

Data Aggregation for Reducing Training Data in Symbolic Regression

Lukas Kammerer^[0000–0001–8236–4294], Gabriel Kronberger^[0000–0002–3012–3189],
and Michael Kommenda

Josef Ressel Center for Symbolic Regression
Heuristic and Evolutionary Algorithms Laboratory
University of Applied Sciences Upper Austria, Hagenberg, Austria
`lukas.kammerer@fh-hagenberg.at`

Abstract. The growing volume of data makes the use of computationally intense machine learning techniques such as symbolic regression with genetic programming more and more impractical. This work discusses methods to reduce the training data and thereby also the runtime of genetic programming. The data is aggregated in a preprocessing step before running the actual machine learning algorithm. K-means clustering and data binning is used for data aggregation and compared with random sampling as the simplest data reduction method. We analyze the achieved speed-up in training and the effects on the trained models’ test accuracy for every method on four real-world data sets. The performance of genetic programming is compared with random forests and linear regression. It is shown, that k-means and random sampling lead to very small loss in test accuracy when the data is reduced down to only 30% of the original data, while the speed-up is proportional to the size of the data set. Binning on the contrary, leads to models with very high test error.

Keywords: Symbolic Regression · Machine Learning · Sampling

1 Introduction

One of the first tasks in data-based modeling of systems is collection and selection of data with which a meaningful model can be learned. One challenge is to provide the right amount of data – w.r.t. both instances and features. There should be enough data to compensate noise and train a sufficiently complex model, but not too much to unnecessarily slow down the training. With the growing volume of data, especially the latter becomes more and more an issue when working with computationally intensive algorithms like genetic programming (GP).

An intuitive idea to keep the training data small is outlined in Figure 1. Instead of using all training data, only a few representative instances are extracted first and then used for the training. Ideally, these few instances retain all

¹ The final publication is available at https://link.springer.com/chapter/10.1007%2F978-3-030-45093-9_46

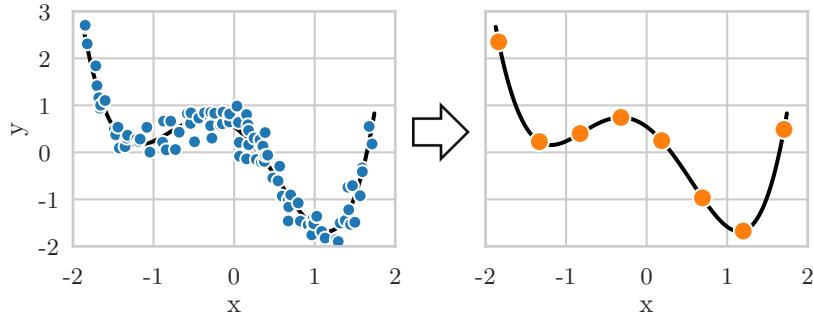


Fig. 1. Schematic outline of extracting a data set of representative instances (right) out of original noisy data (left).

information that are necessary to train a well-generalizing model. This idea has been already applied for support vector machines for classification [9] and regression tasks [3]. In this previous work, the authors heuristically selected those instances, which are likely to determine the support vectors and therefore the SVM model. Another proposed approach for speeding up GP’s evaluation is to use only a small random sample in every evaluation [5]. Kugler et al. [6] suggested to aggregate similar instances together before training a neural network. Grouping together instances should cancel out noise and shrink the data set, which is similar to the initially mentioned idea.

This paper builds on the idea of Kugler et al. [6] and uses clustering algorithms for aggregating and reducing training instances. The applied methods are random sampling, data binning and k-means clustering. The new, aggregated data are then used for training with different machine learning algorithms, with a focus on symbolic regression with GP. The trade-off between speed-up and loss in prediction accuracy due to potential removal of information is analyzed. We test how much we can reduce data so that we still can train accurate and complex models.

2 Aggregation Methods

All three aggregation methods require a predefined number of instances, which should be generated out of the original data. The first one, random sampling without replacement, serves as lower baseline. It will show, how much data can be removed without losing relevant information for modeling.

In data binning, the training data are aggregated based on the target variable. Similar to a histogram, instances with target values in fixed ranges are grouped together into bins. The range is determined by the minimum/maximum of the original target values and the predefined number of bins. To reduce the instances in a bin to one single instance, we use the median for all of the features. Binning should provide an equal target value distribution and reduce noise and variance

[6]. However, binning also implies a "many-to-one mapping" between features and target variable, which incurs a loss of information in cases where interactions of features are relevant.

K-means clustering searches k cluster centroids that minimize the sum of the Euclidean distance between each point and its closest centroid. In this work, the calculated centroids are used as new training data, while in usual applications, k-means is used as unsupervised learning algorithm to separate the data into k groups. We deliberately consider both features and target values when aggregating the data because it is assumed that information about variable interactions is retained in contrast to data binning. The most common algorithm for determining the centroids is the heuristic Lloyd's algorithm [7], is infeasible for larger data sets with high k , since its runtime complexity is linear to the number of instances and k – especially when the goal is actually to speed up the overall modeling process. For our experiments, we use the mini-batch k-means algorithm [10], which reduces the runtime of the preprocessing from days to a few minutes with comparable results.

3 Experimental Setup

This work focuses on the effect of the described aggregation methods on symbolic regression with GP. We use the algorithm framework described by Winkler [13], in which the prediction error of mathematical formulas in syntax tree representation as individuals is minimized. The algorithm implementation applies strict offspring selection with gender-specific selection [1], a separate numerical optimization of constants in the formula [5] and explicit linear scaling [4]. Since we focus on the effect of preprocessing on symbolic regression, we use a standard parameter setting which has shown in our experience to provide good results. GP's maximum selection pressure was set to 100, the mutation rate to 20% and the population size to 300. The trees are build with a grammar of arithmetic and trigonometric symbols, as well as exponential and logarithm functions. The tree size is limited at most 50 symbols and a maximum depth of 30. The crossover operator is subtree swapping and mutation operators are point mutation, tree shaking, changing single symbols and replacing/removing branches [1]. We use for all experiments the *HeuristicLab* framework¹ [11].

Random forests (RF) [2] and linear regression (LR) are run on the same data sets in order to provide comparability of the achieved results. Linear regression serves as a lower bound, as the resulting linear models have low complexity and all relations in the data sets are nonlinear. Random forests are used as a rough indicator, which accuracy values are achievable on different data sets, as this algorithm has shown to be both fast and often very accurate. The settings for RF were set train 50 trees and sample from 30% of instances and 50% of features for each tree.

This work uses four real-world data sets: The *Chemical-I* data set (711 training instances, 57 features) [12], the *Tower* data set [12] (3136 training instances,

¹ <https://dev.heuristiclab.com>

25 features), the *SARCOS* data set [8] (44 500 training instances, 21 features) and the *puma8NH* data set (6144 training instances, 8 features) from the Delve repository². The data sets are split into training and test data set according to the cited papers. The data sets are normalized to a mean of zero and a standard deviation of 1 in the training data, which is especially important for the k-means algorithm, which uses the Euclidean distance.

To analyze the tradeoff between accuracy and speed-up, the training data are reduced stepwise, starting from a reduction to one percent up to 50 percent of the original training data size. For each of these steps, ten reduced training data sets are generated. The three machine learning algorithms are then run on each of these reduced data sets. Figure 2 outline the estimation of the generalization error, where we evaluate the trained models on the not-aggregated test data. The generalization error is measured with Pearson’s R^2 coefficient, which describes the correlation between actual target values predictions in the interval $[0, 1]$.

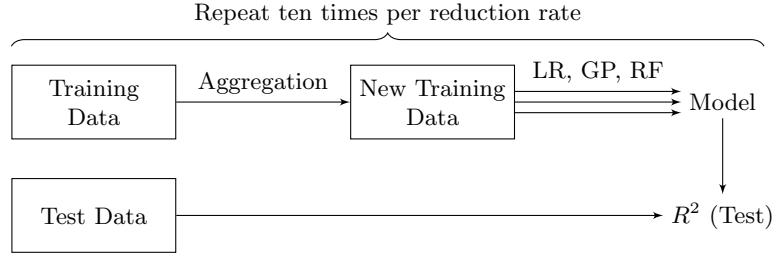


Fig. 2. Experiment workflow of reducing data for each data set and reduction rate.

4 Results

Figure 3 shows, that the speed-up of k-means clustering and sampling is proportional to the reduction rate – how much the data has been reduced relative to the original size. This is expected, as GP spends most of its time evaluating individuals. The computational effort of evaluations increases proportionally with larger numbers of training instances, because every instance’s prediction has to be calculated in every evaluation. However, the proportional speed-up also indicates, that algorithm dynamics such as the earlier convergence due to preprocessing can be ruled out. The runtime for the preprocessing step itself is neglected, because it made up at most three minutes (for the large Sarcos data set) per run, which is only a very small fraction of the runtime of GP. Figure 3 also excludes the runtime for RF and LR because both methods took only seconds to finish in all experiments.

While data binning and k-means led to similar execution times, GP runs with preceding data binning were slightly faster. This is most likely due to a loss

² <https://www.dcc.fc.up.pt/~ltorgo/Regression/puma.html>

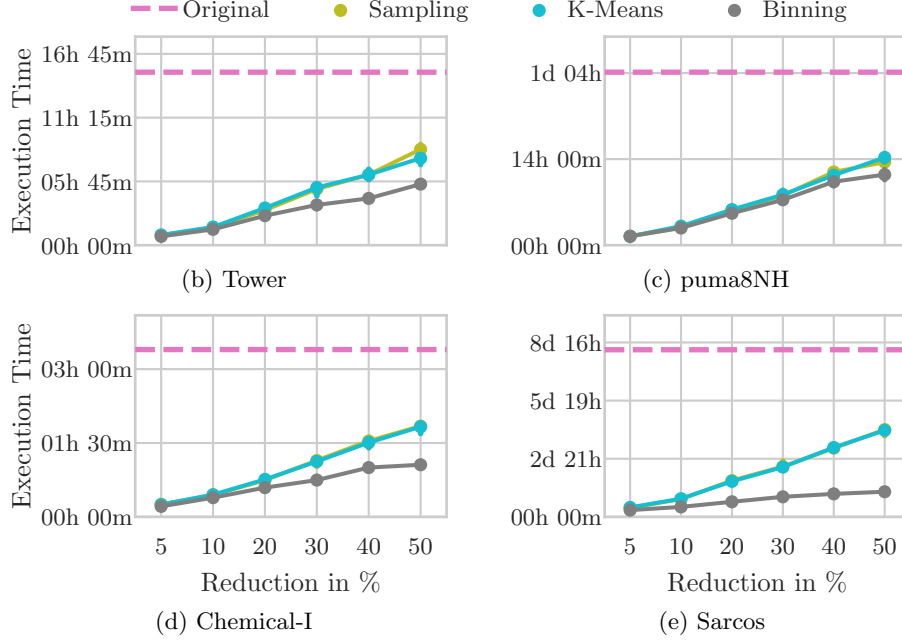


Fig. 3. Median execution of GP for different rates of reduction of the original data set.

of information about the relations in the data. Fewer relations in the training data make the search for a model, which fits the training data well, easier and therefore faster although it degrades its test errors. Table 1 shows that data binning yielded throughout worse models regarding test accuracy.

K-means clustering and random sampling produced very similar results, as listed in Table 1. In both cases, the loss in prediction accuracy is small when the data is reduced to 30%, 40% or 50% of the original data in comparison to runs with the original data. The only difference is the more stable behavior of k-means preprocessing when the training data is reduced to a size of 20% or less of the original data set size. The small difference between preprocessing with k-means and data binning can be explained with the small number of instances, with which each centroid is computed – e.g. if the training data is reduced to 50%, each centroid is the center of only two similar instances on average. This leaves little space for improvements over random sampling. However, all methods failed to yield meaningful models when the training data are reduced to only 1% of their original size.

When compared to RF and LR, GP achieves in most experiments similar accuracy as random forests. However, when the data is strongly reduced to only 30% or less of the original data, GP tends to have less loss in test accuracy than RF. Figure 4 describes the median R^2 results of all three algorithms with differently reduced data for each data set. Modeling results on the original data

Table 1. Test R^2 median and interquartile range for symbolic regression with GP.

		Chemical-I	Sarcos	Tower	puma8NH
Original		.863 (.02)	.949 (.00)	.937 (.01)	.684 (.00)
Sampling	1	.002 (.00)	.928 (.01)	.390 (.31)	.369 (.06)
	5	.431 (.36)	.946 (.00)	.907 (.02)	.572 (.04)
	10	.396 (.36)	.947 (.00)	.919 (.02)	.632 (.02)
	20	.728 (.21)	.948 (.00)	.925 (.01)	.657 (.00)
	30	.787 (.08)	.948 (.00)	.933 (.00)	.668 (.01)
	40	.836 (.01)	.947 (.00)	.934 (.01)	.675 (.01)
K-Means	50	.834 (.02)	.949 (.00)	.936 (.00)	.677 (.01)
	1	.018 (.04)	.943 (.01)	.729 (.14)	.480 (.11)
	5	.523 (.15)	.948 (.00)	.918 (.01)	.596 (.07)
	10	.704 (.13)	.948 (.00)	.926 (.00)	.596 (.09)
	20	.806 (.06)	.948 (.00)	.933 (.01)	.656 (.05)
	30	.828 (.03)	.948 (.00)	.932 (.00)	.665 (.01)
Binning	40	.853 (.04)	.948 (.00)	.932 (.00)	.672 (.01)
	50	.848 (.02)	.947 (.00)	.931 (.00)	.680 (.00)
	1	.009 (.01)	.504 (.45)	.378 (.37)	.021 (.03)
	5	.253 (.30)	.866 (.01)	.313 (.35)	.189 (.31)
	10	.345 (.21)	.893 (.01)	.700 (.27)	.532 (.05)
	20	.578 (.13)	.892 (.01)	.806 (.05)	.530 (.07)
	30	.673 (.05)	.899 (.01)	.853 (.03)	.576 (.05)
	40	.699 (.05)	.898 (.01)	.872 (.05)	.630 (.03)
	50	.718 (.08)	.897 (.01)	.883 (.02)	.648 (.01)

set are shown on the left. The results for data, that was reduced to 1%, as well as the binning results are not shown in Figure 4 since they would degrade the axis scale.

5 Conclusion

The experimental results show, that the original training data can be reduced to only 30% of the original size while still achieving only slightly worse test accuracy. While data binning led to unusable models regarding test accuracy, there is no noticeable difference in the resulting modeling accuracy between k-means and random sampling. Depending on the actual application, whether a higher loss in accuracy is acceptable, k-means might only be useful if the data is reduced to 5-20%, as less variance among test errors compared to random sampling was observed in this range. Otherwise, it is more convenient to use random sampling instead of the additional effort of implementing k-means clustering in the modeling process. The impact of data reduction was smaller for GP than for RF, which underlines the stability and the generalization capabilities of symbolic regression.

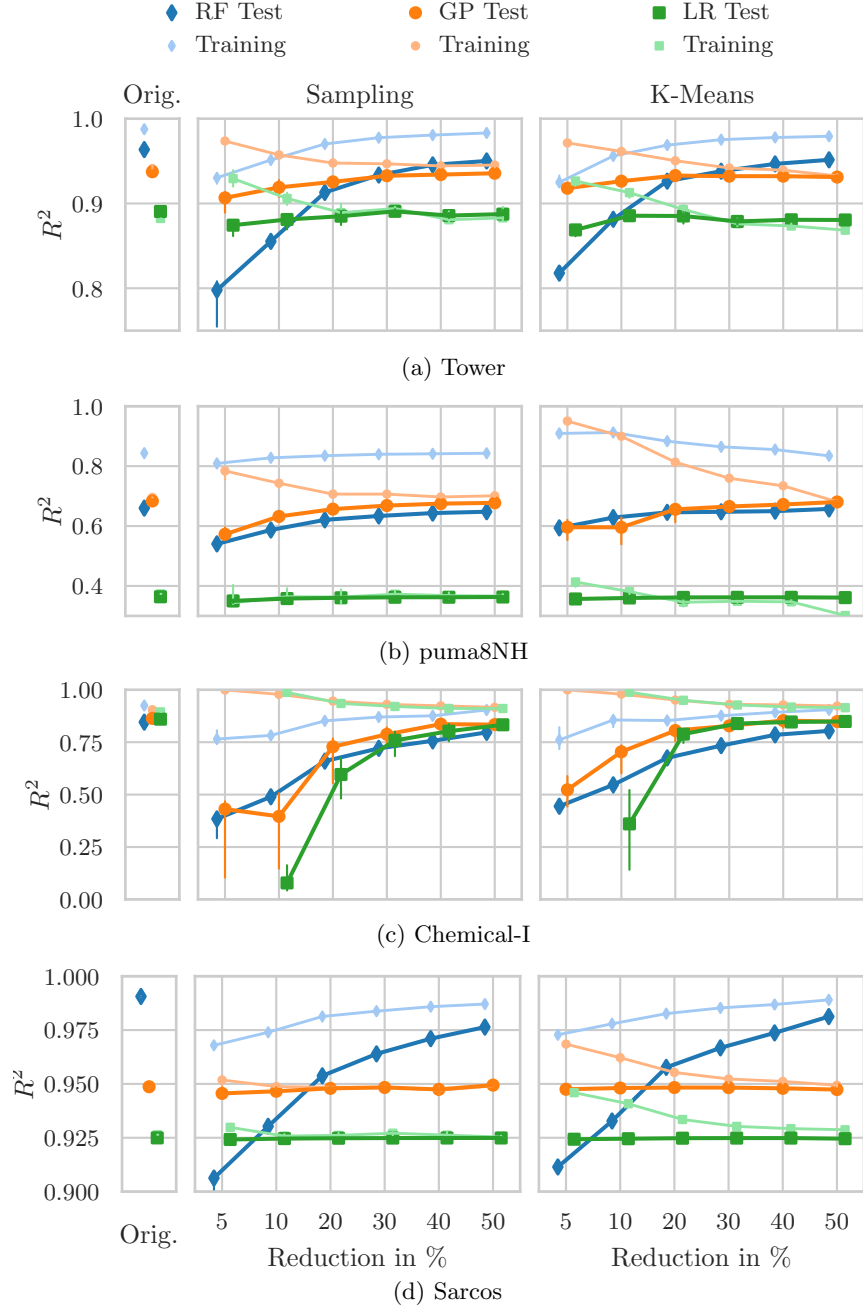


Fig. 4. Median R^2 values of the generated models.

The speed-up for GP is proportional to the data reduction ratio, how much the original data was reduced to. While the proposed reduction methods might not be suitable for a final model in most applications due to the (even slight) loss in accuracy, random sampling as preprocessing step might be a suitable tool for speeding up early experimental phases and meta parameter tuning.

Acknowledgements. The authors gratefully acknowledge support by the Austrian Research Promotion Agency (FFG) within project #867202, as well as the Christian Doppler Research Association and the Federal Ministry of Digital and Economic Affairs within the *Josef Ressel Centre for Symbolic Regression*

References

1. Affenzeller, M., Winkler, S., Wagner, S., Beham, A.: Genetic Algorithms and Genetic Programming - Modern Concepts and Practical Applications, Numerical Insights, vol. 6. CRC Press, Chapman & Hall (2009)
2. Breiman, L.: Random forests. *Machine learning* **45**(1), 5–32 (2001)
3. Guo, G., Zhang, J.S.: Reducing examples to accelerate support vector regression. *Pattern Recognition Letters* **28**(16), 2173–2183 (2007)
4. Keijzer, M.: Scaled symbolic regression. *Genetic Programming and Evolvable Machines* **5**(3), 259–269 (2004)
5. Kommenda, M., Kronberger, G., Affenzeller, M., Winkler, S., Feilmayr, C., Wagner, S.: Symbolic regression with sampling. In: 22nd European Modeling and Simulation Symposium EMSS. pp. 13–18 (2010)
6. Kugler, C., Hochrein, T., Dietl, K., Heidemeyer, P., Bastian, M.: Softsensoren in der Kunststoffverarbeitung: Qualitätssicherung für die Compoundierung und Extrusion. SKZ - Forschung und Entwicklung, Shaker Verlag GmbH (2015)
7. Lloyd, S.: Least squares quantization in pcm. *IEEE transactions on Information Theory* **28**(2), 129–137 (1982)
8. Pagie, L., Hogeweg, P.: Evolutionary consequences of coevolving targets. *Evolutionary computation* **5**(4), 401–418 (1997)
9. Rychetsky, M., Ortmann, S., Ullmann, M., Glesner, M.: Accelerated training of support vector machines. In: IJCNN’99. International Joint Conference on Neural Networks. Proceedings (Cat. No. 99CH36339). vol. 2, pp. 998–1003. IEEE (1999)
10. Sculley, D.: Web-scale k-means clustering. In: Proceedings of the 19th international conference on World wide web. pp. 1177–1178. ACM (2010)
11. Wagner, S., Affenzeller, M.: Heuristiclab: A generic and extensible optimization environment. In: Adaptive and Natural Computing Algorithms, pp. 538–541. Springer (2005)
12. White, D.R., McDermott, J., Castelli, M., Manzoni, L., Goldman, B.W., Kronberger, G., Jaśkowski, W., O’Reilly, U.M., Luke, S.: Better gp benchmarks: community survey results and proposals. *Genetic Programming and Evolvable Machines* **14**(1), 3–29 (2013)
13. Winkler, S.: Evolutionary System Identification: Modern Concepts and Practical Applications. Schriften der Johannes Kepler Universität Linz, Universitätsverlag Rudolf Trauner (2009)