DATA EFFICIENCY OF CLASSIFICATION STRATEGIES FOR CHEMICAL AND MATERIALS DESIGN

Quinn M. Gallagher
Chemical and Biological Engineering
Princeton University
Princeton, NJ 08544
qg1361@princeton.edu

Michael A. Webb
Chemical and Biological Engineering
Princeton University
Princeton, NJ 08544
mawebb@princeton.edu

ABSTRACT

Active learning and design-build-test-learn strategies are increasingly employed to accelerate materials discovery and characterization. Many data-driven materials design campaigns require that materials are synthesizable, stable, soluble, recyclable, or non-toxic. Lack of knowledge about these constraints can reduce efficiency by producing unsatisfactory samples. Acquiring this knowledge during the design campaign is inefficient, and many materials constraints transcend specific design objectives. However, there is no consensus on the most data-efficient algorithm for classifying whether a material satisfies a constraint. To address this gap, we comprehensively compare the performance of 100 strategies for classifying chemical and materials behavior. Performance is assessed across 31 classification tasks sourced from the literature in chemical and materials science. From these results, we recommend best practices for building data-efficient classifiers, showing the neural network- and random forest-based active-learning algorithms are most efficient across tasks. We also show that classification task complexity can be quantified by task metafeatures, most notably the noise-to-signal ratio. Overall, this work provides a comprehensive survey of data-efficient classification strategies, identifies attributes of top-performing strategies, and suggests avenues for further study.

1 Introduction

Computational workflows are increasingly used to design materials more efficiently than the trial-and-error nature of traditional laboratory discovery [1, 2, 3]. These workflows often utilize high-throughput screening or design-of-experiments strategies applied to automated laboratory equipment and computational models. Examples include the design of π-conjugated peptides for organic electronics [4], metal-organic frameworks for gas separation [5], small molecules for organic light-emitting diodes [6], phase-separating intrinsically disordered proteins [7], and many others [8, 9, 10, 11, 12, 13, 14, 15]. Using active learning and Bayesian optimization (AL/BO), these campaigns have produced materials with desired figures of merit despite characterizing a small fraction of the possible design space. Such workflows promise to drastically accelerate materials discovery in increasingly complex spaces.

Materials optimization often targets a constrained domain. Consequently, resources can be wasted on candidates unsuitable for further characterization. Common constraints on materials domains include synthesizability, unwanted phase behavior, instability, and toxicity. For example, when surveying a polymer library for enzyme-stabilizing candidates, Tamasi *et al.* encountered phase-separating or aggregating polymers unsuitable for physical assays with the target enzyme [9]. Likewise, Körbel *et al.* surveyed 1,276 hybrid organic-inorganic halide perovskites of the form $A^+B^{2+}X_3^-$, from which only 203 compounds were considered stable for further density functional theory calculations [16]. An *et al.* sought to find peptide sequences that would form condensed phases and disparate dynamical properties [7], yet no phase-separating systems were identified in an initial survey of 1,266 peptides listed in the DisProt database [17]. Ideally, such behavior would be known or predicted from the outset and incorporated into the data-selection process for any given design campaign. Additionally, knowledge of materials classification can be applied across varied design objectives. Therefore, a viable strategy is to allocate a portion of the resource budget to accurately classify viability within a materials domain, avoiding wasted resources on unsuitable candidates. To maximize resource use,

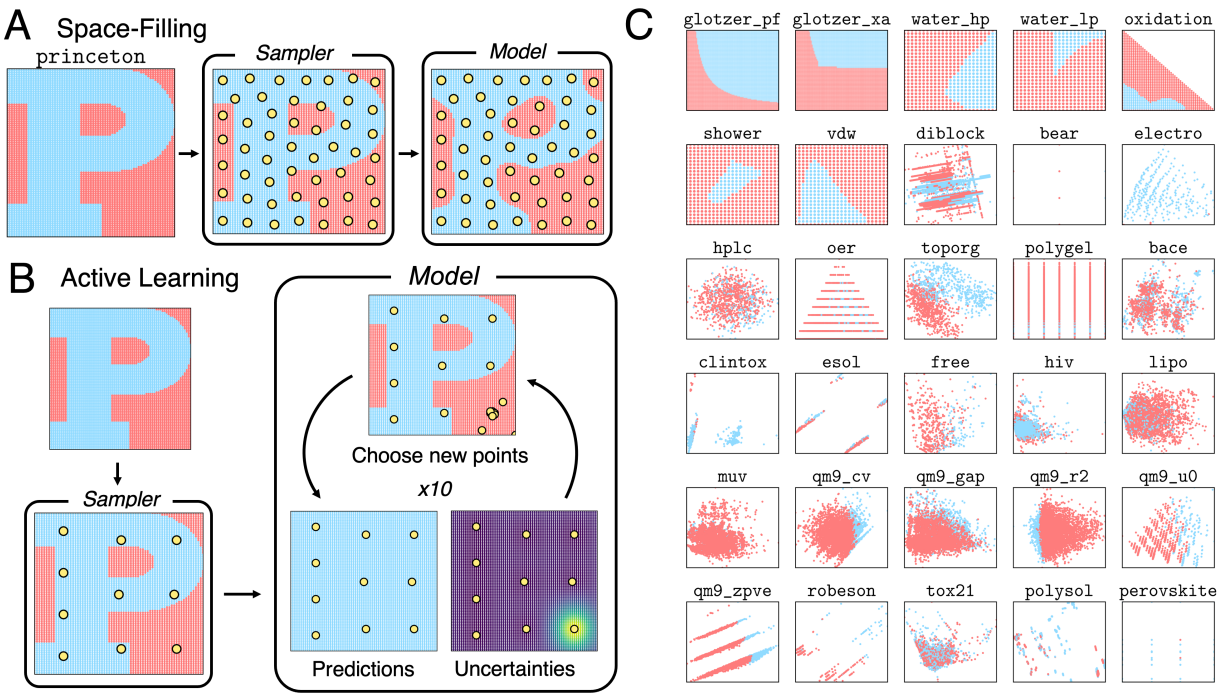

Figure 1: Overview of data-selection strategies and datasets. **(A)** Schematic of a space-filling algorithm applied to the `princeton` task. In space-filling, a one-shot selection of points chosen by the sampler is used to train the model. **(B)** Schematic of an active-learning algorithm applied to the `princeton` task. In active learning, the sampler chooses a set of points to initiate active learning. The model is then trained and used to compute uncertainties on the entire domain, which guide selection of the next batch of points. This process is continued for ten iterations. **(C)** A visual depiction of all other tasks considered in this study. Tasks with more than two features are visualized in two dimensions using principal component analysis. In all panels, red and blue distinguish the two class labels.

it is desirable to use a data-selection strategy and classification algorithm that achieves the highest accuracy with the fewest measurements.

Numerous and varied classification schemes can be found across the literature. Terayama *et al.* used uncertainty-based active learning to build phase diagrams of H_2O, glass-ceramic glazes, block copolymers, and more [18, 19, 20] using label propagation, a semi-supervised machine learning model [21]. Citing the computational expense of the label propagation algorithm, Telleria-Allika *et al.* used a random forest-based active learning scheme to build magnetic and covalency phase diagrams for few electron Hooke atoms and helium dimers [22]. Dai and Glotzer used active learning based on a Gaussian process least-squares classifier and a novel acquisition function to learn the phase diagram of active Brownian particles (ABPs) and quasi-crystals [23]. Hickman *et al.* used Gaussian processes to simultaneously classify viability and optimize performance for several materials design tasks, including small molecule drugs and perovskites [24]. Focusing on the low-data regime, Bhat and Kitchin used heuristics, rather than active learning, to identify classification boundaries in several engineering problems, asserting that active learning would be ineffective in their low-data limit [25]. Other works have continued the trend of applying novel active learning schemes to custom design tasks [26, 27, 28, 29, 30]. The diversity of considered tasks and proposed algorithms indicates no consensus on what constitutes an optimal approach or how to select reasonable strategies.

Here, we investigate the performance of various algorithms across a set of 31 classification tasks in chemical and materials science. From these results, we identify the algorithms that perform optimally and the attributes that lead to maximum data efficiency. We also explore approaches to building classification algorithms that are robust to task variation. To explain algorithm performance across tasks, we demonstrate that metafeatures (*i.e.*, properties of classification tasks) predict an algorithm's performance, with a few metafeatures strongly correlating with classification accuracy regardless of algorithm choice. Through this study, we identify best practices for selecting data-efficient classification algorithms and explain why these practices improve performance.

2 Overview of strategies

We consider space-filling and active-learning algorithms, both of which rely on a *sampler* and a *model*. Space-filling algorithms (Figure 1A) use the sampler to select a batch of points. The model is then trained on this batch and used to make predictions on the rest of the task domain. The accuracy of the algorithm is measured by comparing the predicted labels of the model to the ground truth labels. In this way, space-filling algorithms rely on a one-shot data selection scheme. active-learning algorithms (Figure 1B) use the sampler to select an initial batch of points. The model is trained on these points and used to compute the predicted labels and uncertainties of all points in the task domain. A new batch of points is chosen based on those points which are most uncertain. Model training, uncertainty calculations, and batch selection are repeated until the total allowable number of points is reached. The accuracy of the algorithm is measured by comparing the predicted labels of the final model to the ground truth labels. In this way, active-learning algorithms rely on an iterative, rather than one-shot, data selection scheme. Considering multiple samplers and models produces a combinatorial space of 100 space-filling and active-learning algorithms that are applied to a diverse set of 31 binary classification tasks (mostly) relevant to chemical and materials science. The tasks are visualized in Figure 1C. Further details on the tasks, samplers, models, batch selection schemes, and accuracy metrics are provided in **Methods**.

3 Methods

Tasks. Task domains vary in size (285-10,000) and dimensionality (2-14). For active-learning algorithms, batch sizes are chosen for each classification task so that less than 10% of the task domain and a maximum of 100 points is sampled. A description of the included tasks and their sources is included in Table 1. Briefly, tasks include the classification of phase behavior in active Brownian particles (ABPs), polymer systems, and water; figures of merit in metal alloys, catalysts, and perovskites; performance of experimental equipment for high-performance liquid chromatography and additive manufacturing; and small-molecule properties like aqueous solubility, band gap, heat capacity, and others.

Some classification tasks are prepared from datasets with continuous properties. For these datasets, the task is to classify elements of the domain with property values below the 20th percentile of the property distribution. Some classification tasks require a molecular representation. For these tasks, molecules are represented as the ten most informative physico-chemical features calculated by the Mordred descriptor calculator [31] for the given property. The chosen descriptors are selected by training a logistic regression model with an L_1 loss on the full dataset, with molecules represented by all available Mordred descriptors, and keeping the ten descriptors with the largest absolute coefficients. This scheme emulates molecular design campaigns that use a set of expert-informed features as a molecular representation [32]. Viable alternatives to this choice of molecular representation, like molecular fingerprints [33], graphs [34], and physics-informed structural representations [35], are not considered in this study. We further note that there are many feature-selection strategies, and the present approach may not be optimal. Fixing the representation strategy allows us to focus analysis on data-selection; however, understanding tandem data-selection and feature engineering is of future interest.

Samplers. Five samplers are considered for generating complete datasets for space-filling or initial datasets for active learning. These are referred to as *i.* random, *ii.* maximin, *iii.* medoids, *iv.* max entropy, and *v.* Vendi samplers. For demonstrative purposes, Figure 2 shows the points selected by these five samplers on the `princeton` dataset. These samplers represent different data-selection paradigms from the field of "Design of Experiments" [36], including geometry, information theory, and diversity. Common alternatives like Latin hypercube sampling [37] and Sobol sequences [38] are not considered due to their applicability only on (hyper)cubic domains, which differ from the non-cubic domains present in many of the materials spaces considered here. Extension of such approaches may be feasible, in certain scenarios, but not facile. Thus, we restrict our testing to approaches that can be readily applied, irrespective of the input space.

While the random sampler chooses points at random, non-random samplers choose points that optimize a specific metric. Maximin sampling, also called furthest-point sampling, sequentially selects points that maximize the minimum Euclidean distance between the current point and all previously chosen points. A medoids sampler chooses the centroids produced by the k-medoids algorithm, which selects a set of points that minimizes the average squared Euclidean distance between any point in the domain to a point in the sample. A max entropy sampler, a method created by Paiva [39], chooses a maximally informative set of points by sequentially selecting the point in the domain $\mathbf{x}^*$ to solve

$$\arg\min_{\mathbf{x}^*} \left[\frac{1}{m+1} \sum_{j=1}^{m} \kappa(\mathbf{z}_j - \mathbf{x}^*) - \frac{1}{N} \sum_{i=1}^{N} \kappa(\mathbf{x}_i - \mathbf{x}^*) \right] \tag{1}$$

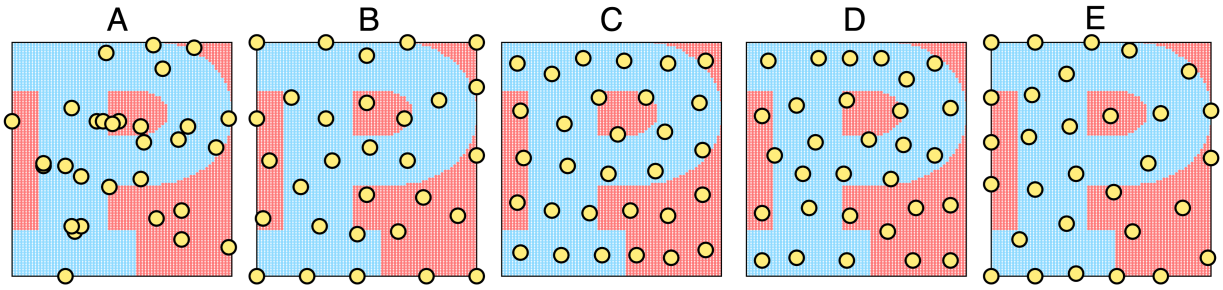

Figure 2: Overview of different sampler algorithms for generating (initial) datasets. Batches of 30 points selected from 6,390 points in the `princeton` dataset using **(A)** random, **(B)** maximin, **(C)** medoids, **(D)** max entropy and (E) Vendi samplers. In all panels, red and blue distinguish the two class labels.

where the set $\{\mathbf{z}_j\}$ are previously chosen points, $\{\mathbf{x}_i\}$ are all points in the domain, and κ is the squared-exponential kernel

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{(2\pi\sigma^2)^{d/2}} \exp\left(-\frac{1}{2\sigma^2}||\mathbf{x}_i - \mathbf{x}_j||^2\right) \tag{2}$$

where d is the dimensionality of the task domain and σ is a bandwidth parameter computed using Silverman's rule of thumb [40]. The Vendi sampler sequentially chooses points that maximize the Vendi score [41], which is a diversity metric computed from the entropy of the eigenvalues of a Gram matrix computed on the domain. For computing the Gram matrix, we use the squared exponential kernel shown in Eq. (2). All methods described here depend on a random seed for the selection of all points (*i.e.,* random), initial guess (*i.e.*, medoids), or initial point (*i.e.,* maximin, max entropy, Vendi), depending on the sampler.

Models. Models include random forests (RFs), gradient boosted decision trees (XGBs), support vector machines (SV), label propagation (LP), neural networks (NNs), Gaussian processes (GPs), and Bayesian kernel density estimation (BKDE). Models predict labels and uncertainties on the task domain. For models without inherent uncertainty estimates (XGBs and NNs), ensembles of models are built using bootstrap aggregation to calculate uncertainties. Gaussian processes are implemented as both least-squares classifiers (GPRs) and as classifiers with a Bernoulli likelihood (GPCs) using both isotropic and anisotropic (ARD) squared exponential kernels [42]. The uncertainties of GPRs use the scheme developed by Dai and Glotzer in Ref. [23]. All models are subject to hyperparameter tuning after each new batch of data was selected. A full description of the chosen models and hyperparameter tuning is available in the Supporting Information (SI) (see Section S1).

The BKDE model is inspired by the Gryffin [43] and Phoenics [44] algorithms. The kernel density of each point is measured using the outputs of a Bayesian autoencoder fit to the training data. Specifically, the kernel density at point $\mathbf{x}$ due to a measured point $\mathbf{x}_k$ can be written as:

$$\rho_k(\mathbf{x}) = \left\langle \sqrt{\frac{\tau_n}{2\pi}} \exp\left[-\frac{\tau_n}{2}(\mathbf{x} - \mathbf{x}_{\text{pred}}(\theta; \mathbf{x}_k))^2\right] \right\rangle_{\text{BNN}} \tag{3}$$

where τ_n is a learnable bandwidth parameter with a prior dependent on the number of measured points, n, and $\mathbf{x}_{\text{pred}}(\theta; \mathbf{x}_k)$ is the prediction of a Bayesian autoencoder with sample parameters θ and input $\mathbf{x}_k$. The average $\langle\rangle_{\text{BNN}}$ refers to the average computed by sampling this value from the Bayesian neural network. The reader is directed to Ref. [44] for additional explanation. Using these kernel density estimates, probabilities for each class can be calculated using the following equation:

$$p_i(\mathbf{x}) = \frac{\sum_{\mathbf{x}_k \in \mathbf{X}_i} \rho_k(\mathbf{x})}{\sum_{\mathbf{x}_k \in \mathbf{X}} \rho_k(\mathbf{x})} \tag{4}$$

where $p_i(x)$ is the probability that point $\mathbf{x}$ is label i, $\mathbf{X}$ is the task domain, and $\mathbf{X}_i$ are all points in the domain with label i. For a given point, the predicted label is the class with the highest probability, and its uncertainty is the entropy of the probability distribution across all classes. Due to the expense of hyperparameter tuning, implementations with BKDE maintain a fixed architecture consistent with its prior usage [43].

Batch Selection. All active-learning algorithms use the "Kriging believer" scheme to select batches of points [45]. The Kriging believer scheme operates as follows. First, uncertainties are computed across the domain, and the point

Table 1: Overview of classification tasks

Name	Size	Dim.	Domain	Label	Ref.
bace	1,513	10	Small molecules	Inhibition of human β-secretase 1	[46]
bear	1,800	4	3D-printed structures	High mechanical toughness	[47]
clintox	1,480	10	Small molecules	FDA approval	[46]
diblock	5,376	3	Diblock copolymers	Lamellar phase	[48]
electro	285	4	Electrocatalysts	High stability	[49]
esol	1,128	10	Small molecules	Low aqueous solubility	[46]
free	642	10	Small molecules	Low hydration free energy	[46]
glotzer_pf	10,000	2	ABP phase diagram (constant PF)	Phase separating	[23]
glotzer_xa	10,000	2	ABP phase diagram (constant x_A)	Phase separating	[23]
hiv	7,215	10	Small molecules	Active HIV inhibitors	[46]
hplc	1,385	5	HPLC process parameters	Low photodegradation	[50]
lipo	4,200	10	Small molecules	Low lipophilicity	[46]
muv	5,000	10	Small molecules	Toxicity	[46]
oer	2,121	6	OER catalysts	Low overpotential	[51]
oxidation	1,275	2	Ternary alloys	Oxidation susceptibility	[25]
perovskite	1,276	14	Perovskites	Stability	[24]
polygel	9,856	9	Polymethacrylates	Predicted solubility	
polysol	6,524	11	Common polymers and solvents	Solubility	[52]
princeton	6,390	2	Princeton “P"	Inside the “P”	
qm9_cv	6,695	10	Small molecules	Low C_V	[53]
qm9_gap	6,695	10	Small molecules	Low band gap	[53]
qm9_r2	6,695	10	Small molecules	Low spatial extent	[53]
qm9_u0	6,695	10	Small molecules	Low internal energy at 0K	[53]
qm9_zpve	6,695	10	Small molecules	Low ZPVE	[53]
robeson	353	10	Linear homopolymer membranes	Above the 1999 Robeson bound	[54]
shower	625	2	Flow rates	Satisfactory temperature	[25]
toporg	1,342	8	Polymer topologies	Low radius of gyration	[55]
tox21	7,831	10	Small molecules	Toxicity	[46]
vdw	625	2	Thermodynamic conditions	Phase separation	[25]
water_hp	625	2	Thermodynamic conditions (high P)	Ice	[19]
water_lp	625	2	Thermodynamic conditions (low P)	Liquid water	[19]

with the highest uncertainty is added to the training set. The model then assumes its prediction for that point is correct and retrains accordingly. Updated uncertainties are then recomputed on the domain to identify the next point with the greatest uncertainty. This process is repeated until the desired batch size is reached. Hyperparameter tuning is not repeated during retraining with assumed labels.

For BKDE-based active-learning algorithms, a custom batch-selection scheme is used due to the computational expense of refitting BKDE to new data. We define $\hat{\rho}_k(\mathbf{x}) = \rho_k(\mathbf{x})/\rho_{\text{k,max}}$ as the normalized kernel density, so that $\hat{\rho}_k(\mathbf{x}) \in [0, 1]$. $\hat{\rho}_k(\mathbf{x})$ represents the influence of point $\mathbf{x}_k$ on every point $\mathbf{x}$ in the domain with a value between 0 and 1. Before batch selection, the uncertainties of every point in the domain are computed, denoted $u_0(\mathbf{x})$. Batch selection begins by selecting the point with the highest uncertainty, denoted $\mathbf{x}_1$. When this point is selected, $\hat{\rho}_1(\mathbf{x})$ is computed. The uncertainties are then recomputed by reducing their magnitude by a factor proportional to the influence of $\mathbf{x}_1$ at that point, producing a new uncertainty function $u_1(\mathbf{x}) = u_0(\mathbf{x}) * (1 - \hat{\rho}_1(\mathbf{x}))$. By consequence, uncertain points uninfluenced by $\mathbf{x}_1$ remain uncertain, while those near $\mathbf{x}_1$ are less likely to be chosen. The point $\mathbf{x}_2$ that maximizes $u_1(\mathbf{x})$ is then chosen, and the process is repeated until the desired batch size is reached. This method allows for a diverse batch of points to be selected by BKDE-based active-learning algorithms without retraining the model for each acquired point.

Metrics. Classification accuracy is assessed using the Macro F_1 score for its robustness to class imbalance, equal weighting of precision and recall, and use in prior studies [19]. For a given class, the F_1 score is defined as:

$$F_1 = \frac{2(\text{TP})}{2(\text{TP}) + \text{FP} + \text{FN}} \tag{5}$$

where ‘TP’, ‘FP’, and ‘FN’ respectively denote the number of true positives, false positives, and false negatives. The Macro F_1 score is calculated by taking the average of F_1 scores computed for each class.

For any given task, what differentiates "good" from "bad" Macro F_1 scores can be ambiguous. Inspired by the use of random selection as a baseline in optimization literature [56], we define a new metric, ξ, as the number of randomly selected points a nearest neighbor classifier requires to achieve the same Macro F_1 score as the specified algorithm. We further define ξ_{max} as the maximum ξ achieved by any algorithm on the task. Then, ξ/ξ_{max} describes how close an algorithm is to the best performance on a given task. Metrics like ξ and ξ/ξ_{max} quantify efficiency in terms of resources saved by employing a given algorithm compared to a naive approach.

Metafeatures analysis. Algorithm performance on tasks is predicted based on metafeatures of the task. Metafeatures include basic characteristics of a classification task (*e.g.*, dimensionality, dataset size, class proportion), information theory-based properties (*e.g.*, feature entropies, mutual information, noise-to-signal ratio), and properties quantifying task complexity (*e.g.*, Fisher's discriminant ratio, feature efficiency, hub score) [57]. A total of 213 metafeatures, computable by the `PyMFE` Python package [58], are considered.

Predictive metafeatures for each algorithm are identified by fitting a linear regression model of metafeatures to the algorithm's 31 Macro F_1 scores across all tasks. A minimal set of predictive metafeatures common to all algorithms is determined using sequential feature addition. Sequential feature addition starts by constructing linear models of individual metafeatures for all algorithms. The metafeature ψ_1 that results in the lowest mean absolute error (MAE) is added to the set of selected metafeatures, with MAE computed via leave-one-out cross-validation. The process is repeated with combinations of ψ_1 and additional metafeatures, adding the metafeature ψ_2 that results in the lowest MAE. This iterative process continues until MAE decreases by less than 1%. The final set of metafeatures $\{\psi_i\}$ is used to build maximally predictive linear models of algorithm performance across tasks. BKDE-based algorithms are excluded from this analysis due to the inability of metafeatures to predict their performance.

4 Results and Discussion

4.1 Active learning with neural networks and random forests generally outperforms other strategies.

All combinations of samplers and models (totaling 100 algorithms) were applied as space-filling and active-learning algorithms to all tasks in Figure 1C. This process was repeated with 30 different random seeds to assess performance variability due to stochastic factors such as sampler initialization, model random states, and hyperparameter tuning. Performance was assessed in terms of overall accuracy and in terms of data efficiency.

Figure 3A shows the 20 highest-performing algorithms, by accuracy, as measured by average relative Macro F_1 score across all tasks for ten rounds of active learning. NN- and RF-based active-learning algorithms are the most accurate classifiers regardless of sampler choice, representing 10 out of the top 11 algorithms. Most variants of XGB-based active-learning algorithms are also present in the top 20, along with a few GP- and SV-based active-learning algorithms. Space-filling algorithms are notably missing from the top performers, suggesting the value of iterative data acquisition. There is no clear indication that the choice of sampler substantially affects performance of these algorithms, with roughly equal representation of all samplers. The presence of all NN- and RF-based active-learning algorithms in the top 20 suggests that choice of model important than of sampler. While the results are statistically robust, we note that the top-20 most accurate algorithms differ in relative Macro F_1 scores by at most ca. 0.04; the practical implication of such a difference would require additional external evaluation.

To better characterize the data efficiency of algorithms, we consider $\langle \xi/\xi_{\text{max}} \rangle$, the performance relative to a naive algorithm that achieves equivalent accuracy. For example, $\langle \xi/\xi_{\text{max}} \rangle = 0.6$ means that the number of points required by a naive method to achieve the same accuracy as that algorithm is 60% of the number of points required by a naive method to achieve the same accuracy as the most accurate algorithm for that task. Figure 3B ranks the top-20 algorithms by $\langle \xi/\xi_{\text{max}} \rangle$ for all tasks after ten rounds of active learning. Figure 3B shows that NN-based active-learning algorithms are the clear top performers, regardless of sampler, when using this metric. RF-based active learning models follow closely behind, followed by a variety of Gaussian process-based active learning methods. Compared to Figure 3A, the metric in Figure 3B provides greater stratification in algorithm performance for high values of Macro F_1. As F_1 scores tend to 1.0, more and more points are required by a naive algorithm to improve its accuracy, which is reflected only by small increases in Macro F_1 score. When appropriately weighting the relative "effort" required for getting a high-resolution understanding of the task, NN-based active-learning algorithms emerge as a consistent top performer. However, the maximum value of $\langle \xi/\xi_{\text{max}} \rangle$ achieved by any algorithm is less than 0.8, indicating that even the top-performing algorithms are not necessarily optimal for many tasks.

The ordering in Figure 3 reflects an average across all tasks and specifically follows after ten rounds of active learning. Variants of Figure 3 for different subsets of tasks and fewer points selected are available in the SI (see Section S2). We find that performance varies depending on the dimensionality of the tasks. Figure S1 shows that when only low-dimensional tasks ($d \leq 8$) are considered, NN-based active-learning algorithms greatly outperform all alternatives.

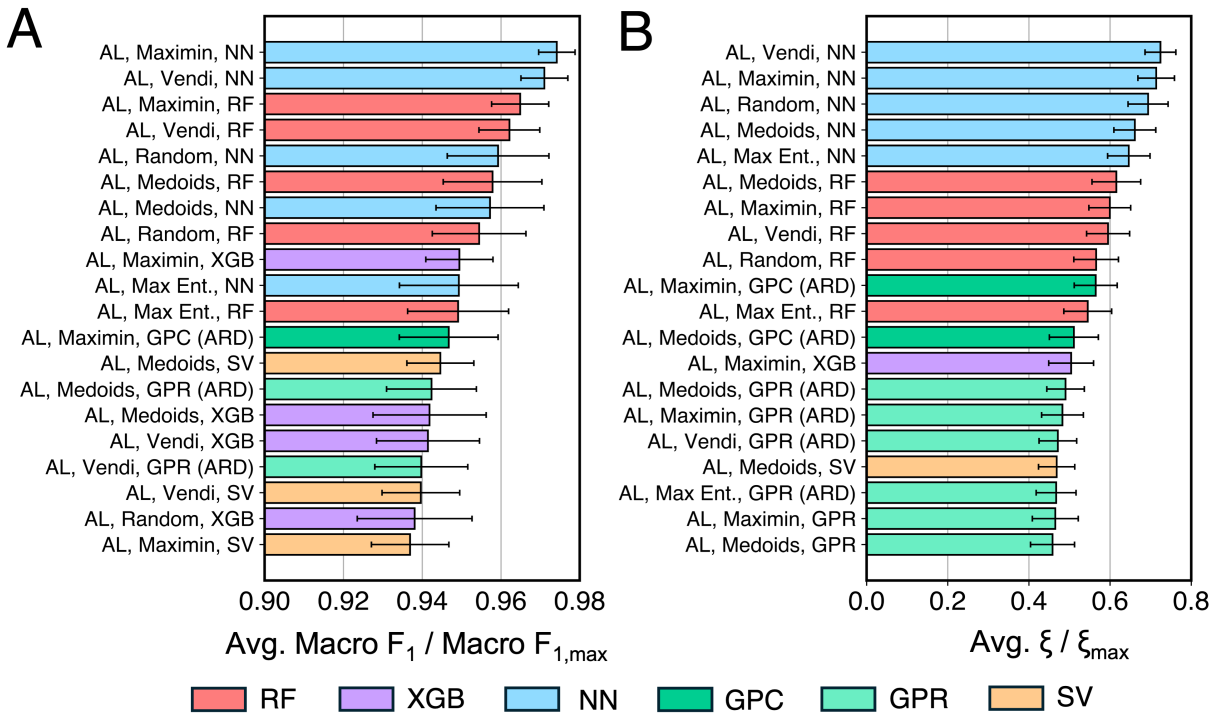

Figure 3: Performances of the top-20 algorithms on all tasks for ten rounds of active learning. Algorithm performance is measured by averaging the **(A)** relative Macro F_1 score and **(B)** relative ξ of each algorithm on all tasks, where "relative" denotes normalizing the metric by the performance of the top-performing algorithm on that task. Results are colored according to the model used by the specified algorithm. Error bars show the standard error.

Figure S2 shows that when higher dimensional tasks are considered ($d > 8$), tree-based algorithms perform better, and there is not a clear advantage to using either NN- or RF-based active-learning algorithms.

Figures S3-S5 show how the results in Figure 3 change when fewer points are selected. At only three rounds of active learning, space-filling algorithms with a variety of models are present in the top-20 algorithms (Figure S3). The top space-filling algorithm, which uses the medoids sampler and neural network model, remains in the top 20 until five rounds of active learning (Figure S4), closely followed by GP-based space-filling algorithms. NN-based active-learning algorithms are the top-performing algorithms for three rounds of active learning onwards, while RF-based active-learning algorithms do not emerge as the clear second best choice until seven rounds of active learning (Figure S5). Results are mostly consistent with Figure 3 for seven rounds of active learning. Therefore, the results of Figure 3 are consistent for many rounds of active learning, but when few batches have been selected, NN- and GP-based space-filling algorithms are competitive alternatives.

From these results, we suggest using NN- or RF-based active-learning algorithms for building accurate classification models on domains with a limited experimental budget. RFs seem preferred for higher-dimensional tasks. This guidance seemingly runs counter to conventional wisdom regarding the relative ineffectiveness of neural networks in low-data regimes and the common utilization of Gaussian processes for AL/BO. It may be interesting to consider whether prior studies (such as Refs. [23, 18, 20, 24, 25]) might be more data-efficient by opting for a different strategy.

4.2 Many algorithms fail to perform well across all tasks.

While the preceding analysis shows that selecting an optimal strategy *a priori* can be challenging, we find certain algorithms are consistently "suboptimal." We define performance as suboptimal if $\xi/\xi_{\max} < 0.9$ for every task. Figure 4 displays the fraction of suboptimal algorithms based on algorithm type and model choice. Of the 100 algorithms studied, 62 are suboptimal. Space-filling algorithms are more often suboptimal compared to active-learning algorithms. Among active-learning algorithms, model choice significantly affects performance. NN- and RF-based active-learning

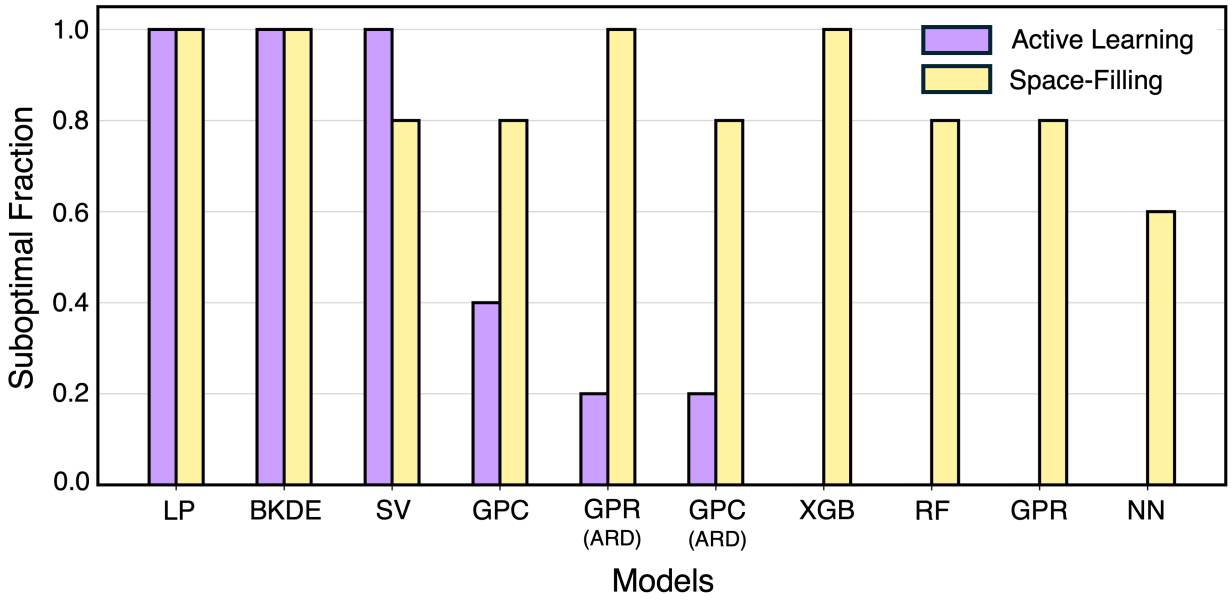

Figure 4: Summary of suboptimal algorithms. Algorithms are considered suboptimal if $\xi/\xi_{\text{max}} < 0.9$ on every task. Data is stratified by algorithm type and model choice.

algorithms are never suboptimal, while BKDE- and LP-based active-learning algorithms are always suboptimal. SV- and isotropic GPC-based active learning schemes are also commonly suboptimal.

The poor performance of BKDE-based algorithms may be attributed to several factors. First, unlike GPs and SVs, BKDE does not use training labels when estimating kernel densities, reducing predictive accuracy. Second, BKDE relies on a Bayesian autoencoder to estimate kernel densities, which can be inaccurate with limited training data. Third, BKDE's kernel density estimates rapidly decay to zero with distance, leading to high uncertainties across much of the task domain. This causes BKDE-based active-learning algorithms to fail in prioritizing points near classification boundaries, reducing accuracy. Consequently, BKDE-based active learning and space-filling algorithms perform similarly across tasks.

The poor performance of LP-based algorithms is likely due to two reasons. First, LP models assign classes to unlabeled points based on neighboring labeled points defined by Euclidean distance. Unlike anisotropic GPs, XGBs, RFs, and NNs, LP models do not have a mechanism to ignore irrelevant features. Second, LP models assign high uncertainties to points near an identified classification boundary but not to points far from those already chosen. As a result, LP models can miss classification boundaries not initially discovered by the sampler. This likely explains why space-filling LP algorithms outperform active-learning LP algorithms. Other methods address this issue by explicitly increasing the uncertainty of distant points (*e.g.*, GPs) or using model ensembles to encourage uncertainties in less sampled regions (*e.g.*, RFs, NNs, XGBs).

4.3 Space-filling occasionally outperforms active learning.

Figures 3 and 4 together suggest that active learning, or iterative data selection, is more data-efficient than one-shot space-filling. To quantify this, we compare the performance of every seed of every active-learning algorithm to that of the space-filling algorithm with the same seed, sampler, and model. For each number of points selected on each task, we compute how often the active learning variant outperforms the space-filling variant.

Figure 5 shows that active learning does not always outperform space-filling, especially with few rounds of active learning. In the first round, active learning outperforms space-filling in less than 50% of cases, suggesting little initial benefit. This fraction increases to about 65% by round 10. To avoid misleading results from poorly performing models, we also analyze the top-performing models from Figure 3B. In this case, active learning outperforms space-filling about 50% of the time in the first round, increasing to nearly 80% by round 10. While active learning generally outperforms space-filling, these results indicate that an arbitrary active learning scheme may not always surpass its space-filling variant on a given task.

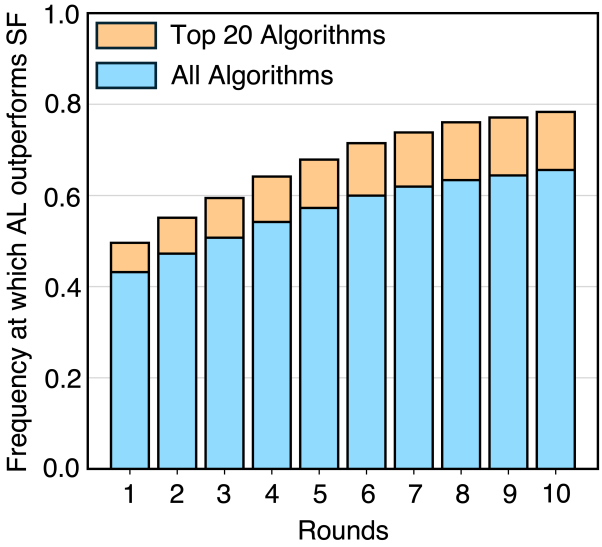

Figure 5: Controlled comparison of active learning (AL) and space-filling (SF). Data corresponds to the fraction of instances that AL outperforms SF when using the same sampler, model, and seed. Data is also stratified for different rounds of AL. For a given round of AL, the AL algorithm is compared to a SF algorithm that has selected the same number of points. The fractions considering all sampler, model, and seed choices are shown in blue. The fractions considering only the top 20 sampler, model, and seed choices in Figure 3B include the blue and orange bars. Statistics are aggregated over all tasks.

The results in Figure 5 depend on the tasks considered. Some tasks (*e.g.*, `princeton`, `tox21`, `electro`) deviate from aforementioned trends (Figures S6-8). In these cases, space-filling algorithms consistently outperform active-learning algorithms. We note that these are also among the most difficult classification tasks, as indicated by the mean performance of all algorithms. When all algorithms struggle, the performance gap between active learning and space filling is less meaningful. Additionally, datasets like `princeton` have complex classification boundaries that benefit from allocating more experimental budget to exploring the task domain rather than refining an already discovered boundary.

Based on these results, we recommend using active-learning algorithms for data-efficient classifiers but acknowledge that factors such as *i.* the number of active learning rounds and *ii.* the expected complexity of the classification task can influence the balance between sampler and active learning points. Determining the optimal trade-off is left for future work.

4.4 Sampler choice has disparate impact on active learning versus space-filling.

The role of initial data selection is an often overlooked aspect of active-learning algorithms and their outcomes. To assess this impact, we compare the performance of each active-learning algorithm with non-random samplers to the same algorithm with a random sampler across all tasks after 10 rounds of active learning. Here, the performance metric is ξ, as defined in **Methods**.

Figure 6 shows that the choice of sampler has little impact on performance for active-learning algorithms. Since the sampler selects fewer than 10% of the points, its impact is minimal. Maximin and medoids samplers offer a slight improvement over random sampling, though the difference is marginal. In the analysis of space-filling algorithms with both non-random and random samplers, the medoids sampler performs significantly better than others, while maximin and Vendi samplers outperform random sampling. Conversely, the max entropy sampler performs worse than random selection.

The results for active-learning algorithms vary with the number of rounds. Variants of Figure 6 for different rounds are shown in the SI (see Section S4). Early rounds show the medoids sampler improving performance by about 1.40 times over the random sampler (Figure S9). As rounds increase, the results converge to those in Figure 6. The medoids sampler performs best for both active learning and space-filling algorithms, but its impact diminishes with more rounds of active learning.

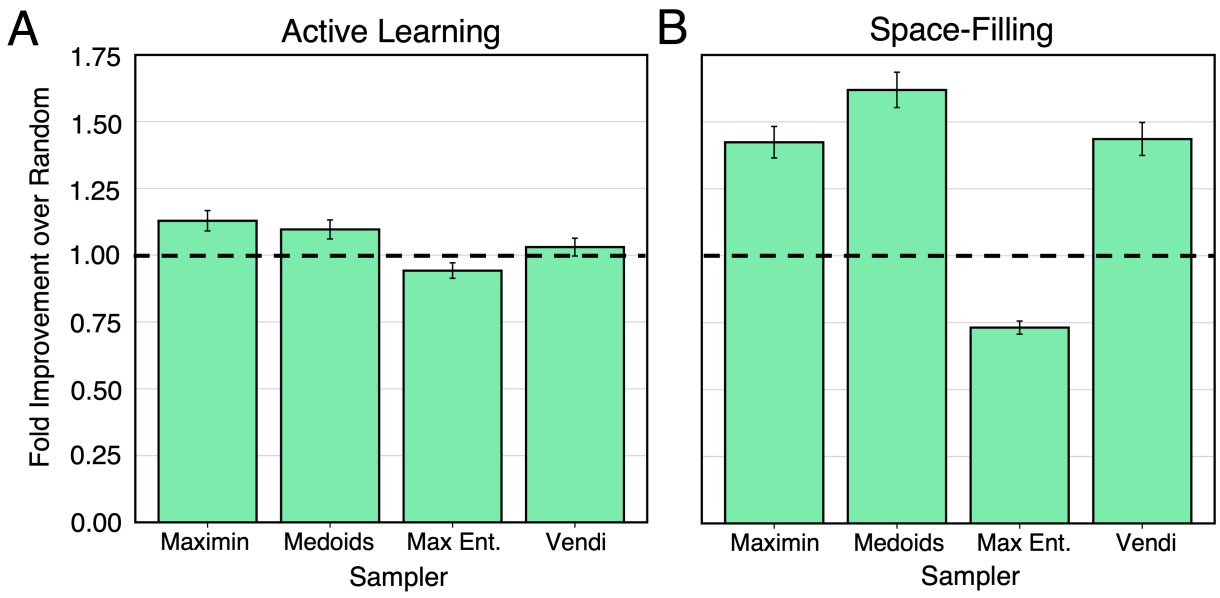

Figure 6: Summary of performance improvements based on different samplers. Average fold improvement in data efficiency ξ based on the specified sampler relative to a random sampler for **(A)** active learning and **(B)** space-filling algorithms. The dotted line is a guide to the eye for $y = 1.00$ (the performance of algorithms with random samplers).

4.5 Model ensembles provide robust performance.

Given that the maximum $\langle \xi/\xi_{\text{max}} \rangle$ in Figure 3B is only about 0.77, we hypothesized that more data-efficient classification algorithms could be developed using ensembling. Based on observed performance, we consider ensemble-based algorithms featuring NNs, RFs, and anisotropic GPCs. Several ensembling strategies were considered, including treating model choice as a hyperparameter, "stacking" models to combine predictions and uncertainties, and "arbitrating" by using the model with the lowest uncertainty for each prediction, and others[59] (see SI, Section S5). From this survey, treating model choice as a hyperparameter was found to be the best-performing scheme, and all "Ensemble" results in the main text refer to this method.

Figure 7A shows the relative performance based on $\langle \xi/\xi_{\text{max}} \rangle$ of NN-, RF-, and ensemble-based active-learning algorithms across all tasks. While ensembles rank among the top-performing algorithms, they do not consistently outperform NN-based active-learning algorithms. However, the results in Figure 7 are task-dependent, suggesting that ensemble-based active learning may be beneficial for certain types of tasks.

To determine if ensemble-based active-learning algorithms outperform NN- and RF-based algorithms on specific tasks, we analyze two task sets. Figure 7B shows tasks where NN-based algorithms are the top performers ($n = 9$). Figure 7C shows tasks where RF-based algorithms excel ($n = 10$). Ensemble schemes generally outperform individual models on tasks where those models are not optimal. This effect is strongest for tasks where NN-based algorithms excel and less pronounced for RF-based tasks. Thus, using ensemble-based active learning may mitigate the risk of selecting a suboptimal model for any given task.

4.6 Metafeatures are predictive of task difficulty.

To understand the factors behind algorithm performance variability across tasks, we identify metafeatures that predict learning algorithm accuracy. This approach allows us to quantify what makes one classification task more challenging than another.

Figure 8 shows that a limited set of task metafeatures identified by sequential feature addition can reasonably predict task complexity. Figure 8A shows results of using just four metafeatures (noise-to-signal ratio [57], maximum weighted distance between two points in the task domain [60], maximum mutual information between features and labels, and the performance of the linear discriminant classifier) to predict the accuracy of all algorithms across all tasks. To reduce the influence of poorly performing algorithms, the same analysis is performed using just the top-20 algorithms. This

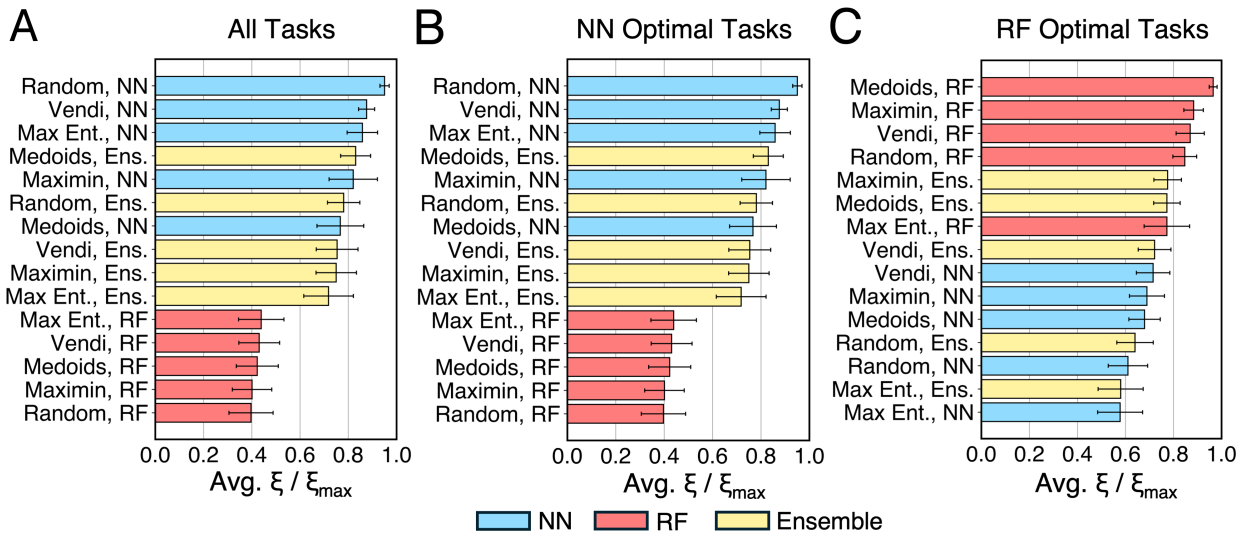

Figure 7: Performance decomposition of NN-, RF-, and ensemble-based active-learning algorithms. Performances are measured on **(A)** all tasks, **(B)** those tasks where NN-based algorithms are optimal, and **(C)** those tasks where RF-based algorithms are optimal. Performance is measured by $\langle \xi / \xi_{\max} \rangle$ across the specified set of tasks, and error bars show the standard error.

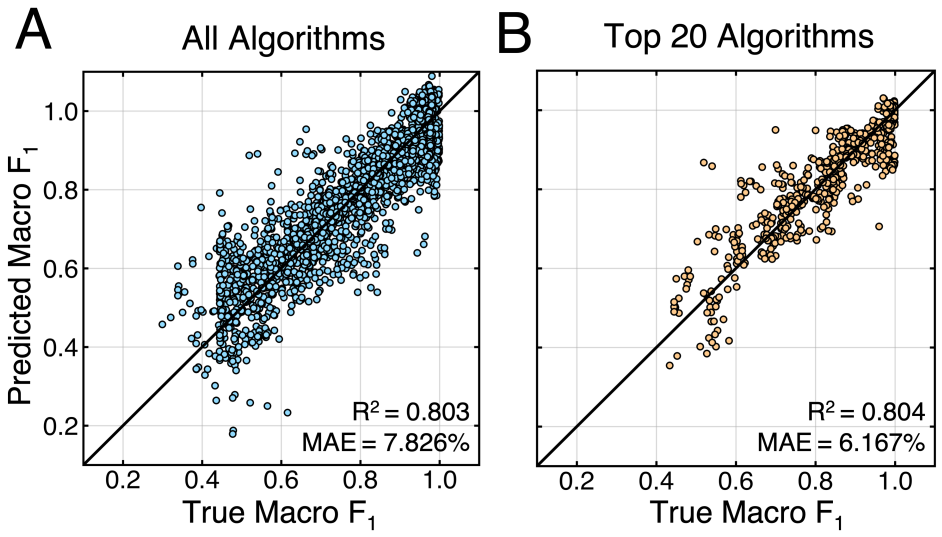

Figure 8: Parity plots of linear models trained on a set of predictive metafeatures. The parity line is shown in black. **(A)** shows the predictions of linear models constructed for all algorithms using *i.* noise-to-signal ratio, *ii.* maximum weighted distance between points, and *iii.* maximum mutual information between features and labels, and *iv.* the relative mean performance of the linear discriminant classifier. **(B)** shows the predictions of linear models constructed for the top-20 algorithms using *i.* noise-to-signal ratio, *ii.* average mutual information between features and labels, and *iii.* maximum performance of the naive Bayes classifier.

yields Figure 8B, which uses noise-to-signal ratio, the average mutual information between features and labels, and the performance of the naive Bayes classifier. In both cases, simple linear models based on these few features capture the data well.

The particular metafeatures selected resonate with intuition. Noise-to-signal ratio is the most predictive metafeature of algorithm performance across all tasks. Linear models using only this ratio achieve an MAE of 11.260% and $R^2 = 0.619$ for all algorithms, and an MAE of 8.370% and $R^2 = 0.696$ for top-performing algorithms. Tasks with low noise-to-signal ratios require fewer measurements because each measurement provides valuable information about labels. Related to the noise-to-signal ratio, mutual information and the performance of the naive Bayes classifier indicate how useful individual features are for predicting labels. When features are individually predictive of labels, less data is required for accurate predictions than for tasks where features are uninformative. The maximum weighted distance between point pairs [60] likely identifies outliers in the task domain, which require more measurements to account for their influence. The performance of the linear discriminant classifier indicates the linear separability of a task. Linearly separable tasks have simple classification boundaries, requiring less data for accurate prediction. In simple and expected terms, less data is needed to train accurate models for tasks with informative features, few outliers, and linearly separable classes.

5 Conclusion

We characterized the relative efficiency and performance of strategies for building effective machine learning classifiers with relevance across chemical and materials science. In total, 100 space-filling and active-learning algorithms were evaluated across 31 classification tasks. Our findings indicated that NN- and RF-based active-learning algorithms were the most data-efficient, while BKDE and LP algorithms performed poorly in comparison. Space-filling methods were competitive with active learning, particularly when few rounds of active learning were used. We also demonstrated that using the k-medoids algorithm for point selection improved accuracy over other sampling strategies in both active learning and space-filling. Ensemble-based algorithms were found to perform generally well, irrespective of task. Additionally, task metafeatures were predictive of algorithm performance, with a few key metafeatures, particularly the noise-to-signal ratio, effectively quantifying classification task complexity. These results have implications for data-driven materials design in constrained domains.

This study opens several avenues for future research. Key areas for further investigation include exploring algorithm design choices not covered here, such as feature and label transformations, batch-selection schemes, and batch sizes. Additionally, applying the current findings to materials design campaigns that involve simultaneous optimization and classification, as discussed by Hickman *et al.* [24], could be valuable. Beyond algorithm design, incorporating domain knowledge could enhance data efficiency. Utilizing pre-trained models, incorporating priors from foundation models, and applying physical constraints on model predictions may offer significant improvements in data efficiency compared to changes in sampler or model. Specifically, constructing pre-trained material representations optimized for metafeatures predictive of algorithm performance, like the noise-to-signal ratio, could boost data efficiency across materials domains. This approach could be beneficial for both classification and regression tasks [61]. Moreover, employing identified metafeatures in the development of featurization strategies that optimize these metrics could lead to more data-efficient classification. Latent representations from autoencoders, multitask models, or large language models trained on extensive datasets could be optimized to minimize the noise-to-signal ratio or maximize linear separability, thereby enhancing performance on related tasks. Finally, establishing a unified set of classification tasks for testing would strengthen the generalizability of the findings here and for future studies.

6 Acknowledgements

Q.M.G. acknowledges support from the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE-2039656 and from the National Science Foundation under Grant No. 2118861. M.A.W. also acknowledges support from the National Science Foundation under Grant No. 2118861. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. Simulations and analyses were performed using resources from Princeton Research Computing at Princeton University, which is a consortium led by the Princeton Institute for Computational Science and Engineering (PICSciE) and Office of Information Technology's Research Computing.

7 Data availability

All code required to reproduce measurements of classification algorithm performance on all tasks is available in the following GitHub repository: `https://github.com/webbtheosim/classification-suite.git`. The produced data, along with scripts for reproducing the analysis presented in this work are available in the following GitHub repository: `https://github.com/webbtheosim/classification-analysis.git`.

References

[1] Daniel Reker and Gisbert Schneider. Active-learning strategies in computer-assisted drug discovery. *Drug Discovery Today*, 20(4):458–465, apr 2015.

[2] Chiho Kim, Anand Chandrasekaran, Anurag Jha, and Rampi Ramprasad. Active-learning and materials design: the example of high glass transition temperature polymers. *MRS Communications*, 9(3):860–866, jun 2019.

[3] Roshan A. Patel and Michael A. Webb. Data-driven design of polymer-based biomaterials: High-throughput simulation, experimentation, and machine learning. *ACS Applied Bio Materials*, jan 2023.

[4] Kirill Shmilovich, Rachael A. Mansbach, Hythem Sidky, Olivia E. Dunne, Sayak Subhra Panda, John D. Tovar, and Andrew L. Ferguson. Discovery of Self-Assembling π-Conjugated Peptides by Active Learning-Directed Coarse-Grained Molecular Simulation. *The Journal of Physical Chemistry B*, 124(19):3873–3891, May 2020. Publisher: American Chemical Society.

[5] Zhenpeng Yao, Benjamin Sanchez-Lengeling, N. Scott Bobbitt, Benjamin J. Bucior, Sai Govind Hari Kumar, Sean P. Collins, Thomas Burns, Tom K. Woo, Omar Farha, Randall Q. Snurr, and Alan Aspuru-Guzik. Inverse Design of Nanoporous Crystalline Reticular Materials with Deep Generative Models, August 2020.

[6] Rafael Gómez-Bombarelli, Jorge Aguilera-Iparraguirre, Timothy D. Hirzel, David Duvenaud, Dougal Maclaurin, Martin A. Blood-Forsythe, Hyun Sik Chae, Markus Einzinger, Dong-Gwang Ha, Tony Wu, Georgios Markopoulos, Soonok Jeon, Hosuk Kang, Hiroshi Miyazaki, Masaki Numata, Sunghan Kim, Wenliang Huang, Seong Ik Hong, Marc Baldo, Ryan P. Adams, and Alán Aspuru-Guzik. Design of efficient molecular organic light-emitting diodes by a high-throughput virtual screening and experimental approach. *Nature Materials*, 15(10):1120–1127, October 2016. Number: 10 Publisher: Nature Publishing Group.

[7] Yaxin An, Michael A. Webb, and William M. Jacobs. Active learning of the thermodynamics-dynamics trade-off in protein condensates. *Science Advances*, 10(1), January 2024.

[8] Marcus Reis, Filipp Gusev, Nicholas G. Taylor, Sang Hun Chung, Matthew D. Verber, Yueh Z. Lee, Olexandr Isayev, and Frank A. Leibfarth. Machine-learning-guided discovery of 19f MRI agents enabled by automated copolymer synthesis. *Journal of the American Chemical Society*, 143(42):17677–17689, oct 2021.

[9] Matthew J. Tamasi, Roshan A. Patel, Carlos H. Borca, Shashank Kosuri, Heloise Mugnier, Rahul Upadhya, N. Sanjeeva Murthy, Michael A. Webb, and Adam J. Gormley. Machine Learning on a Robotic Platform for the Design of Polymer–Protein Hybrids. *Advanced Materials*, 34(30):2201809, 2022. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/adma.202201809.

[10] Elizabeth A. Pogue, Alexander New, Kyle McElroy, Nam Q. Le, Michael J. Pekala, Ian McCue, Eddie Gienger, Janna Domenico, Elizabeth Hedrick, Tyrel M. McQueen, Brandon Wilfong, Christine D. Piatko, Christopher R. Ratto, Andrew Lennon, Christine Chung, Timothy Montalbano, Gregory Bassen, and Christopher D. Stiles. Closed-loop superconducting materials discovery. *npj Computational Materials*, 9(1):1–8, October 2023. Publisher: Nature Publishing Group.

[11] Guangqi Wu, Haisen Zhou, Jun Zhang, Zi-You Tian, Xingyi Liu, Shuo Wang, Connor W. Coley, and Hua Lu. A high-throughput platform for efficient exploration of functional polypeptide chemical space. *Nature Synthesis*, 2(6):515–526, May 2023.

[12] Ismael J. Gomez, Jie Wu, John Roper, Haskell Beckham, and J. Carson Meredith. High Throughput Screening of Mechanical Properties and Scratch Resistance of Tricomponent Polyurethane Coatings. *ACS Applied Polymer Materials*, 1(11):3064–3073, November 2019. Publisher: American Chemical Society.

[13] Ramya Kumar, Ngoc Le, Zhe Tan, Mary E. Brown, Shan Jiang, and Theresa M. Reineke. Efficient polymer-mediated delivery of gene-editing ribonucleoprotein payloads through combinatorial design, parallelized experimentation, and machine learning. *ACS Nano*, 14:17626–17639, nov 2020.

[14] Shashank Kosuri, Carlos H. Borca, Heloise Mugnier, Matthew Tamasi, Roshan A. Patel, Isabel Perez, Suneel Kumar, Zachary Finkel, Rene Schloss, Li Cai, Martin L. Yarmush, Michael A. Webb, and Adam J. Gormley. Machine-assisted discovery of chondroitinase ABC complexes toward sustained neural regeneration. *Advanced Healthcare Materials*, 11(10):2102101, feb 2022.

[15] Kevin Maik Jablonka, Giriprasad Melpatti Jothiappan, Shefang Wang, Berend Smit, and Brian Yoo. Bias free multiobjective active learning for materials design and discovery. *Nature Communications*, 12(1):2312, apr 2021.

[16] Sabine Körbel, Miguel A. L. Marques, and Silvana Botti. Stable hybrid organic–inorganic halide perovskites for photovoltaics from ab initio high-throughput calculations. *Journal of Materials Chemistry A*, 6(15):6463–6475, April 2018. Publisher: The Royal Society of Chemistry.

[17] András Hatos, Borbála Hajdu-Soltész, Alexander M Monzon, Nicolas Palopoli, Lucía Álvarez, Burcu Aykac-Fas, Claudio Bassot, Guillermo I Benítez, Martina Bevilacqua, Anastasia Chasapi, Lucia Chemes, Norman E Davey, Radoslav Davidović, A Keith Dunker, Arne Elofsson, Julien Gobeill, Nicolás S González Foutel, Govindarajan Sudha, Mainak Guharoy, Tamas Horvath, Valentin Iglesias, Andrey V Kajava, Orsolya P Kovacs, John Lamb, Matteo Lambrughi, Tamas Lazar, Jeremy Y Leclercq, Emanuela Leonardi, Sandra Macedo-Ribeiro, Mauricio Macossay-Castillo, Emiliano Maiani, José A Manso, Cristina Marino-Buslje, Elizabeth Martínez-Pérez, Bálint Mészáros, Ivan Mičetić, Giovanni Minervini, Nikoletta Murvai, Marco Necci, Christos A Ouzounis, Mátyás Pajkos, Lisanna Paladin, Rita Pancsa, Elena Papaleo, Gustavo Parisi, Emilie Pasche, Pedro J Barbosa Pereira, Vasilis J Promponas, Jordi Pujols, Federica Quaglia, Patrick Ruch, Marco Salvatore, Eva Schad, Beata Szabo, Tamás Szaniszló, Stella Tamana, Agnes Tantos, Nevena Veljkovic, Salvador Ventura, Wim Vranken, Zsuzsanna Dosztányi, Peter Tompa, Silvio C E Tosatto, and Damiano Piovesan. DisProt: intrinsic protein disorder annotation in 2020. *Nucleic Acids Research*, nov 2019.

[18] Kei Terayama, Koji Tsuda, and Ryo Tamura. Efficient recommendation tool of materials by an executable file based on machine learning. *Japanese Journal of Applied Physics*, 58(9):098001, August 2019. Publisher: IOP Publishing.

[19] Kei Terayama, Ryo Tamura, Yoshitaro Nose, Hidenori Hiramatsu, Hideo Hosono, Yasushi Okuno, and Koji Tsuda. Efficient construction method for phase diagrams using uncertainty sampling. *Physical Review Materials*, 3(3):033802, March 2019. Publisher: American Physical Society.

[20] Kei Terayama, Kwangsik Han, Ryoji Katsube, Ikuo Ohnuma, Taichi Abe, Yoshitaro Nose, and Ryo Tamura. Acceleration of phase diagram construction by machine learning incorporating Gibbs' phase rule. *Scripta Materialia*, 208:114335, February 2022.

[21] Xiaojin Zhu and Zoubin Ghahramani. Learning from Labeled and Unlabeled Data with Label Propagation.

[22] Xabier Telleria-Allika, Jose M. Mercero, Xabier Lopez, and Jon M. Matxain. Building machine learning assisted phase diagrams: Three chemically relevant examples. *AIP Advances*, 12(7):075206, July 2022.

[23] Chengyu Dai and Sharon C. Glotzer. Efficient Phase Diagram Sampling by Active Learning. *The Journal of Physical Chemistry B*, 124(7):1275–1284, February 2020.

[24] Riley Hickman, Matteo Aldeghi, and Alán Aspuru-Guzik. Anubis: Bayesian optimization with unknown feasibility constraints for scientific experimentation, October 2023.

[25] Maya Bhat and John R. Kitchin. Sequential Sampling Methods for Finding Classification Boundaries in Engineering Applications. *Industrial & Engineering Chemistry Research*, 62(37):15326–15339, September 2023. Publisher: American Chemical Society.

[26] Turab Lookman, Prasanna V. Balachandran, Dezhen Xue, and Ruihao Yuan. Active learning in materials science with emphasis on adaptive sampling using uncertainties for targeted design. *npj Computational Materials*, 5(1):21, February 2019.

[27] Jingjin He, Xiaopo Su, Changxin Wang, Junjie Li, Yuxuan Hou, Zhonghua Li, Chuanbao Liu, Dezhen Xue, Jiangli Cao, Yanjing Su, Lijie Qiao, Turab Lookman, and Yang Bai. Machine learning assisted predictions of multi-component phase diagrams and fine boundary information. *Acta Materialia*, 240:118341, November 2022.

[28] Ryoji Katsube, Kei Terayama, Ryo Tamura, and Yoshitaro Nose. Experimental Establishment of Phase Diagrams Guided by Uncertainty Sampling: An Application to the Deposition of Zn–Sn–P Films by Molecular Beam Epitaxy. *ACS Materials Letters*, 2(6):571–575, June 2020.

[29] Siva Dasetty, Igor Coropceanu, Joshua Portner, Jiyuan Li, Juan J. de Pablo, Dmitri Talapin, and Andrew L. Ferguson. Active learning of polarizable nanoparticle phase diagrams for the guided design of triggerable self-assembling superlattices. *Molecular Systems Design & Engineering*, 7(4):350–363, April 2022. Publisher: The Royal Society of Chemistry.

[30] Yuan Tian, Ruihao Yuan, Dezhen Xue, Yumei Zhou, Yunfan Wang, Xiangdong Ding, Jun Sun, and Turab Lookman. Determining Multi-Component Phase Diagrams with Desired Characteristics Using Active Learning. *Advanced Science*, 8(1):2003165, January 2021.

[31] Hirotomo Moriwaki, Yu-Shi Tian, Norihito Kawashita, and Tatsuya Takagi. Mordred: a molecular descriptor calculator. *Journal of Cheminformatics*, 10(1):4, February 2018.

[32] Nicholas Angello, David Friday, Changhyun Hwang, Seungjoo Yi, Austin Cheng, Tiara Torres-Flores, Edward Jira, Wesley Wang, Alàn Aspuru-Guzik, Martin Burke, Charles Schroeder, Ying Diao, and Nicholas Jackson. Closed-Loop Transfer Enables AI to Yield Chemical Knowledge, September 2023.

[33] David Rogers and Mathew Hahn. Extended-Connectivity Fingerprints. *Journal of Chemical Information and Modeling*, 50(5):742–754, May 2010. Publisher: American Chemical Society.

[34] Steven Kearnes, Kevin McCloskey, Marc Berndl, Vijay Pande, and Patrick Riley. Molecular graph convolutions: moving beyond fingerprints. *Journal of Computer-Aided Molecular Design*, 30(8):595–608, August 2016.

[35] Felix Musil, Andrea Grisafi, Albert P. Bartók, Christoph Ortner, Gábor Csányi, and Michele Ceriotti. Physics-Inspired Structural Representations for Molecules and Materials. *Chemical Reviews*, 121(16):9759–9815, August 2021. Publisher: American Chemical Society.

[36] R.A. Fisher. *The design of experiments*. Oliver and Boyd Ltd., Edinburg: Tweeddale Court, second edition, 1937.

[37] M. D. McKay, R. J. Beckman, and W. J. Conover. A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code. *Technometrics*, 21(2):239–245, 1979. Publisher: [Taylor & Francis, Ltd., American Statistical Association, American Society for Quality].

[38] I. M Sobol'. On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, 7(4):86–112, January 1967.

[39] Antonio R. C. Paiva. Information-theoretic dataset selection for fast kernel learning. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 2088–2095, Anchorage, AK, USA, May 2017. IEEE.

[40] B. W. Silverman. *Density estimation for statistics and data analysis*. Number 26 in Monographs on statistics and applied probability. Chapman & Hall/CRC, Boca Raton, Fla., 1. crc reprint edition, 1998.

[41] Dan Friedman and Adji Bousso Dieng. The Vendi Score: A Diversity Evaluation Metric for Machine Learning, July 2023. arXiv:2210.02410 [cond-mat, stat].

[42] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press, Cambridge, Mass, 2006. OCLC: ocm61285753.

[43] Florian Häse, Matteo Aldeghi, Riley J. Hickman, Loïc M. Roch, and Alán Aspuru-Guzik. Gryffin: An algorithm for Bayesian optimization of categorical variables informed by expert knowledge. *Applied Physics Reviews*, 8(3):031406, July 2021.

[44] Florian Häse, Loïc M. Roch, Christoph Kreisbeck, and Alán Aspuru-Guzik. Phoenics: A Bayesian Optimizer for Chemistry. *ACS Central Science*, 4(9):1134–1145, September 2018. Publisher: American Chemical Society.

[45] David Ginsbourger, Rodolphe Le Riche, and Laurent Carraro. A Multi-points Criterion for Deterministic Parallel Global Optimization based on Gaussian Processes. Technical report, March 2008.

[46] Zhenqin Wu, Bharath Ramsundar, Evan N. Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S. Pappu, Karl Leswing, and Vijay Pande. MoleculeNet: a benchmark for molecular machine learning. *Chemical Science*, 9(2):513–530, 2018. Publisher: Royal Society of Chemistry.

[47] Aldair E. Gongora, Bowen Xu, Wyatt Perry, Chika Okoye, Patrick Riley, Kristofer G. Reyes, Elise F. Morgan, and Keith A. Brown. A Bayesian experimental autonomous researcher for mechanical design. *Science Advances*, 6(15):eaaz1708, April 2020. Publisher: American Association for the Advancement of Science.

[48] Akash Arora, Tzyy-Shyang Lin, Nathan J. Rebello, Sarah H. M. Av-Ron, Hidenobu Mochigase, and Bradley D. Olsen. Random Forest Predictor for Diblock Copolymer Phase Behavior. *ACS Macro Letters*, 10(11):1339–1345, November 2021. Publisher: American Chemical Society.

[49] Masanori Kodera and Kazuhiro Sayama. An automatic robot system for machine learning–assisted high-throughput screening of composite electrocatalysts. *Digital Discovery*, 2(6):1683–1687, December 2023. Publisher: RSC.

[50] Loïc M. Roch, Florian Häse, Christoph Kreisbeck, Teresa Tamayo-Mendoza, Lars P. E. Yunker, Jason E. Hein, and Alan Aspuru-Guzik. ChemOS: An Orchestration Software to Democratize Autonomous Discovery, March 2018.

[51] Florian Häse, Matteo Aldeghi, Riley J. Hickman, Loïc M. Roch, Melodie Christensen, Elena Liles, Jason E. Hein, and Alán Aspuru-Guzik. Olympus: a benchmarking framework for noisy optimization and experiment planning. *Machine Learning: Science and Technology*, 2(3):035021, July 2021. Publisher: IOP Publishing.

[52] Jeffrey G. Ethier, Rohan K. Casukhela, Joshua J. Latimer, Matthew D. Jacobsen, Boris Rasin, Maneesh K. Gupta, Luke A. Baldwin, and Richard A. Vaia. Predicting Phase Behavior of Linear Polymers in Solution Using Machine Learning. *Macromolecules*, 55(7):2691–2702, April 2022. Publisher: American Chemical Society.

[53] Raghunathan Ramakrishnan, Pavlo O. Dral, Matthias Rupp, and O. Anatole Von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data*, 1(1):140022, August 2014.

[54] Qi Yuan, Mariagiulia Longo, Aaron W. Thornton, Neil B. McKeown, Bibiana Comesaña-Gándara, Johannes C. Jansen, and Kim E. Jelfs. Imputation of missing gas permeability data for polymer membranes using machine learning. *Journal of Membrane Science*, 627:119207, June 2021.

[55] Shengli Jiang, Adji Bousso Dieng, and Michael Webb. Property-Guided Generation of Complex Polymer Topologies Using Variational Autoencoders, February 2024.

[56] Ryan Turner, David Eriksson, Michael McCourt, Juha Kiili, Eero Laaksonen, Zhen Xu, and Isabelle Guyon. Bayesian Optimization is Superior to Random Search for Machine Learning Hyperparameter Tuning: Analysis of the Black-Box Optimization Challenge 2020. 2020.

[57] Donald Michie, editor. *Machine learning, neural and statistical classification*. Ellis Horwood series in artificial intelligence. Ellis Horwood, New York, 1. publ., 2. [print.] edition, 1995.

[58] Edesio Alcobaça, Felipe Siqueira, Adriano Rivolli, Luís P. F. Garcia, Jefferson T. Oliva, and André C. P. L. F. de Carvalho. MFE: Towards reproducible meta-feature extraction. *Journal of Machine Learning Research*, 21(111):1–5, 2020.

[59] Pavel Brazdil, Jan N. Van Rijn, Carlos Soares, and Joaquin Vanschoren. *Metalearning: Applications to Automated Machine Learning and Data Mining*. Cognitive Technologies. Springer International Publishing, Cham, 2022.

[60] Ricardo Vilalta. Understanding accuracy performance through concept characterization and algorithm analysis. (Proceedings of the ICML-99 workshop on recent advances in meta-learning and future work):3–9, 1999.

[61] Matteo Aldeghi, David E. Graff, Nathan Frey, Joseph A. Morrone, Edward O. Pyzer-Knapp, Kirk E. Jordan, and Connor W. Coley. Roughness of Molecular Property Landscapes and Its Impact on Modellability. *Journal of Chemical Information and Modeling*, 62(19):4660–4671, October 2022. Publisher: American Chemical Society.