experiments. We also test several metric learning methods, however, none is able to shrink this gap. We conclude that the metafeatures circulated by the community are inadequate for this task and learning a better metafeature representation would benefit automated data cleansing (as well as AutoML) and our understanding of models' behaviors on various classes of datasets.

[*]Work done during an internship at Adobe Research.



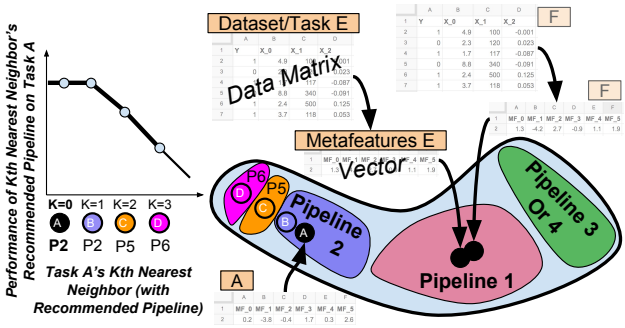


Figure 1: *Metafeature manifold*—Datasets (more generally tasks) whose embeddings are nearby in *metafeature space* (light blue) should share similar preprocessing pipelines. We challenge the community to learn such an embedding.

## Meta-learning in Practice

Currently, metafeatures are chosen manually and a set of standard features are used in the meta-learning literature [1, 2]. These include simple counts such as numbers of samples & features and distributional statistics such as mean feature skew. When meta-learning components are employed in automated systems, it is assumed that these features retain a sufficient amount of information for distinguishing datasets in terms of which models are best suited for them. Our results suggest otherwise.

Sequential model based optimization (SMBO) iteratively updates an initial model seed (e.g., a data cleansing pipeline) to improve downstream performance (e.g., classification error). Using meta-learning to provide a model seed has been shown to significantly improve performance over randomly selected model seeds [1], however, we demonstrate this is likely an artifact of the meta-learning process and not a result of what meta-learning aims to promise.

## Experimental Setup

As discussed in the introduction, the choice of distance metric is typically ignored. By exploring a number of distance metrics, we expect we might discover one that better represents the metafeature manifold than $L_1$ distance. We evaluated $M$ data cleansing pipelines on $N$ datasets and com-

puted a metafeature vector for each of the $N$ datasets. We used 22 standard metafeatures from the literature. We repeated this experiment for three different data sources (available on author's website): one constructed by artificially creating 432 binary classification datasets using Scikit-Learn's `make_classification` method [4], another using 816 binary classification tasks created from web activity of various companies which use Adobe's digital marketing solutions, and a third consisting of 397 binary classification tasks downloaded from OpenML.org [5]. We considered 192 different possible data cleansing pipelines each constituting a sequence of preprocessing components with hyperparameters (mean/mode imputation, feature selection by percentile, $L_1/L_2$ feature normalization, over/under sampling, PCA) in conjunction with $L_2$-regularized logistic regression. We were then able to rank the $M$ data cleansing pipelines by classification accuracy on each of the $N$ datasets. To be able to compare pipeline performance across datasets, we scaled pipeline scores for each dataset to [0,1] (0 being the score for the worst pipeline).

Using an appropriate distance metric, we expect that two *nearby* datasets as measured by the metric, would rank the data cleansing pipelines similarly. More importantly, they would most likely agree on the top performing pipeline so that similar datasets should be cleaned similarly. For instance, an optimal metric would ensure the performance of neighbor's recommended pipelines decayed monotonically with increasing dataset distance. In order to discover this metric, we looked at five static metrics ($L_1, L_2, L_\infty$, cos, Canberra), two learned $L_1$ variants we developed, four Mahalanobis metric learning variants (e.g., LEGO [3]), and three deep, Mahalanobis *pseduo*-metrics inspired by [6]. To train these metrics, we designed two different sets of labels from the pipeline performances (see previous paragraph). Specifically, we trained our learned metrics using either a ranking distance between dataset's rankings of full pipelines or $d_{swap}$. Let $D_i$ denote dataset $i$, $P^*_{D_i}$ denote the pipeline with the highest score on $D_i$, and $Err(D_i, P_j)$ denote the normalized error obtained when cleaning $D_i$ with $P_j$. Then $d_{swap} = \frac{1}{2}(Err(D_i, P^*_{D_j}) + Err(D_j, P^*_{D_i}))$.

## Results

Figure 2 compares various metrics to the optimal metric. We omit other learned metrics due to space limits. Metric performance is most important for the first few nearest neighbors because only their recommended pipelines are accepted in meta-learning approaches. Meta-learning with a random metric (neighbors' distances are generated randomly) outperforms randomly selecting a single pipeline from the 192 possible pipelines (lower bound). This is because neighbors only recommend their top 1 (or $p$) pipeline(s) and so generally low performing pipelines are pruned from the recommendation pool. Using a metric improves over the random metric, however, the learned metric does not provide any significant improvement. This was true across all the metric learning algorithms. These results suggest the chosen metafeatures are inadequate for describing a smooth manifold, and that work on learning a more representative, yet explainable set of metafeatures is necessary to advance meta-learning.

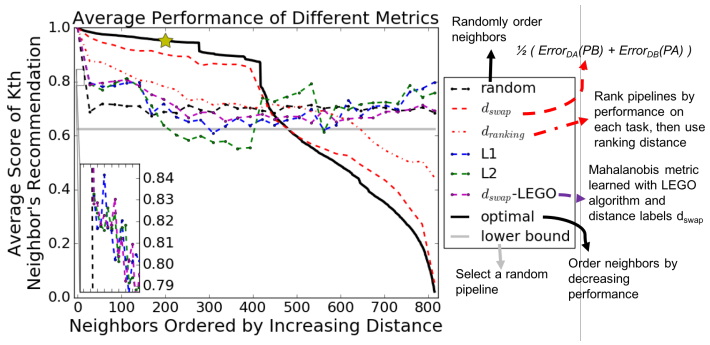


Figure 2: Mean performance of $k$th nearest neighbor's top recommended pipeline for Adobe experiment. For example, on average, using an optimal metric, a dataset's 200th nearest neighbor recommends a pipeline that would perform only 5% worse than the best pipeline (see yellow star). In contrast, $L_1$ distance performs nearly 30% worse.

## Challenge

Our challenge to the community is to learn a metafeature representation (with accompanying evaluation test) that is well suited for meta-learning in the data cleansing task. Accomplishing this task will have impact beyond just automated data cleansing and the related goal of automated machine learning. An explainable metafeature representation can be used to enhance our understanding of models tendencies in the face of varying data as well as provide an important educational tool.

## References


[1] M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter. Efficient and robust automated machine learning. In *Advances in Neural Information Processing Systems*, pages 2962–2970, 2015.

[2] T. Gomes, R. Prudêncio, C. Soares, A. Rossi, and A. Carvalho. Combining meta-learning and search techniques to select parameters for support vector machines. *Neurocomputing*, 75(1):3–13, 2012.

[3] P. Jain, B. Kulis, I. Dhillon, and K. Grauman. Online metric learning and fast similarity search. In *Advances in neural information processing systems*, pages 761–768, 2009.

[4] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[5] J. Vanschoren, J. van Rijn, B. Bischl, and L. Torgo. Openml: Networked science in machine learning. *SIGKDD Explorations*, 15(2):49–60, 2013.

[6] J. Wang, A. Kalousis, and A. Woznica. Parametric local metric learning for nearest neighbor classification. In *Advances in Neural Information Processing Systems*, pages 1601–1609, 2012.