Modeling Generalization in Machine Learning: A Methodological and Computational Study

Pietro Barbiero
Cambridge University
United Kingdom
barbiero@tutanota.com

Giovanni Squillero
Politecnico di Torino
Italy
squillero@polito.it

Alberto Tonda
Université Paris-Saclay, INRAE
France
alberto.tonda@inrae.fr

July 22, 2024

Abstract

As machine learning becomes more and more available to the general public, theoretical questions are turning into pressing practical issues. Possibly, one of the most relevant concerns is the assessment of our confidence in trusting machine learning predictions. In many real-world cases, it is of utmost importance to estimate the capabilities of a machine learning algorithm to generalize, i.e., to provide accurate predictions on unseen data, depending on the characteristics of the target problem. In this work we perform a meta-analysis of 109 publicly-available classification data sets, modeling machine learning generalization as a function of a variety of data set characteristics, ranging from number of samples to intrinsic dimensionality, from class-wise feature skewness to $F1$ evaluated on test samples falling outside the convex hull of the training set. Experimental results demonstrate the relevance of using the concept of the convex hull of the training data in assessing machine learning generalization, by emphasizing the difference between interpolated and extrapolated predictions. Besides several predictable correlations, we observe unexpectedly weak associations between the generalization ability of machine learning models and all metrics related to dimensionality, thus challenging the common assumption that the *curse of dimensionality* might impair generalization in machine learning.

***K*eywords** Convex hull · Curse of dimensionality · Data set characteristics · Extrapolation · Generalization · Interpolation · Machine Learning · Symbolic regression

1 Introduction

The term *machine learning* (ML) traditionally includes algorithms that are able to improve their performance on a specific task over time, given an increasing amount of relevant data [1]. In recent years, this field of research is enjoying a growing popularity, driven by the breakthrough of Deep Learning [2] and an impressive track record of success stories in different fields, ranging from natural language processing [3] to autonomous vehicles [4], image classification [5] human-competitive performance in boardgames [6]. An interesting *online collection* about various uses of ML has been compiled by the journal Nature is late 2018[1], although the fast pace the field is progressing made it to appear outdated after few quarters.

As out-of-the-box ML solutions are becoming increasingly available to both researchers and the general public [7, 8, 9, 10], theoretical questions are suddenly turning into practical issues. Among all common inquiries, perhaps the most basic is: can ML work on a specific problem? Or, in other words: given the characteristics of a target data set, can the effectiveness of a ML approach be predicted? Interestingly, this latter question can be further rephrased as: what are the characteristics of a data set that are well correlated with the possibility, or the impossibility, of obtaining ML models able to effectively extrapolate to unknown instances of the problem? It is well known that ML algorithms are

Authors are listed in alphabetical order

[1] https://www.nature.com/collections/csgqqsrfxh/content/machine-learning (*The multidisciplinary nature of machine intelligence*, 26 September 2018)

affected by the *curse of dimensionality* [11], but ML practitioners also know that it could be possible to obtain reliable models even for high-dimensional data sets, and with a relatively small number of samples [12]. The common approach among practitioners in the field, when dealing with a new data set, seems to be: try as many different ML algorithms as possible in a cross-validation, and evaluate the outcomes; then focus on the techniques that provided the best results, possibly applying them in an *ensemble* [13].

Taking inspiration from [14], where the authors find links between data set characteristics and efficiency of feature selection techniques, we propose to empirically explore the relation between data-set characteristics and effectiveness of standard ML models, in order to obtain a general meta-model able to extrapolate. In order to answer the question, we analyzed 109 publicly available classification data sets from open-access, curated sources. We decided to focus on classification, as supervised ML represents a quite significant portion of real-world problems; and, differently from regression, several sophisticated quality metrics have already been developed for this task [15].

During the analysis, we take into account characteristics such as number of features, number of classes, number of samples, and we look for correlations with quality metrics, such as accuracy of a ML model on training and test points. Extrapolation is assessed not just by alternatively dividing the data into training and test sets, but by analyzing whether data points fall inside or outside of the convex hull of the training data. After collecting the meta-data on the performance of a state-of-the-art classification algorithm on the data sets, the statistical analysis presents both predictable and surprising results, hinting at the fact that dimensionality might not be so cursed after all.

Main contributions

1. We ran a quantitative evaluations of ML models over 109 publicly available data sets.
2. In Section 2 and Section 3 we provide a general overview on generalization in ML, and of the so-called *curse of dimensionality*.
3. In Section 4 we propose to take into consideration the convex hull of the training set to better understand generalization in ML, as it makes it possible to separate cases when an algorithm interpolates or extrapolates on test points.
4. In Section 5 we empirically model associations between the ability to generalize and the data set characteristics, challenging the relationship between generalization ability and the dimensionality.

2 The curses of dimensionality

The *curse of dimensionality* denotes a variety of different phenomena that impair data analysis if a large number of variables need to be considered at the same time [16, 17]. While in most cases it has no closed form nor a unique solution, it has distinctive deleterious effects. Problems like *data sparsity*, *collinearity*, and *overfitting*, seem to confirm the platitude that in ML dimensionality is cursed [11].

As an example, let consider a collection of n data points generated by sampling two random variables X_1 and X_2 originated from two standard normal distributions ($\mu = 0, \sigma = 1.0$). The amount of samples falling within the interval $x_i \pm \sigma$ with $x_i = 1.5$ is around the 30% both for X_1 and X_2, individually (grey histograms in Figure 1). However, when the joint distribution of X_1 and X_2 is considered, the amount of samples falling within the joint interval $(x_1 \pm \sigma, x_2 \pm \sigma)$ with $x_{1,2} = 1.5$ drops. The worst-case scenario arises when the two random variables are uncorrelated (Figure 1, left). As the number of random variables (p) increases, the fraction of points within the p-dimensional interval of radius σ decreases rapidly: $\sim 5\%$ for $p = 2$, $\sim 1\%$ for $p = 3$, and $\sim 0\%$ for $p = 4$ [11]. In practice, the sparser the samples the harder will be collecting data that are representative of the population. The best-case scenario occurs when the random variables are perfectly correlated (Figure 1, right). In this case, the decrease is still significant, but much slower (from 30% to 20% rather than 5% for $p = 2$).

In most real-world scenarios, a considerable number of variables are correlated. As supervised ML algorithms are usually devised to select and use variables strongly correlated with the target variable, the data-sparsity problem may be usually mitigated by considering together highly correlated sets of variables. However, exploiting correlated variables may also have a catastrophic impact when variables are used for prediction: as there could be more than one subset of variables yielding approximately the same result, considering them together can make to make impossible to understand the individual impact of each variable. For example, suppose that two variables X_1 and X_2 can be used to predict a target variable Y by means of a predictive model f:

$$Y = f(X_1, X_2) \tag{1}$$

Now suppose that there exists another variable X_3 that can be expressed as a function of both X_1 and X_2, e.g., $X_3 = X_1 + 2X_2$. Such a system of equations can perfectly be solved by using an arbitrary pair of variables $\{(X_1, X_2), (X_1, X_3), (X_2, X_3)\}$, as they are all perfectly correlated. Even though the predictive problem appears to be solved, the causative source of variability of Y is now uncertain, and the relative importance of the variables cannot be estimated from data. The situation gets critical when the number of variables exceeds the number of samples ($p > n$): at least one of the variables can always be expressed as a linear combination of the others, thus yielding multiple perfect correlations [18].

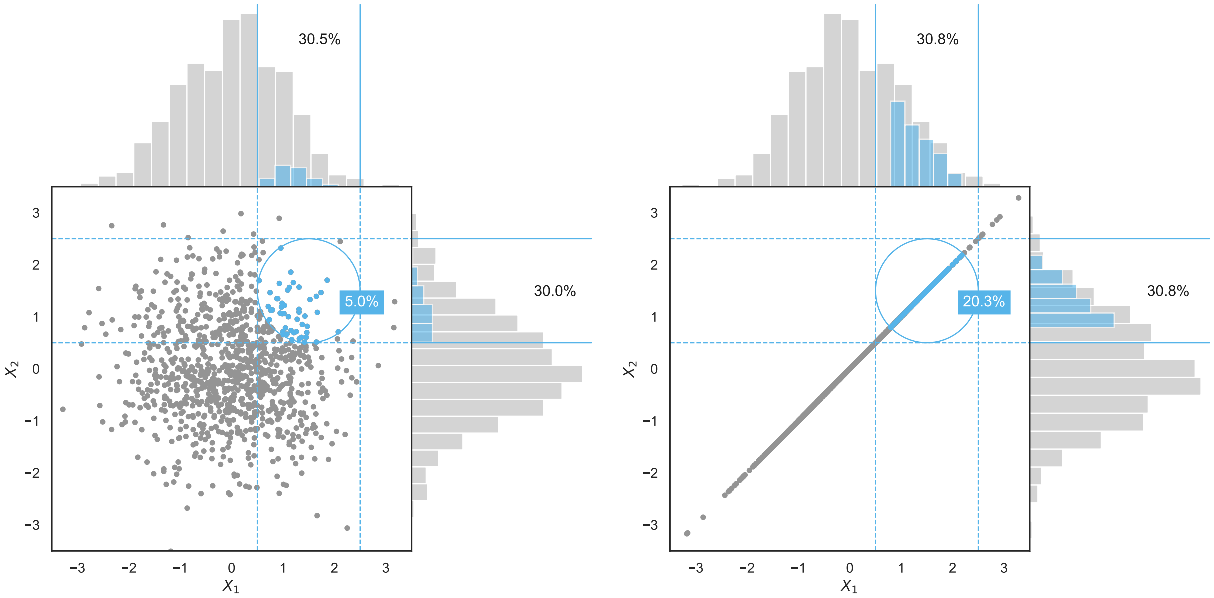


Figure 1: Visual representation of the data sparsity phenomenon in case of correlated random variables (left) and collinear random variables (right).

In case of multicollinearity, one of the possible solutions exploited by supervised ML algorithms (e.g., logistic regression [19]) is to associate a weight to each variable, corresponding to its relevance. Instead of discarding variables which may be the true causative source of variability, these approaches make it possible to take into account all the observed variables at the same time, by increasing the number of model's parameters. Such increase in model complexity may in turn be a possible cause of overfitting, another phenomenon related to the curse of dimensionality. In fact, the increased flexibility makes the model not only able to fit the underlying relationship between variables, but also the random idiosyncrasies of the observations [20].

3 Extrapolation and interpolation in machine learning

The objective of supervised ML can be roughly summarized as automatically obtaining a *predictive model* for a specific phenomenon starting from a set of known cases. Usually such known cases consist in different instances of measurements, or *samples*, each one specifying the values of all different variables, or *features* of the problem. One of these features is the *target variable*, that is, what should be predicted by the final model. ML algorithms try to find a mathematical relationship between the target feature and the others in the set of known data, or *training set* of data. Eventually, such relation can be encoded with a linear model [21], more complex structures [2], or even ensembles of simpler models [13, 22]. Indeed, it is quite important to verify to what extent the model obtained is able to *generalize*, that is, to provide meaningful prediction for new instances of the problem. A model that is only able to deal with the data it was trained on is said to be *overfitted* [20], and it is generally considered useless regardless its performance on the train set.

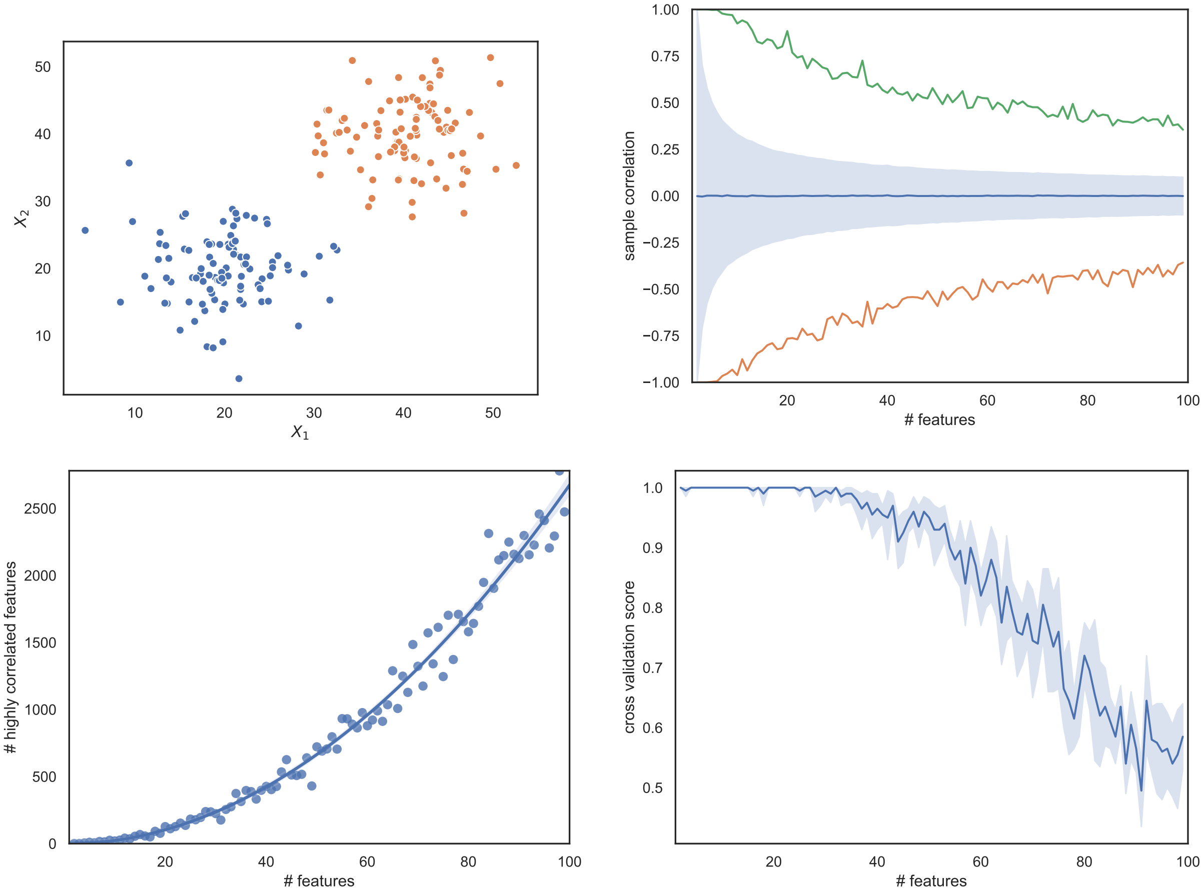


Figure 2: Two-class samples drawn from two correlated Gaussian random variables. Notably the classification problem is trivial as the two classes are linearly separable (top-left). By increasing the number of correlated random variables the highest value (in green), the lowest value (in orange), and the variance of samples' correlation shrink towards zero (top-right). The number of highly correlated variables ($\rho > 0.8$) increases polynomially with the overall number of features (bottom-left). On the other hand, the generalization accuracy of logistic regression in predicting class labels decreases, down to the point of providing almost random estimates (bottom-right).

In ML the term *capacity* may describe the ability of a model to represent complex relationships — the term *complexity*, when referred to a model, can also be used with a similar meaning. While it sounds obvious that a second-degree polynomial has a higher capacity than a linear regression, and can better fit more instances of data, there are relatively few contributions in literature that attempt to provide a more formal framing [23, 24, 25], and these terms are often used in rather intuitive and qualitative statements. As a fast but gross approximation, ML practitioners often assess model capacity by evaluating the number of parameters that can be tuned inside a model. In theory, the best ML model for a task is the one with just enough capacity to properly represent the training data: Models with lower capacity would *underfit*, i.e., they deliver unsatisfying results as they are unable to cope with the complexity of the phenomenon; models with higher capacity would risk to *overfit* and consequently generalize poorly.

A simple depiction of overfitting and underfitting is provided in Figure 4. In practice, unfortunately, it's extremely hard to estimate the capacity necessary to represent a data set; and the solution that many ML practitioners use is — yet again — to apply several techniques with increasing capacity to the data set, until either the gain in fitting stops, or the improvements are not considered important enough to justify an increase in capacity. More interestingly, even estimating effective model capacity is not trivial, as there is evidence from works on deep learning that models with enough parameters to theoretically overfit the training data are actually able to generalize well in real-world case studies [26]. The trade-off between fitting and capacity has been independently explored by different ML communities, with the definition of model-dependent metrics that attempt to take into account both fitting and capacity to assess overall

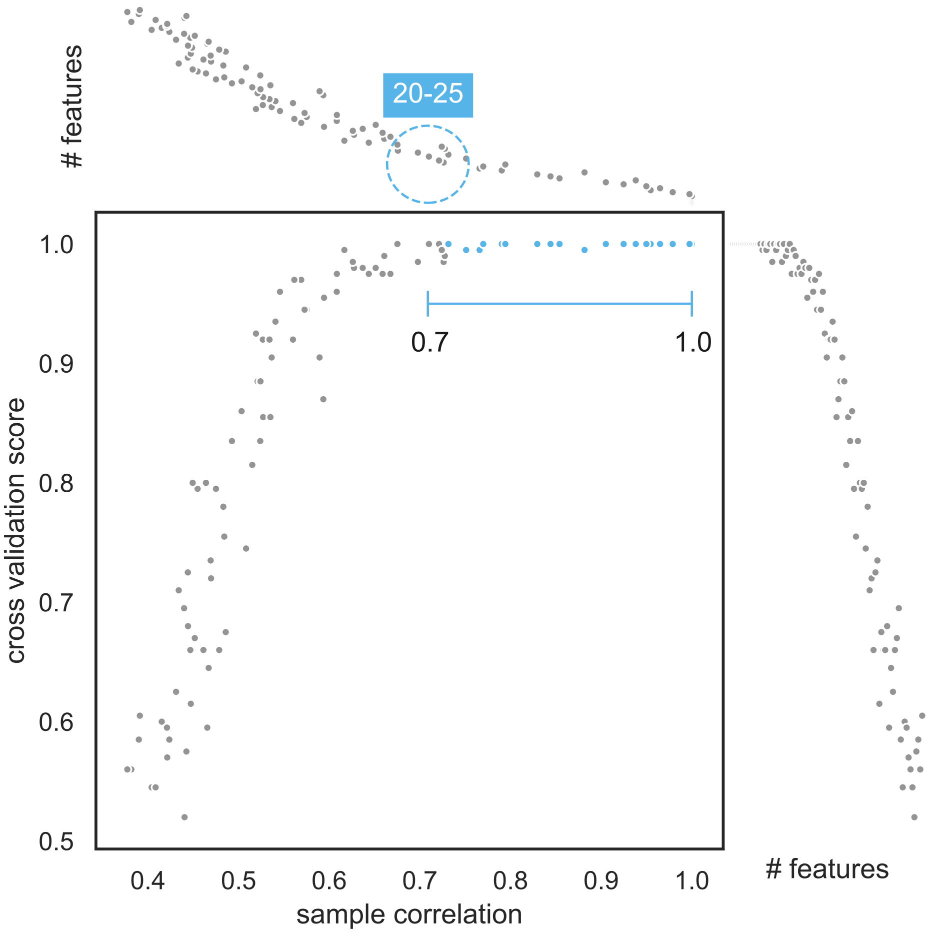


Figure 3: Mutual relationships among generalization ability (cross-validation accuracy), sample correlation, and number of features when all variables are correlated and gaussian. When the number of features is higher than 20–25, samples' correlation drops leading to a dramatic decrease in generalization ability.

quality, to facilitate model selection: A few examples include the Akaike Information Criterion [27], the Bayesian Information Criterion [28], and Pareto-based approaches used mainly in symbolic regression [29].

While evaluating model capacity can help reducing the chance of overfitting, *measuring* overfitting remains far from trivial. Ideal ML models should be able to obtain good predictions even for *unknown* samples of the same problems, but - by definition - the models cannot be tested on unknown data sets. Given this practical need, ML researchers found ways to at least *assess* overfitting, through different techniques. A basic, but extremely efficient technique, is cross-validation (with all its variations, such as leave-one-out cross-validation, stratified cross-validation and the like): The training data is split into k folds of equal size, a ML algorithm is iteratively trained on all folds minus one, and tested on the remaining fold [30]. Analyzing the results of a k-fold cross-validation, for example the average performance, or single instances where the performance on a test fold differs greatly from the others, can provide further insight on the problem characteristics.

As the real objective of evaluating overfitting is to assess a model's capability of extrapolating to unknown instances of the same problem, it is worth it to spend a few words on the meaning of *extrapolation* in supervised ML. As for model capacity, there is an intuitive and imprecise concept of extrapolation, defined as the ability of the model to correctly predict data points that are *considerably different* from the information provided in the training data, but still belong to the same problem. An alternative outlook on extrapolation comes from computational geometry: Interpreting the data points in the training set as points in $\mathbb{R}^f$, where f is the number of features, it is possible to compute its *convex hull*, the smallest polygon that contains all training points. Given the convex hull of the training set, it is then possible

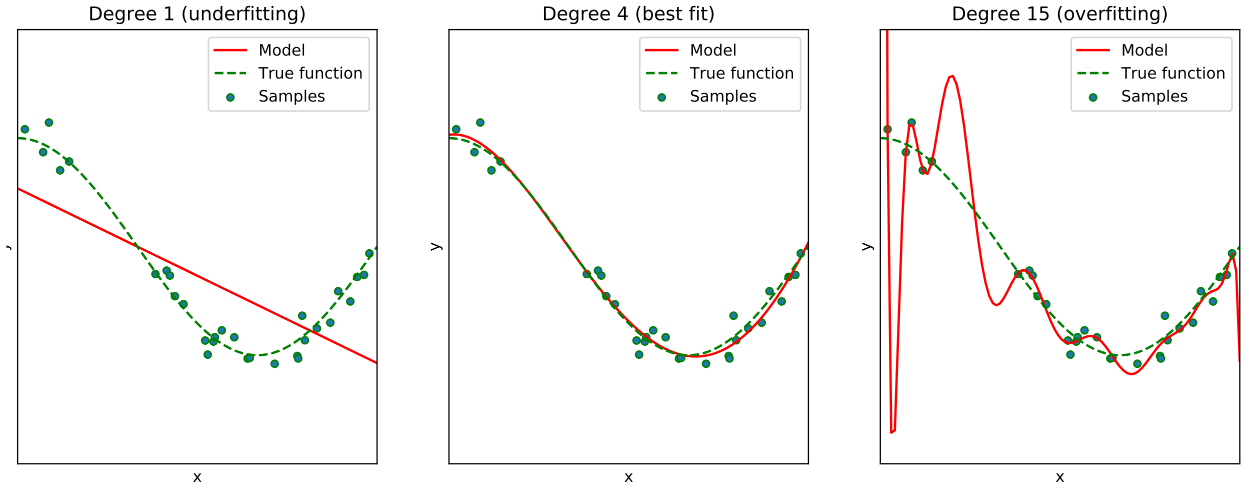


Figure 4: Visual representation of the effect of model capacity on fitting. The original data points can be properly represented by a polynomial of degree 4 (**middle**), so a polynomial regression with lower capacity will underfit (**left**), while a polynomial regression with higher capacity will overfit (**right**).

to assess whether an unseen test data point will fall inside or outside the convex hull. It is reasonable to assume that, for points inside the convex hull, a ML model will *interpolate* known data to obtain a prediction; while the same model will *extrapolate* for test points placed outside of the convex hull. An example is presented in Figure 5. It is important to notice that, depending on the characteristics and the distribution of the training points, this interpretation of interpolation/extrapolation might not correspond to the actual difficulty of predictions for the model. For example, it is possible to imagine a situation where the model will provide better predictions for of a test point outside of the convex hull of the training data, but still relatively close to known points, than for a test point located inside the convex hull of the training data, but in a part of the space where training points are relatively sparse. Still, in most practical scenarios, it is generally harder for models to reliably predict values for test points outside of the convex hull of the training data. A more in-depth discussion on the convex hull is provided in the following Section.

4 Assessing generalization tasks using the convex hull

In an Euclidean space, the convex hull of a set of points $X = \{x_i \in \mathbb{R}^d\}$ is the smallest convex set containing all the points in X. If the number of points n in X is finite (i.e. if X is a matrix $\mathbb{R}^{n\times d}$), then the convex hull forms a convex polytope in $\mathbb{R}^d$. Finding faces or the set of extreme points of such convex polytope is a NP-hard problem [31]. However, checking if a point z lies inside or outside the polytope is much easier, and can be performed in polynomial time [32]. When dealing with ML models, evaluating the convex hull of a training set can provide extra information on unseen test points: If a test point falls inside the convex hull, it is expected that the ML model will probably *interpolate* known points to find the predicted value; vice versa, if a test point falls outside the convex hull of the training set, the ML model will likely be required to *extrapolate* to obtain a prediction.

The problem of checking whether a point z lies inside the polytope of X has a simple linear programming formulation [33]:

$$\begin{aligned} \min_{y} \quad & c^T y \\ \text{s.t.} \quad & Ay = b \\ & c, y \in \mathbb{R}^n. \end{aligned} \tag{2}$$

where:

$$c = \mathbf{0} \qquad A = \begin{bmatrix} X^T \\ \mathbf{1}^T \end{bmatrix} \qquad b = \begin{bmatrix} z \\ 1 \end{bmatrix} \tag{3}$$

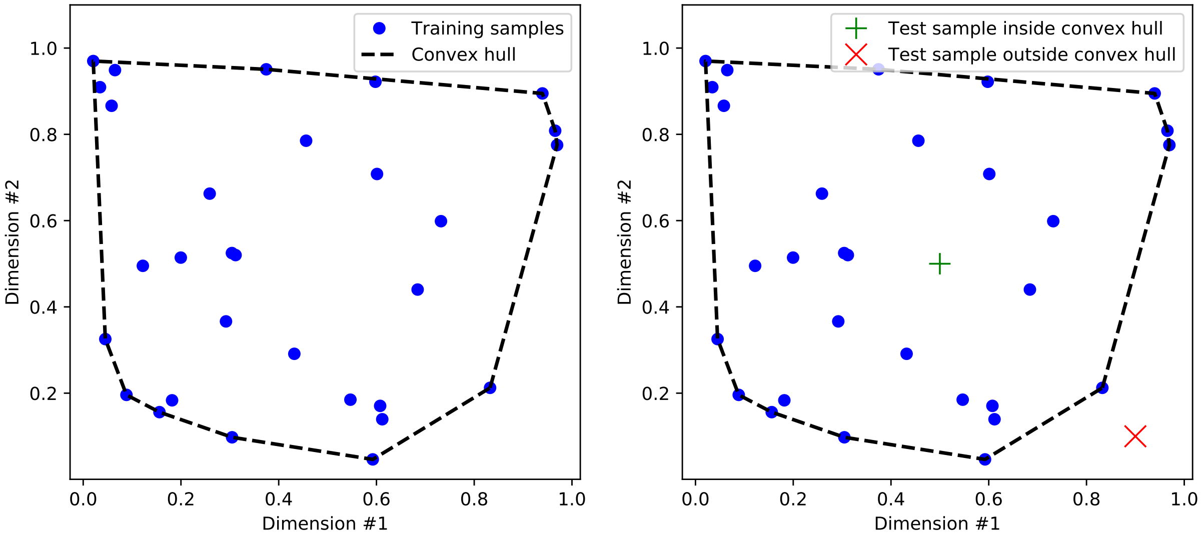


Figure 5: An example of convex hull. The convex hull of a set of training samples is the minimal hyper-polygon (in this case, a 2D polygon) that contains all the data inside its edges (left). For a machine learning model trained on the initial training set, predicting a value for an unseen test point inside the convex hull (in green) probably requires interpolation; but predicting for a test point outside the convex hull requires extrapolation (right).

Such formulation is known as a *Phase I* method [34], as the final goal is not the actual optimization of variable z, but rather checking whether a feasible solution does exist. In such contexts, the cost function can be a constant, as the only objective is satisfying the constraints. The first n constraints impose that the position of z in the feature space must be a combination of the points X:

$$z = \sum_{i=1}^{n} y_i x_i \tag{4}$$

The last constraint imposes that such combination must be convex, which implies, by definition, that the coefficients y_i must sum to 1:

$$\sum_{i=1}^{n} y_i = 1 \tag{5}$$

If the Phase I problem is feasible, point x can be expressed as a convex combination of the set of points X. By definition, this means that point z lies inside the convex polytope of X. On the contrary, if the Phase I problem does not have any feasible solution, then point z lies outside the convex hull of X.

Even the proposed approach exploiting the convex hull can be affected by the curse of dimensionality. The critical point occurs when the dimension of the Euclidean space exceeds the number of observations ($d > n$). In this case the upper bound of the rank of the matrix X corresponds to the number of samples n [35, 36]:

$$rank(X) \leq n \tag{6}$$

Therefore, the maximal number of linearly independent columns of X cannot be higher than n. Whatever the number of dimensions d, the points x_i will lie in a subspace $\mathbb{R}^s$ where $s \leq rank(X) \leq n < d$. As a consequence, the convex hull generated by the set of points x_i will belong to the same subspace. By definition, for $s < d$, the subspace $\mathbb{R}^s$ has *measure zero* in $\mathbb{R}^d$ and can be considered as negligible [37]. Hence, when a new point z is added to the space, it is *almost sure* that it will fall outside negligible subspaces as $\mathbb{R}^s$ [38]. In summary, when $d > n$, the new point z will *almost never* belong to the convex hull of X, making the computation of the convex hull ultimately useless.

5 Modeling generalization as a function of data set characteristics

The objective of this work consists in assessing generalization abilities of machine learning models with empirical experiments. To this aim, first we selected (i) a large set of publicly available data sets for classification, (ii) a representative set of machine learning classifiers, and (iii) a set of data set characteristics. Then, on each data set we perform a cross-validation using ML models and we compute relevant metrics for each fold. Finally, we analyze the relationships between data set characteristics and generalization ability of ML models using both classical correlation metrics, and association models derived through symbolic regression.

In the following subsections, the methodology adopted for selecting the data sets and the corresponding metrics to characterize them are introduced and discussed.

5.1 Data sets

Following the analyses presented in [14] and [15], this study focuses on classification, as it is easier to characterize classification rather than regression data sets, as several of the characteristics analyzed are based on comparing batches of samples belonging to different classes (see *Statistical measures* paragraph in subsection 5.2). Additionally, as the following analysis is partly based on evaluating convex hulls, only data sets with real-valued features are considered.

The data sets examined are acquired from the OpenML repository [39], on online, curated collection that, as the time of writing, includes over 3,100 data sets of different kinds. After selecting only data sets related to classification problems, with real-valued features exclusively, and discarding those containing errors, a total of 109 data sets was ultimately retained for the analysis. The number of samples in the selected collection ranges from 47 to $44,819$, while the number of features spans from 2 to $3,758$. All selected data sets have real-valued features and a discrete target (suited for classification). The mean feature correlation of the data sets is 0.619, with a standard error of the mean of 0.007, the average intrinsic dimensionality ratio is 0.629, with a standard error of the mean of 0.047.

5.2 Data set characterization

As in previous meta-analyses [14, 15], each data set is characterized using metrics, grouped into four categories: simple, statistical, Euclidean, and generalization metrics. For each data set, such measures are computed over a stratified 10-fold cross validation [30].

Simple metrics Simple metrics describe general characteristics of data sets, namely number of samples (n), number of features (d, a.k.a. *dimensionality*), and number of classes (c) [40].

Statistical metrics Statistical metrics assess (i) class differences in feature distributions and shapes, and (ii) relationships between features and classification target.

Levene's test [41] is an inferential statistical test used to assess if a vector of random variables is *homoscedastic*, i.e. if the variance of the random variables is *almost equal*. In the following, Levene's test is used to compare class covariances for each data set feature. The lower the p-value, the higher will be the probability that the class covariances of the feature under study are different [42]. In the following experiments, the score λ collected for each data set is the average of the Levene's p-values of its features.

Pearson's correlation coefficient [43] measures the linear relationship between two variables, providing an indication of the interdependence between pairs of features. The correlations between all pairs of attributes are calculated for each class separately. Since the objective is to evaluate the strength of the relationship and not its sign (positive or negative), the absolute value of the coefficient is used. For each data set, the collected score ρ is the average of the coefficient over all pairs of features and over all classes.

Skewness [44] corresponds to the third standardized moment of a random variable. It indicates the magnitude of the asymmetry of a feature around its mean, yielding an estimate of the feature's departure from normality. The skewness for a class is computed as a weighted average of the skewness of the feature values of its samples. The final skewness score γ represents the average skewness over all classes.

Kurtosis [45] corresponds to the fourth standardized moment of a random variable. It indicates the "thickness" of the tails of a density function. Distributions with kurtosis less than 3 are called *platykurtic*, i.e. they produce fewer and less extreme outliers than the normal distribution. Inversely, distributions with kurtosis higher than 3 are called *leptokurtic* and produce more outliers with respect to the normal distribution. The kurtosis for a class is computed as a weighted average of the kurtosis of the feature values of its samples. The final kurtosis score κ represents the average kurtosis over all classes.

Table 1: Summary of the metrics used to characterize the data sets analyzed in the study.

Metric type	Symbol	Description
Standard metrics	n	Number of samples
	d	Number of features
	c	Number of classes
Euclidean metrics	$\mathfrak{I}$	Intrinsic dimensionality
	$\mathfrak{I}_r$	Intrinsic dimensionality ratio
	$\mathfrak{N}$	Feature noise $(1 - \mathfrak{I}_r)$
	$\mu_{\mathfrak{D}}$	Average sample distance
	$\sigma_{\mathfrak{D}}$	Standard deviation of sample distance
Statistical metrics	λ	Average of Levene's test p-values
	ρ	Average of class-wise feature correlation
	γ	Average of class-wise feature skewness
	κ	Average of class-wise feature kurtosis
	η	Average of feature-target mutual information
Generalization metrics	CI_{train}	Class-imbalance of training samples
	CI_{test}	Class-imbalance of test samples
	T_{in}	Ratio of test samples inside the convex hull
	T_{out}	Ratio of test samples outside the convex hull $(1 - T_{in})$
	CI_{in}	Class-imbalance of test samples inside the convex hull
	CI_{out}	Class-imbalance of test samples outside the convex hull
	$F1_{train}$	$F1$ for the training set
	$F1_{test}$	$F1$ for the whole test set
	$F1_{in}$	$F1$ for the part of the test set inside the convex hull (interpolation ability)
	$F1_{out}$	$F1$ for the part of test set outside the convex hull (extrapolation ability)

Mutual information [46, 47] measures the mutual dependence between two variables. In the following experiments, it is used to estimate the amount of information obtained about the classification target by observing a data set feature. The overall mutual information score η is computed as the average over all features.

Euclidean metrics Euclidean metrics assess the shape of the data manifold.

The *intrinsic dimensionality ratio* provides a normalized estimate of the dimensionality of the data, considering a linear manifold. It is computed counting the number of principal components needed to explain 90% of the variance in the target [48] ($\mathfrak{I}$). The final score $\mathfrak{I}_r$ is normalized over the number of original features .

Feature noise is an estimate of the amount of useless information. Following [14] and [49], the score is computed through the difference between dimensionality (the original number of features) and intrinsic dimensionality. The final score $\mathfrak{N}$ is normalized over the original dimensionality.

The *average sample distance* $\mu_{\mathfrak{D}}$ and the *standard deviation of sample distances* $\sigma_{\mathfrak{D}}$ [50] measure the average pairwise distance between two data set points, and the standard deviation of the resulting distribution, respectively.

Generalization metrics Generalization metrics estimate the nature and hardness of the classification task. Given the convex hull of a training data set, the ratio of test points inside it T_{in} and outside of it T_{out} assesses the type of generalization task ML models are asked to perform. Indeed, if test points often fall inside the convex hull of the training set, it is expected that the ML model will probably *interpolate* known points to find predicted values, most of the time; vice versa, if test points frequently fall outside the convex hull, the ML model will likely be required to recurrently *extrapolate* to obtain predictions. Class-imbalance may also play a role in impairing classifier performance. It has been computed for training and test samples, both inside and outiside the convex hull.

Classification performance estimates how difficult it is for a given classifier to learn from the training set and generalize to the test set. Since the analyzed data sets usually have more than two classes, the $F1$ score [51] is used to measure the classification performance. More specifically, the weighted $F1$ score is adopted, to account for label imbalance. Two scores related to the test set are computed, assessing effectiveness in both interpolation ($F1_{in}$) and extrapolation ($F1_{out}$).

6 Experimental results and discussion

This section describes the experimental results obtained through the analysis of the selected data sets. First, classical linear correlations between the chosen metrics are considered. Then, more complex non-linear models are explored and discussed, outlining the importance of the convex hull. While different algorithms might perform differently on the same data set, testing all possible classifying alternatives is impractical. For this purpose, we selected a *Logistic Regression* (LR) [52] classifier, a *Support Vector Machines* classifier with radial basis function kernel (SVC) [53], and a *Random Forest* classifier with 100 decision tree estimators (RF) [54] as representative classifiers for the following experiments, taking into account their considerable efficiency and their heterogeneous capacity.

All the code and data necessary to reproduce the experiments is available in a public GitHub repository[2].

6.1 Correlations between data set characteristics: sometimes dimensionality not so cursed

Once all the considered metrics described in Sec. 5 have been computed for the 109 selected data sets, classical statistical correlations can be evaluated. In particular, computing Pearson's correlation coefficients results in the matrix presented in Fig. 6. Analyzing the matrix, several predictable correlations can be found: In the following, we will analyze a few of the least immediately obvious.

$\gamma \diamond \kappa$: the mean skewness and mean kurtosis of a dataset are highly correlated (0.84), as both metrics assess the difference in data distribution with respect to a reference Gaussian distribution.

$\mathfrak{I} \diamond \rho$: very often, the higher the correlation between features, the lower the intrinsic dimensionality of a dataset, so as expected the two metrics are anti-correlated (-0.71).

d: dimensionality is positively correlated with $\mathfrak{I}$ (0.85) and negatively correlated with ρ (-0.70). In other words, as the number of features increase, intrinsic dimensionality tends to rise; and, at the same time, features are less likely to be strongly correlated with each other.

$\mu_{\mathfrak{D}} \diamond \rho$: as the number of features/dimensions increases, the average distance between samples also increases, *unless* the additional features are strongly correlated with existing ones. For this reason, as expected, the average distance between samples is negatively correlated with the average correlation between features (-0.76). For the same reason, we find a 0.85 positive correlation between average sample distance and intrinsic dimensionality ($\mathfrak{I}$), and an equally strong positive correlation between $\mu_{\mathfrak{D}}$ and number of features d.

$T_{in} \diamond \mathfrak{I}_r$: the correlation between the ratio of test samples falling inside the convex hull of the training set and the intrinsic dimensionality ratio (0.81) is maybe the less intuitive of the relationships analyzed so far. When the intrinsic dimensionality $\mathfrak{I}$ is lower than the number of dimensions d, training points used to compute the convex hull might actually all lie in a polytope of dimension $d' \approx \mathfrak{I} < d$; the likelihood of test points laying exactly inside this polytope becomes then low, especially when compared to a situation where $\mathfrak{I} \approx d$, and the convex hull of the training set occupies a much larger portion of the feature space. The very same concept is also expressed by the negative correlation between T_{in} and $\mathfrak{N}$ (-0.81), as a higher feature noise also represents a lower effective dimensionality. The same reasoning holds for the correlations $T_{out} - \mathfrak{I}_r$ and $T_{out} - \mathfrak{N}$.

Regarding classifier-specific metrics, such as $F1$ for training and validation, we can observe how LR presents the highest correlation between the two, suggesting a similar behavior for training and unseen samples. SVC and RF, on the other hand, have poorer correlations, as they tend to overfit the training set more, as expected by classifiers with higher capacity. It is important to notice that the strength of this correlation does not imply a poor performance, as RF, with the lowest correlation, shows the highest $F1$ on the test set (see Table 2).

Correlations that are unexpectedly weak in the analysis, those between $F1_{test}$ and all metrics related to dimensionality (d, $\mathfrak{I}$, $\mathfrak{I}_r$), hint at a surprising conclusion: The performance of the ML algorithms on an unseen test set is almost independent from the dimensionality of the data set. This is particularly true for LR ($F1_{test} \diamond \mathfrak{I}_r = 0.1$) and RF ($F1_{test} \diamond \mathfrak{I}_r = 0.9$), while corresponding correlations for SVC are higher, but still not very strong ($F1_{test} \diamond d = -0.43$, $F1_{test} \diamond \mathfrak{I} = -0.37$, but $F1_{test} \diamond \mathfrak{I}_r = 0.28$).

[2] `https://github.com/pietrobarbiero/dataset-characteristics`

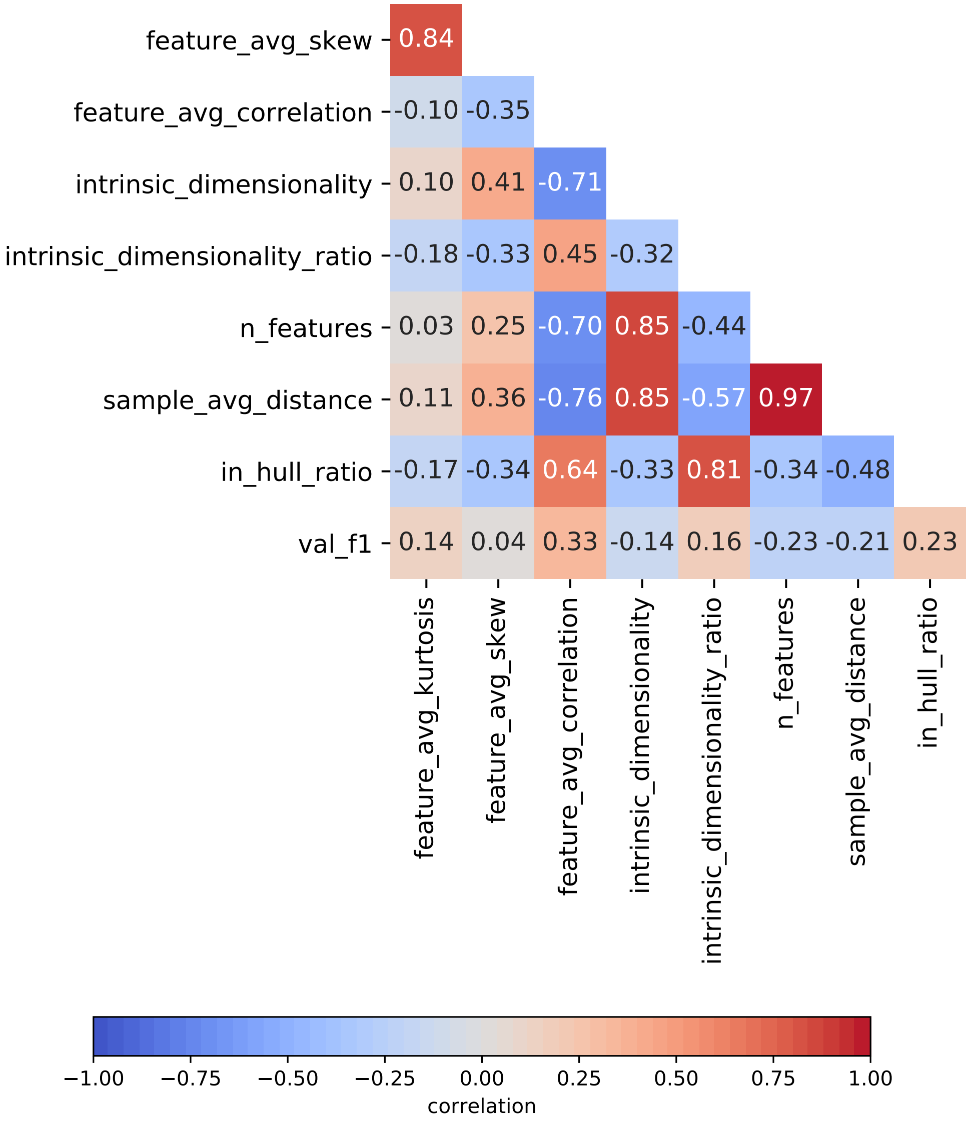


Figure 6: Heatmap showing a selection of the most relevant correlations between data set characteristics. The $F1$ score computed on test samples (*val_f1*) corresponds to the average score over the three ML models used for the experiments, i.e. LR, SVC, and RF.

For the complete correlation matrices between all considered characteristics, see Figs. 9, 10, 11, in Appendix B.

6.2 The key role of the convex hull and association models

In Table 2 we reported $F1$-scores of ML models on training and validation sets. A high difference between $F1_{train}$ and $F1_{test}$ corresponds to overfitting. We record RF exhibiting higher overfitting, while still providing the best generalization performance during validation. On the other hand, the difference between $|F1_{train} - F1_{in}|$ and $|F1_{train} - F1_{out}|$ reveals the discrepancy between interpolation and extrapolation performances, pinpointing the importance of the convex hull in assessing machine learning generalization.

While classical correlation metrics try to optimize the coefficients of models of known structure (e.g., often linear), it might be useful to extend such analysis to models of different structure. In statistical terms, this implies assessing *association*, a relationship between variables more general than correlation: Two or more variables are *associated* if the values of some provide information on the value of the others [55]. To this purpose, we propose the use of *symbolic*

Table 2: Average $F1$-score and standard error of the mean of ML models.

ML model	Sample set	$F1$-score	Δ
LR	$F1_{train}$	0.87 ± 0.01	
	$F1_{test}$	0.79 ± 0.01	0.08 ± 0.03
	$F1_{in}$	0.82 ± 0.01	0.05 ± 0.03
	$F1_{out}$	0.78 ± 0.01	0.09 ± 0.03
SVC	$F1_{train}$	0.86 ± 0.00	
	$F1_{test}$	0.78 ± 0.01	0.08 ± 0.01
	$F1_{in}$	0.85 ± 0.01	0.01 ± 0.01
	$F1_{out}$	0.76 ± 0.01	0.10 ± 0.01
RF	$F1_{train}$	0.99 ± 0.00	
	$F1_{test}$	0.84 ± 0.00	0.15 ± 0.01
	$F1_{in}$	0.88 ± 0.00	0.11 ± 0.01
	$F1_{out}$	0.82 ± 0.00	0.17 ± 0.01

regression [56, 57], a technique that searches the space of mathematical expressions to find the model that best fits a given data set. The irony of using machine learning techniques to analyze the results of a meta-analysis on machine learning is not lost on the authors, but symbolic regression has the advantage of returning completely human-readable models, that can later be interpreted and explained, all the while considering relationships between data set characteristic more complex than just linear correlations. Furthermore, symbolic regression can deliver multiple candidate solutions, models of increasing complexity and fitting, whose meta-analysis can deliver additional information to the user. For this task, the commercial symbolic regression software Eureqa Formulize[3] is employed. All available building blocks for equations were selected, with the exception of those specifically designed for time series analysis. Each run is stopped when the convergence metric of Eureqa crosses the threshold of 90%.

In order to assess generalization abilities of machine learning models, we analyze and compare the results provided by Eureqa in different scenarios. For each classifier, symbolic regression generates a set of Pareto-optimal models, predicting the performance of the classifier in terms of $F1$-score. Pareto optimality is considered as a function of both R^2 (i.e. the accuracy of the formula) and complexity (the number of terms and complexity of formula's building blocks). Among all Pareto-optimal equations proposed by symbolic regression, we manually analyze a selection representing reasonable compromises between fitting and complexity. Eqs. 7-13 represent candidate equations of similar complexity ($C = 9$), describing nonlinear associations between data set characteristics and generalization metrics for each machine learning classifier.

$$F1_{in}^{LR} = 0.857 + (0.959 \cdot F1_{train}^{LR} - 0.831)\ \text{step}(2.38 \cdot \mathfrak{I}_r - 0.0019) \qquad [R^2 = 0.68, C = 9] \tag{7}$$

$$F1_{out}^{LR} = 0.843 \cdot F1_{train}^{LR} + 0.176 \cdot \lambda - 0.040 - 0.097 \cdot \mathfrak{N} \qquad [R^2 = 0.47, C = 9] \tag{8}$$

$$F1_{in}^{SVC} = 1.072 \cdot F1_{train}^{SVC} + 0.021 \cdot \sigma_{\mathfrak{D}} - 0.08 - 0.003 \cdot c \qquad [R^2 = 0.67, C = 9] \tag{9}$$

$$F1_{out}^{SVC} = 0.820 + (1.605 \cdot F1_{train}^{SVC} - 1.389)\ \cos(5.424 + 2.377 \cdot \mathfrak{I}_r) \qquad [R^2 = 0.45, C = 9] \tag{10}$$

$$F1_{in}^{RF} = 7.66 \cdot \rho - 1.822 - 5.266 \cdot \rho^2 \qquad [R^2 = 0.36, C = 9] \tag{11}$$

$$F1_{out}^{RF} = 0.818 + (1.237 \cdot \rho - 0.767)\ \text{erfc}(4.65 \cdot \rho - 2.883) \qquad [R^2 = 0.28, C = 9] \tag{12}$$

[3] Eureqa Formulize is developed by Nutonian, Inc. `https://www.nutonian.com/products/eureqa/`

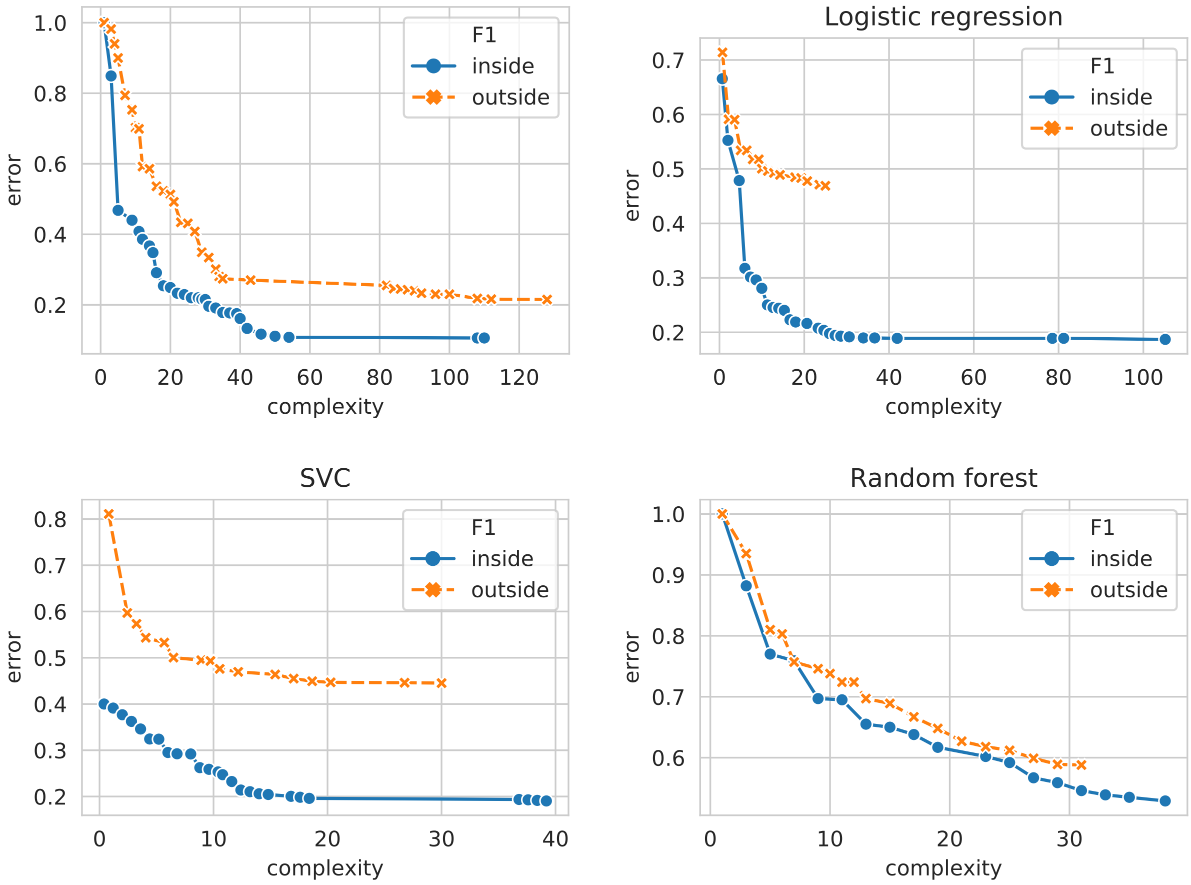


Figure 7: Pareto-optimal models predicting $F1_{in}$ and $F1_{out}$ based on data set characteristics taking into account the results on all ML models (**top-left**). Pareto fronts for each ML model: logistic regression (**top-right**), SVC (**bottom-left**), and random forest (**bottom-right**).

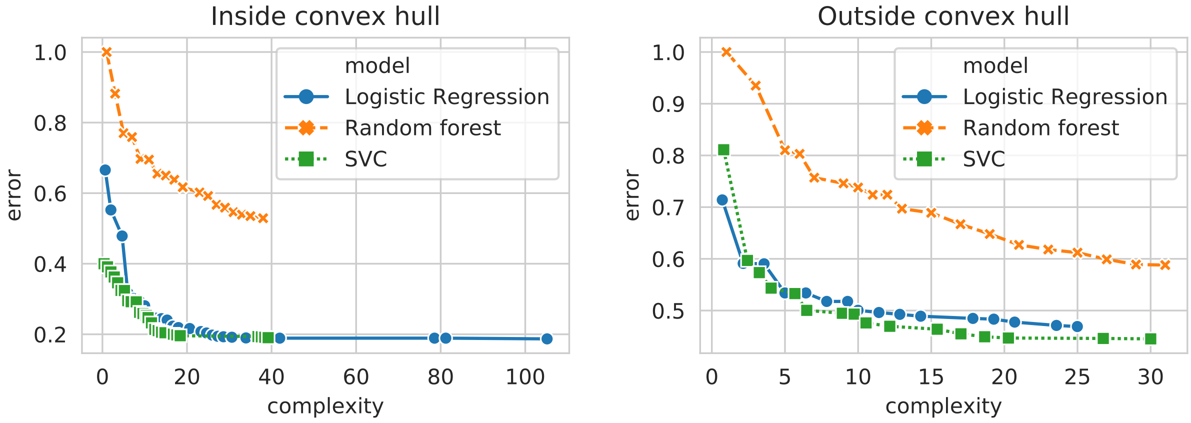


Figure 8: Comparison of Pareto fronts predicting $F1_{in}$ and $F1_{out}$ based on data set characteristics.

$$T_{in} = 0.46 + 0.42 \cdot \text{erf}(4.65 \cdot \rho - 2.20 - 2.98 \cdot \mathfrak{N}) \qquad [R^2 = 0.88, C = 9] \tag{13}$$

Overall, the interpolation ability ($F1_{in}$) of a ML algorithm on a data set can be predicted in a satisfying way using only the data set characteristics that we analyzed in this study. On the contrary, predicting extrapolation ability ($F1_{out}$) from data set characteristics seems much harder, as pointed out by the lower R^2 scores of models for corresponding complexity. Besides, the difference between predictors found for LR or SVC, with respect to RF, is noteworthy: In fact, the generalization ability of the models for the first two algorithms seems strongly associated with their training performance ($F1_{train}$) and either the intrinsic dimensionality ratio ($\mathfrak{I}_r$) or the feature noise ($\mathfrak{N} = 1 - \mathfrak{I}_r$). On the other hand, the test accuracy of RF seems much harder to predict, and associated with the average class-wise correlation among features (ρ) *only*. Moreover, we observe how the ratio of test samples falling inside the convex hull could be easily estimated by considering the class-wise feature correlation (ρ) and either the feature noise ($\mathfrak{N}$) or the intrinsic dimensionality ratio ($\mathfrak{I}_r = 1 - \mathfrak{N}$), confirming the reasoning derived from the previous analysis of the linear correlations: Higher feature noise leads to a lower effective dimensionality, thus reducing the likelihood of test samples falling inside the convex hull.

It is interesting to remark how $F1_{train}$ is the variable that explains most of the variance for LR and SVC models; while the same variable does not appear in models for RF, showing how the generalization ability of this classifier is poorly correlated with its performance on the training set, it is generally harder to predict (lower R^2), and seems to solely depend on the class-wise feature correlation ρ. The model for T_{in} also presents interesting insights, displaying rather high $R^2 = 0.88$ and depending on just two variables, ρ and $\mathfrak{N}$. The negative influence of $\mathfrak{N}$ can be explained intuitively: As the feature noise increases, the intrinsic dimensionality ratio reduces, and thus the points belonging to the convex hull of the training set lie more and more on a polytope of lower dimensionality than the entire feature space, making it more difficult for test points to fall inside its hypervolume. The positive influence of ρ on T_{in} is harder to explain: We speculate that a higher class-wise feature correlation would bring points belonging to the same class closer together in the feature space, as depicted in Fig. 1. Having the data points gathered in a smaller part of the feature space might imply that more test points will fall inside the convex hull of the training set, without necessarily reducing the true dimensionality of the feature space. This is somehow confirmed by the results reported in Fig. 6, where intrinsic dimensionality ratio and average class-wise feature correlation are positively correlated, albeit weakly ($\mathfrak{I}_r \diamond \rho = 0.45$).

We further extend the analysis of symbolic regression results by comparing Pareto fronts of ML models both inside and outside the convex hull to further inspect whether data set characteristics have a significant impact on models' performances. If Pareto front *A* dominates Pareto front *B* it is likely that data set characteristics have a higher impact on ML performances in the first scenario rather than in the second one. The Pareto front analysis is presented in Figs. 7 and 8. In Fig. 7, we analyze the link between data set characteristics and generalization ability by comparing *interpolation* and *extrapolation* results of ML models. In all scenarios the relationship looks stronger when predictions are made inside the convex hull of training samples. However, while this difference is emphasized for LR and SVC, it is less pronounced for RF. In Fig. 8 we further inspect symbolic regression results by comparing Pareto fronts inside and outside the convex hull. Once more, we observe stronger associations between data set characteristics and LR or SVC compared to RF. This means that the impact of data set-specific properties on RF performances is lower, as if the higher capacity of the model makes it more robust.

Highlights of experimental findings

- The convex hull has a key role in assessing ML algorithm generalization, in terms of interpolation and extrapolation abilities.
- The structure and fitting of the association models found for $F1_{in}$ suggests that predicting the interpolation ability of a ML algorithm on a data set might be feasible, using only the data set characteristics that we analyzed in this study. On the contrary, after analyzing the association models obtained for $F1_{out}$, predicting extrapolation abilities from data set characteristics seems much harder.
- ML models with high capacity seem to generalize better (both inside and outside the convex hull), despite their higher potential for overfitting. The phenomenon is similar to what has already been observed in neural networks by Zhang et al. [26].
- The performance of the ML algorithms on an unseen test set seems almost independent from the dimensionality of the data set, thus challenging the common assumption that the *curse of dimensionality* might impair generalization in machine learning.

7 Conclusions

After presenting our analysis of the correlations found on the 109 data sets we analyzed, there is a (rather ironic) question we have to face: how *general* are the results we found? Or, in other words, how well do the *predictions* we perform *extrapolate* to unknown data sets? Frankly speaking, we cannot claim that the correlations described in this work hold for all possible data sets, but the sheer number of different data sets analyzed gives us some hope of generality.

There is, however, a possible bias in the selection of data sets for this study: we focused on openly accessible, curated data sets, that already had to pass several quality checks in order to be hosted on repositories such as OpenML. This pre-selection process might make the data sets considered in this work not representative of all real-world data sets. In other words, usually a data set is uploaded on OpenML because the authors already know that at least one ML technique is going to work well for that specific data set; thus, what we analyzed might be representative only of data sets for which ML techniques work well.

Another possible explanation for the most counter-intuitive correlations we uncovered is that real-world data sets are a subset of all possible data sets. While some general mathematical conclusions, such as the curse of dimensionality, might hold for the set of all hypothetical data sets, they might not necessarily be true for the subset of data that is measured from real phenomena. This observation mirrors the remarks by Lin et al. [58]: In an attempt to explain the effectiveness of neural networks and ML at representing physical phenomena, the authors notice that laws of physics can typically be approximated by a tiny subset of all possible polynomials, of order ranging from 2 to 4; this is a consequence of such phenomena usually being symmetrical when it comes to rotation and translation. As the data sets we analyzed come from either simulations or real-world experiments, their characteristics might lead ML algorithms to represent them more easily than expected.

References

[1] Tom M. Mitchell. *Machine Learning*. McGraw-Hill series in computer science. McGraw-Hill, New York, 1997.

[2] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[3] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[4] Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2722–2730, 2015.

[5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[6] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.

[7] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

[8] Frank Eibe, MA Hall, and IH Witten. The WEKA workbench. online appendix for data mining: Practical machine learning tools and techniques. *Morgan Kaufmann*, 2016.

[9] Datarobot. Automated machine learning for predictive modeling. `https://www.datarobot.com/`, May 2019.

[10] François Chollet et al. Keras. `https://keras.io`, 2015.

[11] Naomi Altman and Martin Krzywinski. The curse (s) of dimensionality. *Nature Methods*, 15:399–400, 2018.

[12] Pietro Barbiero, Andrea Bertotti, Gabriele Ciravegna, Giansalvo Cirrincione, Elio Piccolo, and Alberto Tonda. Understanding cancer phenomenon at gene-expression level by using a shallow neural network chain. In *Italian Workshop on Neural Networks (WIRN 2018)*, volume 6, pages 1–11, 2018.

[13] Naomi Altman and Martin Krzywinski. Points of significance: Ensemble methods: bagging and random forests, 2017.

[14] Dijana Oreski, Stjepan Oreski, and Bozidar Klicek. Effects of dataset characteristics on the performance of feature selection techniques. *Applied Soft Computing*, 52:109–119, mar 2017.

[15] Donald Michie, D. J. Spiegelhalter, C. C. Taylor, and John Campbell, editors. *Machine Learning, Neural and Statistical Classification*. Ellis Horwood, Upper Saddle River, NJ, USA, 1994.

[16] Richard Bellman. Dynamic programming. *Science*, 153(3731):34–37, 1966.

[17] Richard E Bellman. *Adaptive control processes: a guided tour*, volume 2045. Princeton university press, 2015.

[18] Arthur Zimek, Erich Schubert, and Hans-Peter Kriegel. A survey on unsupervised outlier detection in high-dimensional numerical data. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 5(5):363–387, 2012.

[19] Richard D McKelvey and William Zavoina. A statistical model for the analysis of ordinal level dependent variables. *Journal of mathematical sociology*, 4(1):103–120, 1975.

[20] Jake Lever, Martin Krzywinski, and Naomi Altman. Points of significance: model selection and overfitting, 2016.

[21] Jake Lever, Martin Krzywinski, and Naomi Altman. Points of significance: Logistic regression, 2016.

[22] Martin Krzywinski and Naomi Altman. Points of significance: Classification and regression trees, 2017.

[23] Vladimir N Vapnik and A Ya Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. In *Measures of complexity*, pages 11–30. Springer, 2015.

[24] Peter L Bartlett and Shahar Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

[25] Tomaso Poggio, Ryan Rifkin, Sayan Mukherjee, and Partha Niyogi. General conditions for predictivity in learning theory. *Nature*, 428(6981):419, 2004.

[26] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

[27] Hirotugu Akaike. A new look at the statistical model identification. In *Selected Papers of Hirotugu Akaike*, pages 215–222. Springer, 1974.

[28] Gideon Schwarz et al. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464, 1978.

[29] Guido F Smits and Mark Kotanchek. Pareto-front exploitation in symbolic regression. In *Genetic programming theory and practice II*, pages 283–299. Springer, 2005.

[30] Mervyn Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society: Series B (Methodological)*, 36(2):111–133, 1974.

[31] Hans Raj Tiwary. On the hardness of computing intersection, union and minkowski sum of polytopes. *Discrete & Computational Geometry*, 40(3):469–479, 2008.

[32] Daniel A Spielman and Shang-Hua Teng. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *Journal of the ACM (JACM)*, 51(3):385–463, 2004.

[33] Panos M Pardalos, Y Li, and WW Hager. Linear programming approaches to the convex hull problem in rm. *Computers & Mathematics with Applications*, 29(7):23–29, 1995.

[34] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[35] Leonid Mirsky. *An introduction to linear algebra*. Courier Corporation, 2012.

[36] George Mackiw. A note on the equality of the column, and row rank of a matrix. *Mathematics Magazine*, 68(4):285, 1995.

[37] Gerald B Folland. *Real analysis: modern techniques and their applications*. John Wiley & Sons, 2013.

[38] Jean Jacod and Philip Protter. *Probability essentials*. Springer Science & Business Media, 2012.

[39] Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. Openml: Networked science in machine learning. *SIGKDD Explorations*, 15(2):49–60, 2013.

[40] Sophia Daskalaki, Ioannis Kopanas, and Nikolaos Avouris. Evaluation of classifiers for an uneven class distribution problem. *Applied artificial intelligence*, 20(5):381–417, 2006.

[41] Howard Levene. Robust tests for equality of variances: In ingram olkin, harold hotelling, et alia, 1960.

[42] Valentin Amrhein, Fränzi Korner-Nievergelt, and Tobias Roth. The earth is flat (p> 0.05): significance thresholds and the crisis of unreplicable research. *PeerJ*, 5:e3544, 2017.

[43] Karl Pearson. Notes on the history of correlation. *Biometrika*, 13(1):25–45, 1920.

[44] Karl Pearson. Skew variation, a rejoinder. *Biometrika*, 4(1-2):169–212, 1905.

[45] Karl Pearson. Das Fehlergesetz und Seine Verallgemeiner-Ungen Durch Fechner und Pearson. A Rejoinder. *Biometrika*, 4(1-2):169–212, 1905.

[46] LF Kozachenko and Nikolai N Leonenko. Sample estimate of the entropy of a random vector. *Problemy Peredachi Informatsii*, 23(2):9–16, 1987.

[47] Brian C Ross. Mutual information between discrete and continuous data sets. *PloS one*, 9(2):e87357, 2014.

[48] Ian Jolliffe. *Principal component analysis*. Springer, 2011.

[49] Victoria López, Alberto Fernández, Salvador García, Vasile Palade, and Francisco Herrera. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Information sciences*, 250:113–141, 2013.

[50] Howard Anton and Chris Rorres. *Elementary linear algebra: applications version*. John Wiley & Sons, 2010.

[51] Cornelis Joost Van Rijsbergen. Information retrieval. 1979.

[52] Hsiang-Fu Yu, Fang-Lan Huang, and Chih-Jen Lin. Dual coordinate descent methods for logistic regression and maximum entropy models. *Machine Learning*, 85(1-2):41–75, 2011.

[53] John Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

[54] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[55] Naomi Altman and Martin Krzywinski. Association, correlation and causation. *Nature Methods*, 12(10):899–900, September 2015.

[56] John R Koza and John R Koza. *Genetic programming: on the programming of computers by means of natural selection*, volume 1. MIT press, 1992.

[57] Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *science*, 324(5923):81–85, 2009.

[58] Henry W Lin, Max Tegmark, and David Rolnick. Why does deep and cheap learning work so well? *Journal of Statistical Physics*, 168(6):1223–1247, 2017.

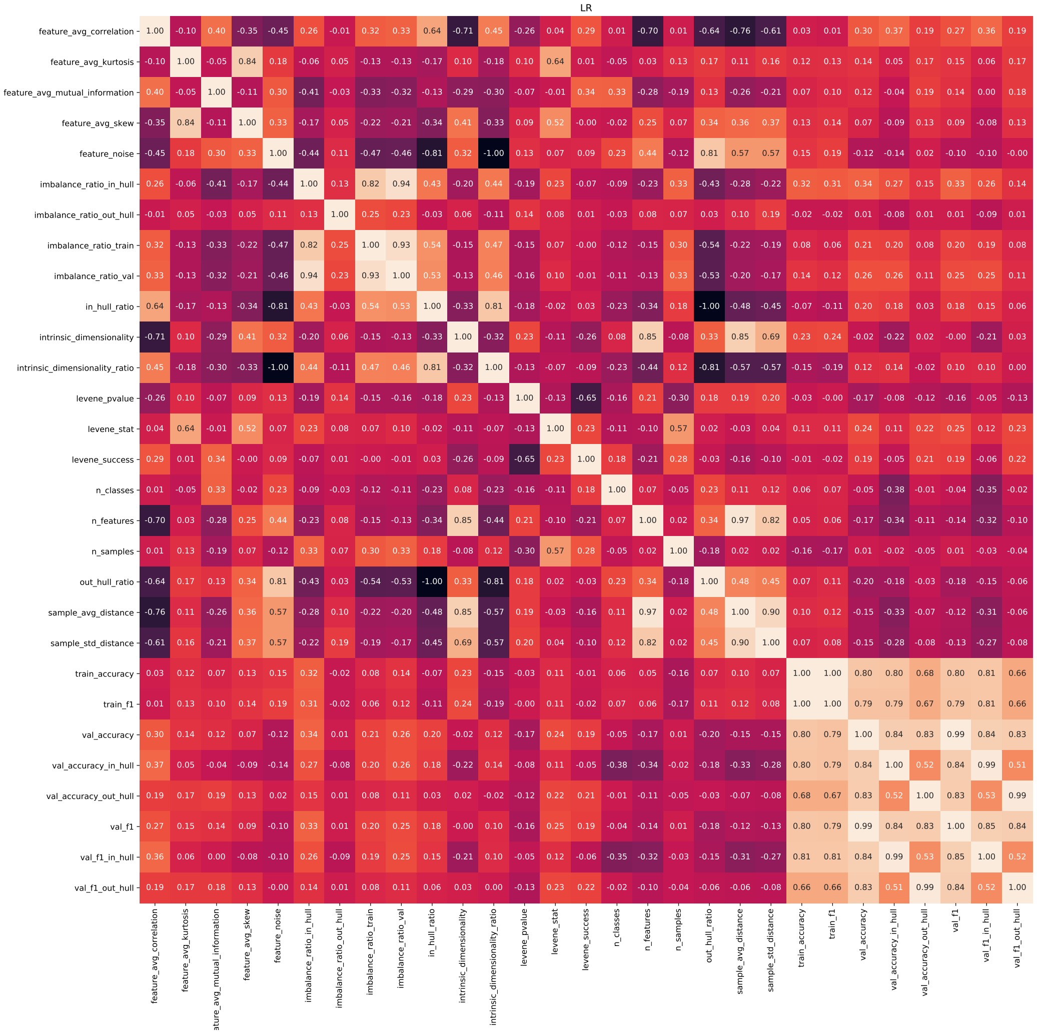
A Code

Listing 1: Python implementation of the convex hull test

```
import numpy as np
from scipy.optimize import linprog

def convex_hull_check(X_train, x_test):
    n_points = len(X_train)
    c = np.zeros(n_points)
    A = np.r_[X_train.T, np.ones((1, n_points))]
    b = np.r_[x_test, np.ones(1)]
    lp = linprog(c, A_eq=A, b_eq=b)
    return lp.success
```

B Full correlation matrices



	feature_avg_correlation	feature_avg_kurtosis	ature_avg_mutual_information	feature_avg_skew	feature_noise	imbalance_ratio_in_hull	imbalance_ratio_out_hull	imbalance_ratio_train	imbalance_ratio_val	in_hull_ratio	intrinsic_dimensionality	intrinsic_dimensionality_ratio	levene_pvalue	levene_stat	levene_success	n_classes	n_features	n_samples	out_hull_ratio	sample_avg_distance	sample_std_distance	train_accuracy	train_f1	val_accuracy	val_accuracy_in_hull	val_accuracy_out_hull	val_f1	val_f1_in_hull	val_f1_out_hull
feature_avg_correlation	1.00	-0.10	0.40	-0.35	-0.45	0.26	-0.01	0.32	0.33	0.64	-0.71	0.45	-0.26	0.04	0.29	0.01	-0.70	0.01	-0.64	-0.76	-0.61	0.03	0.01	0.30	0.37	0.19	0.27	0.36	0.19
feature_avg_kurtosis	-0.10	1.00	-0.05	0.84	0.18	-0.06	0.05	-0.13	-0.13	-0.17	0.10	-0.18	0.10	0.64	0.01	-0.05	0.03	0.13	0.17	0.11	0.16	0.12	0.13	0.14	0.05	0.17	0.15	0.06	0.17
feature_avg_mutual_information	0.40	-0.05	1.00	-0.11	0.30	-0.41	-0.03	-0.33	-0.32	-0.13	-0.29	-0.30	-0.07	-0.01	0.34	0.33	-0.28	-0.19	0.13	-0.26	-0.21	0.07	0.10	0.12	-0.04	0.19	0.14	0.00	0.18
feature_avg_skew	-0.35	0.84	-0.11	1.00	0.33	-0.17	0.05	-0.22	-0.21	-0.34	0.41	-0.33	0.09	0.52	-0.00	-0.02	0.25	0.07	0.34	0.36	0.37	0.13	0.14	0.07	-0.09	0.13	0.09	-0.08	0.13
feature_noise	-0.45	0.18	0.30	0.33	1.00	-0.44	0.11	-0.47	-0.46	-0.81	0.32	-1.00	0.13	0.07	0.09	0.23	0.44	-0.12	0.81	0.57	0.57	0.15	0.19	-0.12	-0.14	0.02	-0.10	-0.10	-0.00
imbalance_ratio_in_hull	0.26	-0.06	-0.41	-0.17	-0.44	1.00	0.13	0.82	0.94	0.43	-0.20	0.44	-0.19	0.23	-0.07	-0.09	-0.23	0.33	-0.43	-0.28	-0.22	0.32	0.31	0.34	0.27	0.15	0.33	0.26	0.14
imbalance_ratio_out_hull	-0.01	0.05	-0.03	0.05	0.11	0.13	1.00	0.25	0.23	-0.03	0.06	-0.11	0.14	0.08	0.01	-0.03	0.08	0.07	0.03	0.10	0.19	-0.02	-0.02	0.01	-0.08	0.01	0.01	-0.09	0.01
imbalance_ratio_train	0.32	-0.13	-0.33	-0.22	-0.47	0.82	0.25	1.00	0.93	0.54	-0.15	0.47	-0.15	0.07	-0.00	-0.12	-0.15	0.30	-0.54	-0.22	-0.19	0.08	0.06	0.21	0.20	0.08	0.20	0.19	0.08
imbalance_ratio_val	0.33	-0.13	-0.32	-0.21	-0.46	0.94	0.23	0.93	1.00	0.53	-0.13	0.46	-0.16	0.10	-0.01	-0.11	-0.13	0.33	-0.53	-0.20	-0.17	0.14	0.12	0.26	0.26	0.11	0.25	0.25	0.11
in_hull_ratio	0.64	-0.17	-0.13	-0.34	-0.81	0.43	-0.03	0.54	0.53	1.00	-0.33	0.81	-0.18	-0.02	0.03	-0.23	-0.34	0.18	-1.00	-0.48	-0.45	-0.07	-0.11	0.20	0.18	0.03	0.18	0.15	0.06
intrinsic_dimensionality	-0.71	0.10	-0.29	0.41	0.32	-0.20	0.06	-0.15	-0.13	-0.33	1.00	-0.32	0.23	-0.11	-0.26	0.08	0.85	-0.08	0.33	0.85	0.69	0.23	0.24	-0.02	-0.22	0.02	-0.00	-0.21	0.03
intrinsic_dimensionality_ratio	0.45	-0.18	-0.30	-0.33	-1.00	0.44	-0.11	0.47	0.46	0.81	-0.32	1.00	-0.13	-0.07	-0.09	-0.23	-0.44	0.12	-0.81	-0.57	-0.57	-0.15	-0.19	0.12	0.14	-0.02	0.10	0.10	0.00
levene_pvalue	-0.26	0.10	-0.07	0.09	0.13	-0.19	0.14	-0.15	-0.16	-0.18	0.23	-0.13	1.00	-0.13	-0.65	-0.16	0.21	-0.30	0.18	0.19	0.20	-0.03	-0.00	-0.17	-0.08	-0.12	-0.16	-0.05	-0.13
levene_stat	0.04	0.64	-0.01	0.52	0.07	0.23	0.08	0.07	0.10	-0.02	-0.11	-0.07	-0.13	1.00	0.23	-0.11	-0.10	0.57	0.02	-0.03	0.04	0.11	0.11	0.24	0.11	0.22	0.25	0.12	0.23
levene_success	0.29	0.01	0.34	-0.00	0.09	-0.07	0.01	-0.00	-0.01	0.03	-0.26	-0.09	-0.65	0.23	1.00	0.18	-0.21	0.28	-0.03	-0.16	-0.10	-0.01	-0.02	0.19	-0.05	0.21	0.19	-0.06	0.22
n_classes	0.01	-0.05	0.33	-0.02	0.23	-0.09	-0.03	-0.12	-0.11	-0.23	0.08	-0.23	-0.16	-0.11	0.18	1.00	0.07	-0.05	0.23	0.11	0.12	0.06	0.07	-0.05	-0.38	-0.01	-0.04	-0.35	-0.02
n_features	-0.70	0.03	-0.28	0.25	0.44	-0.23	0.08	-0.15	-0.13	-0.34	0.85	-0.44	0.21	-0.10	-0.21	0.07	1.00	0.02	0.34	0.97	0.82	0.05	0.06	-0.17	-0.34	-0.11	-0.14	-0.32	-0.10
n_samples	0.01	0.13	-0.19	0.07	-0.12	0.33	0.07	0.30	0.33	0.18	-0.08	0.12	-0.30	0.57	0.28	-0.05	0.02	1.00	-0.18	0.02	0.02	-0.16	-0.17	0.01	-0.02	-0.05	0.01	-0.03	-0.04
out_hull_ratio	-0.64	0.17	0.13	0.34	0.81	-0.43	0.03	-0.54	-0.53	-1.00	0.33	-0.81	0.18	0.02	-0.03	0.23	0.34	-0.18	1.00	0.48	0.45	0.07	0.11	-0.20	-0.18	-0.03	-0.18	-0.15	-0.06
sample_avg_distance	-0.76	0.11	-0.26	0.36	0.57	-0.28	0.10	-0.22	-0.20	-0.48	0.85	-0.57	0.19	-0.03	-0.16	0.11	0.97	0.02	0.48	1.00	0.90	0.10	0.12	-0.15	-0.33	-0.07	-0.12	-0.31	-0.06
sample_std_distance	-0.61	0.16	-0.21	0.37	0.57	-0.22	0.19	-0.19	-0.17	-0.45	0.69	-0.57	0.20	0.04	-0.10	0.12	0.82	0.02	0.45	0.90	1.00	0.07	0.08	-0.15	-0.28	-0.08	-0.13	-0.27	-0.08
train_accuracy	0.03	0.12	0.07	0.13	0.15	0.32	-0.02	0.08	0.14	-0.07	0.23	-0.15	-0.03	0.11	-0.01	0.06	0.05	-0.16	0.07	0.10	0.07	1.00	1.00	0.80	0.80	0.68	0.80	0.81	0.66
train_f1	0.01	0.13	0.10	0.14	0.19	0.31	-0.02	0.06	0.12	-0.11	0.24	-0.19	-0.00	0.11	-0.02	0.07	0.06	-0.17	0.11	0.12	0.08	1.00	1.00	0.79	0.79	0.67	0.79	0.81	0.66
val_accuracy	0.30	0.14	0.12	0.07	-0.12	0.34	0.01	0.21	0.26	0.20	-0.02	0.12	-0.17	0.24	0.19	-0.05	-0.17	0.01	-0.20	-0.15	-0.15	0.80	0.79	1.00	0.84	0.83	0.99	0.84	0.83
val_accuracy_in_hull	0.37	0.05	-0.04	-0.09	-0.14	0.27	-0.08	0.20	0.26	0.18	-0.22	0.14	-0.08	0.11	-0.05	-0.38	-0.34	-0.02	-0.18	-0.33	-0.28	0.80	0.79	0.84	1.00	0.52	0.84	0.99	0.51
val_accuracy_out_hull	0.19	0.17	0.19	0.13	0.02	0.15	0.01	0.08	0.11	0.03	0.02	-0.02	-0.12	0.22	0.21	-0.01	-0.11	-0.05	-0.03	-0.07	-0.08	0.68	0.67	0.83	0.52	1.00	0.83	0.53	0.99
val_f1	0.27	0.15	0.14	0.09	-0.10	0.33	0.01	0.20	0.25	0.18	-0.00	0.10	-0.16	0.25	0.19	-0.04	-0.14	0.01	-0.18	-0.12	-0.13	0.80	0.79	0.99	0.84	0.83	1.00	0.85	0.84
val_f1_in_hull	0.36	0.06	0.00	-0.08	-0.10	0.26	-0.09	0.19	0.25	0.15	-0.21	0.10	-0.05	0.12	-0.06	-0.35	-0.32	-0.03	-0.15	-0.31	-0.27	0.81	0.81	0.84	0.99	0.53	0.85	1.00	0.52
val_f1_out_hull	0.19	0.17	0.18	0.13	-0.00	0.14	0.01	0.08	0.11	0.06	0.03	0.00	-0.13	0.23	0.22	-0.02	-0.10	-0.04	-0.06	-0.06	-0.08	0.66	0.66	0.83	0.51	0.99	0.84	0.52	1.00

Figure 9: Correlations between dataset characteristics using logistic regression.

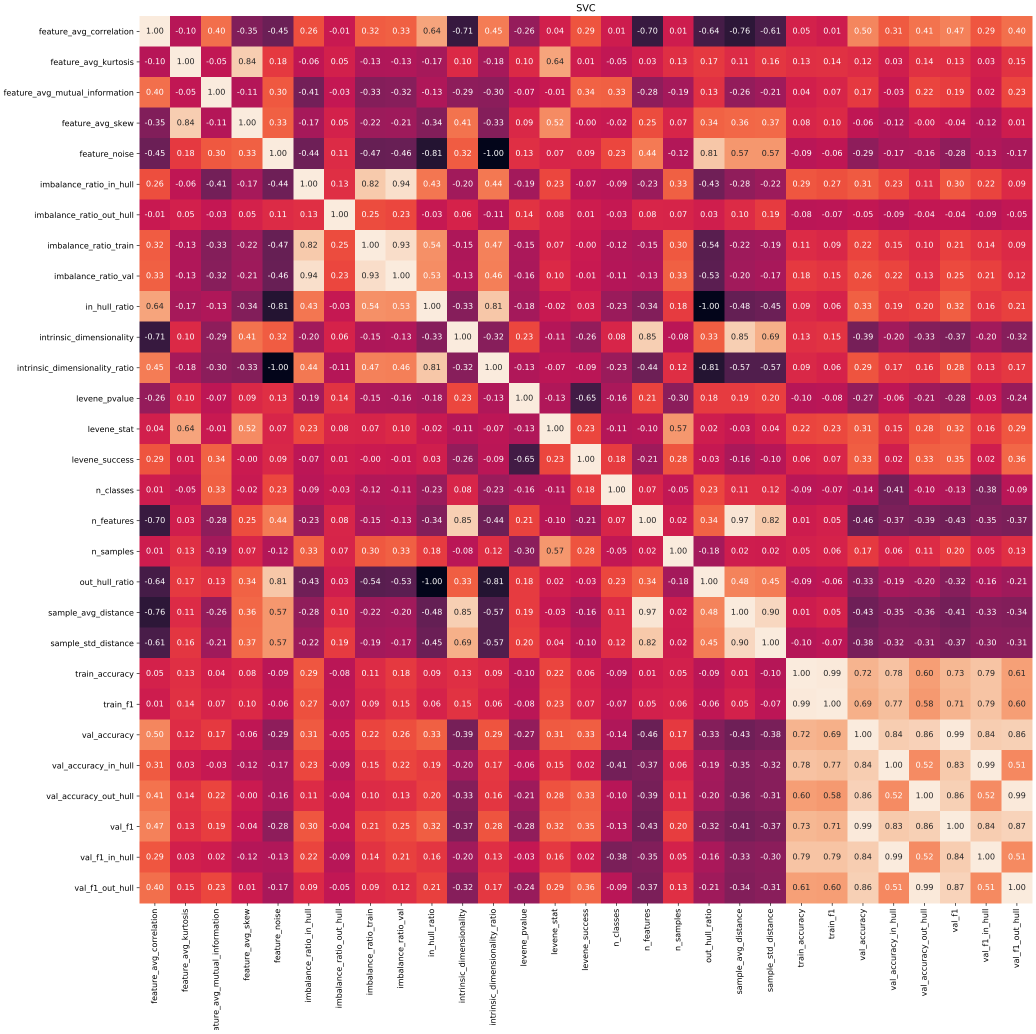


Figure 10: Correlations between dataset characteristics using SVC.

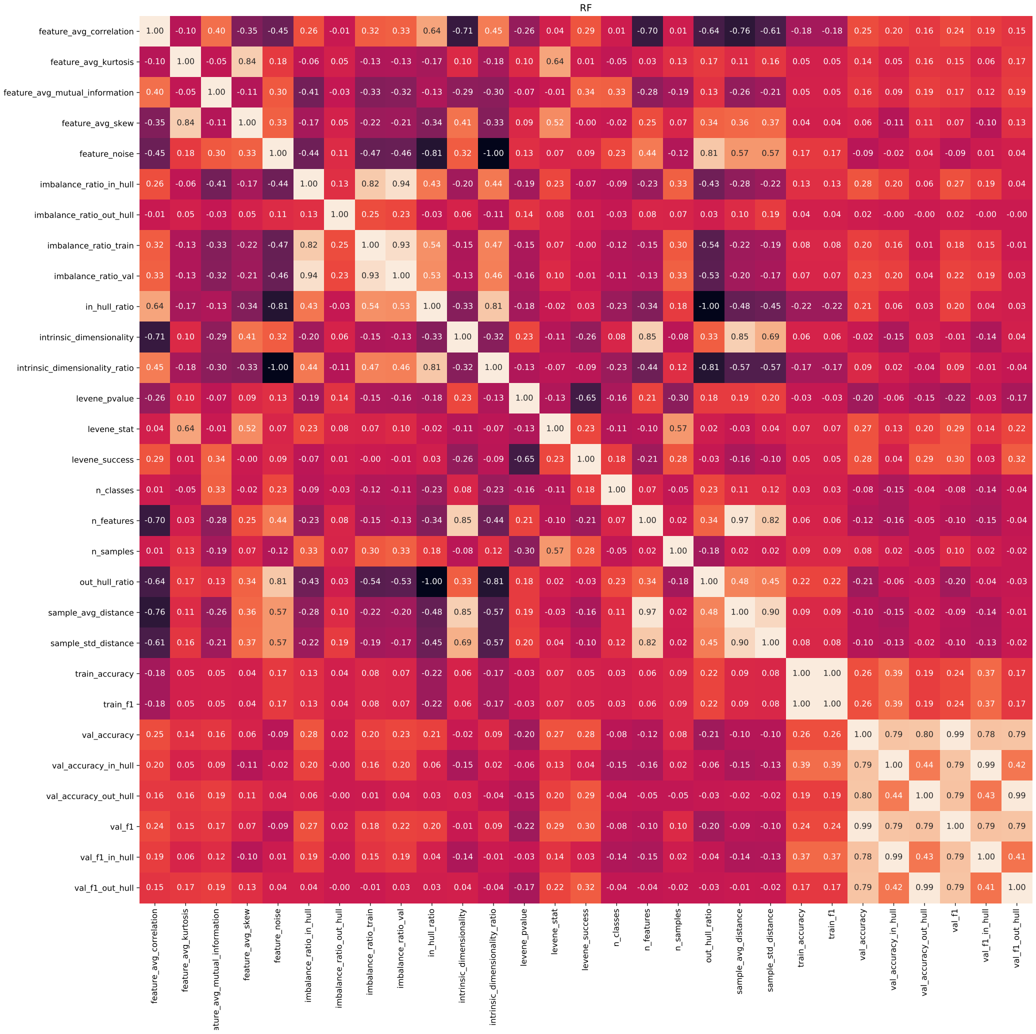


Figure 11: Correlations between dataset characteristics using random forest.