9 August 2026

Chemical space separation: Scaffold and UMAP splits are closer to random than we think

Bryn Marie Reimer[1], Vy Do[1], Anna G. Green[1]

1. University of Massachusetts Amherst

Abstract

The successful application of molecular property prediction models depends on their ability to generalize beyond their training data, and various splitting strategies attempt to partition out-of-distribution test data to measure model generalizability. In this study, we assessed the ability of current state-of-the art splitting methods including Bemis-Murcko scaffold, Uniform Manifold Approximation and Projection (UMAP), and a recently-introduced splitting framework called SPECTRA, to generate test sets that are truly out-of-distribution. We compare to random data splitting as a baseline, as this is assumed to create easier, more in-distribution data splits. We characterize the difficulty of each splitting strategy using cross-split (CSO), which quantifies the similarity between training and test sets, and examine its relationship with model performance across eight MoleculeNet datasets using six classical machine learning models and Chemprop, a modern deep learning model. Our results indicate that in 3 out of 8 datasets, scaffold and UMAP splits do not consistently produce more challenging tasks than random splits as measured by CSO, while SPECTRA generates a series of train-test splits that span a broad range of dissimilarity, providing a more comprehensive view of model generalizability than the single-split strategies.

Keywords

machine learning, molecular property prediction, data splitting, generalizability

Chemical space separation: Scaffold and UMAP splits are closer to random than we think

Bryn Marie Reimer*[1], Vy Do*[1], and Anna G. Green†[1]

[1]Manning College of Information and Computer Sciences, University of Massachusetts Amherst, Amherst, MA, USA
*These authors contributed equally.

†Email: annagreen@umass.edu

Abstract

The successful application of molecular property prediction models depends on their ability to generalize beyond their training data, and various splitting strategies attempt to partition out-of-distribution test data to measure model generalizability. In this study, we assessed the ability of current state-of-the-art splitting methods including Bemis-Murcko scaffold, Uniform Manifold Approximation and Projection (UMAP), and a recently-introduced splitting framework called SPECTRA, to generate test sets that are truly out-of-distribution. We compare to random data splitting as a baseline, as this is assumed to create easier, more in-distribution data splits. We characterize the difficulty of each splitting strategy using cross-split (CSO), which quantifies the similarity between training and test sets, and examine its relationship with model performance across eight MoleculeNet datasets using six classical machine learning models and Chemprop, a modern deep learning model. Our results indicate that in 3 out of 8 datasets, scaffold and UMAP splits do not consistently produce more challenging tasks than random splits as measured by CSO, while SPECTRA generates a series of train-test splits that span a broad range of dissimilarity, providing a more comprehensive view of model generalizability than the single-split strategies.

Keywords: machine learning, molecular property prediction, data splitting, generalizability

Introduction

Chemical space is extremely large; estimates of the total size of the "drug-like" space of chemical compounds that follow Lipinski's Rule of Five[1] range from 10^{23} to 10^{60} molecules, depending on whether synthetic feasibility is also considered.[2–4] Current datasets with information about the properties and activities of molecules, then, necessarily represent a very small fraction of what is potentially knowable about chemical space. Nevertheless, both traditional quantitative structure-property relationship (QSPR)[5] models and more modern deep learning (DL) models such as Chemprop[6–8] have shown promise for accelerating the drug discovery and design process by increasing throughput and accuracy for validated computational strategies such as virtual screening,[9] and interest has been growing in the pharmaceutical industry and beyond in the further development of these tools.[10] The question of *generalizability* remains: given that molecular datasets are small and biased relative to chemical space,[11] can these models perform well on prediction tasks involving out-of-distribution molecules? And how can scientists assess the generalizability of models which are being developed?

A rigorous and implementable definition of generalizability is elusive. The community defines generalizability as the ability of a model to maintain its performance when applied to out-of-distribution molecules, but the idea of "out-of-distribution" molecules itself does not have a precise definition in the literature. The community instead attempts to generate approximately "out-of-distribution" data with splitting strategies intended to partition harder test sets — *i.e.*, test sets that are more dissimilar to the training data. If

a model maintains its original performance on these "hard" test sets, the model is assumed to have good generalizability. However, when a model performs well on a particular data split, the good performance may be due to the generalizability of the model, or to a test set that is actually more similar to the training data than intended. The actual similarity between the train and test datasets is rarely explicitly measured.

The scientific community has invested substantial effort into designing split strategies for molecular data, as the choice of splitting strategy can impact both the performance metric as well as the conclusions drawn about the model's real-world utility. In this introduction, we will discuss splitting strategies to split data into a train set and a test set, but we note that many models also require a validation dataset used, for example, for hyperparameter tuning.

Random splitting is perhaps the most commonly used strategy in any machine learning setting because it is straightforward to implement. In a random split, molecules are assigned to train and test sets uniformly at random, sometimes stratified by label distribution for classification tasks. Random splits are presumed to be an "easier" type of split as there is no mechanism in place to penalize similar (or even identical) data points between the train and test datasets. Indeed, random splits may result in overly optimistic estimates of model performance when duplicates or very close analogs are included in the dataset.[12]

To mitigate this issue and attempt to assess performance on more dissimilar molecules, many benchmarking assessments have begun to use scaffold splitting strategies , which force all molecules with the same scaffold to be in either the train or test partition. This procedure is well suited to scenarios where one wants to predict properties for a new series rather than for additional analogs of an existing series. While in theory scaffolds may be custom-defined per dataset, studies typically rely on automated scaffold detection *via* algorithms such as Bemis-Murcko molecular framework detection as implemented in RDKit.[13,14] Automated scaffold detection is not always robust: for example, Bemis–Murcko molecular frameworks can fragment trivially modified chemical moieties into different scaffold groups, resulting in very similar scaffolds appearing in both the train and test sets.[15]

More recently, splitting strategies based on clustering, with or without dimensionality reduction,[16] have been explored as another way to control the similarity between the train and test sets.[15] Common implementations include Uniform Manifold Approximation and Projection (UMAP) splits and Butina splits. While conceptually appealing, these approaches introduce additional design choices and complexity: clustering performance strongly depends on hyperparameters, and the resulting clusters do not always align with functional differences relevant for the property of interest. As a result, while UMAP-splitting and other clustering-based splits do sometimes represent a more challenging benchmark for models, they may not consistently deliver the desired separation between chemical series, and they can be sensitive to implementation details and dataset idiosyncrasies.

SPECTRA is a new framework from the biomolecular literature that generates splits based on spectral property graphs to reduce cross-split overlap between train and test sets (see Experimental Section for details).[17] SPECTRA has already been applied to DNA and protein sequence data for phenotype prediction tasks. Because it provides an interface for a user-defined spectral property, it can be modified to assess small molecule dataset similarity, enabling us to use SPECTRA splits to better understand the relationship between train and test set overlap and model generalization.

In this study, we investigate whether current state-of-the-art data splitting strategies truly deliver dissimilar train and test sets for measuring model generalizability. To accomplish this goal, we assess the performance of several models on eight molecular property prediction datasets from MoleculeNet[18], comparing model performance and cross-split overlap for random, scaffold, UMAP, and SPECTRA splits. We find that (1) scaffold and UMAP splits do not always produce "harder" splits than random splitting; (2) therefore, split type is not well-correlated with model performance, a commonly-used proxy for generalizability; and (3) the cross-split overlap (as defined below) is a better, model-agnostic metric for defining dataset similarity and for predicting model performance. Our findings highlight the need for explicit measurements of similarity between train and test sets for any splitting method, and for benchmarks that incorporate multiple splitting strategies with increasingly dissimilar train and test sets to assess generalizability for traditional QSPR and deep learning models alike.

Experimental Section

Dataset Pre-processing

Eight datasets were obtained from the MoleculeNet[18] database including five classification tasks (BACE, BBBP, Tox21, SIDER, and ClinTox) and three regression tasks (ESOL, FreeSolv, and Lipophilicity). Datasets containing more than 10,000 molecules (e.g., HIV) were excluded from the initial selection due to the computational cost of finding SPECTRA splits for these large datasets. A description of each dataset is provided in Table 1.

Table 1: Description of datasets obtained from MoleculeNet. All classification tasks are binary classification tasks. Datasets of task type "Classification:n" are multi-task classification datasets where n properties are provided per SMILES string. "Phys. Chem" stands for physical chemistry. Categorizations are provided by MoleculeNet.[18]

Category	Dataset	Task Type	# Compounds	Description
Biophysics	BACE	Classification	1513	Binding results of BACE-1 inhibitors
Physiology	BBBP	Classification	2050	Blood-brain barrier permeability
Physiology	Tox21	Classification:12	7831	Toxicity
Physiology	SIDER	Classification:27	1427	Reported side effects
Physiology	ClinTox	Classification:2	1484	Clinical trial toxicity
Phys. Chem.	ESOL	Regression	1128	Water solubility
Phys. Chem.	FreeSolv	Regression	642	Hydration free energies in water
Phys. Chem.	Lipophilicity	Regression	4200	Octanol/Water distribution coefficient

For classification tasks, the area under the receiver operating characteristic (AUC) was used as the performance metric. For regression tasks, the root mean squared error (RMSE) was used.

SMILES strings were processed using RDKit[14] by converting SMILES string into molecular objects. Entries that returned None or with errors (for example, incorrect valences) were excluded due to invalid SMILES representation. To identify replicate structures, each SMILES string was converted to its canonical form by generating a molecular object and re-encoding it to a SMILES string. This approach standardizes different representations of the same molecule into a single canonical form for comparison. Replicate entries were handled based on task type. In classification task, a single instance was kept if the labels between replicate match, while all replicates were removed if mismatch in labels occurred. In regression tasks, all replicate were removed.

Finally, SMILES strings were also converted to Morgan fingerprints using RDKit with radius 2, and 1024-bit vector size.[14]

Cross-Split Overlap (CSO)

The **cross-split overlap** metric was used to quantify the similarity between training and test sets after data splitting, following the precedent for small molecule similarity metrics used in the Ektefaie *et al* usage examples.[17] The cross-split overlap between a train set N and a test set M is computed by taking the average pairwise similarity between objects in N and M.

In our study, the objects in the train and test sets are molecules represented by binary Morgan fingerprints. Molecular similarity between two molecules A and B was determined by computing the Tanimoto similarity between these binary fingerprints, again following the example usage shown in the code associated with the work in Ektefaie *et al* (Equation 1).[17,19]

$$S_{A,B} = \frac{A \cap B}{A \cup B} \tag{1}$$

Note that for highly similar molecules, the ratio of the intersection to the union of the molecular fingerprints will be close to 1; for highly dissimilar molecules, the ratio is close to 0. Given a train set N and test set M, the cross-split overlap was calculated as in Equation 2.

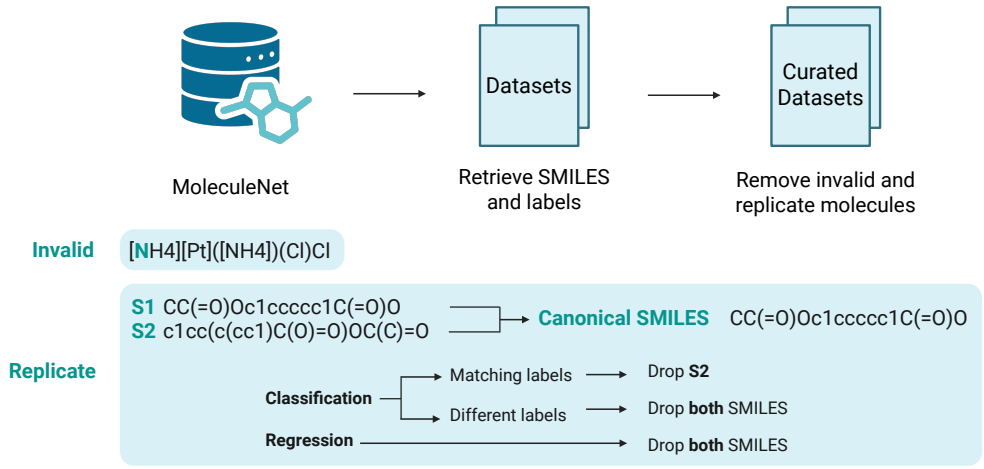


Figure 1: Data pre-processing pipeline used to curate each of the eight MoleculeNet datasets. Methods for detecting invalid molecules and handling replicate molecules are illustrated.

$$\text{Cross-split overlap} = \frac{1}{|N||M|} \sum_{A \in N} \sum_{B \in M} S_{A,B} \tag{2}$$

This metric provides insight into how similar or dissimilar the train and test set are to one another. A cross-split overlap near 1 shows very high similarity between the train and test set; a cross-split overlap near 0 shows almost no similarity between train and test set. Lower cross-split overlap can be expected to indicate a more challenging prediction task. Note that the possible range of the cross-split overlap differs per similarity metric (though we only use Tanimoto similarity in this work), as well as per dataset.

Splitting strategies

Random and Scaffold

Random splitting was performed by passing Morgan fingerprints to the random split function in DeepChem.[20] Scaffold splits were performed by first assigning each molecule in a dataset a Bemis-Murcko scaffold, and then randomly assigning groups of molecules with the same scaffold to either test or train. In both cases, an 80:20 train:test split ratio was enforced with a tolerance of 1% to allow for cases when a perfect 80:20 division was not possible. In all cases, this tolerance was sufficient. For each dataset and each split type, five different replicate splits were generated.

Uniform Manifold Approximation and Projection (UMAP)

The 1024-bit Morgan fingerprints were reduced to two dimensions using the Uniform Manifold Approximation and Projection (UMAP) method. The UMAP clustering split was then performed following the method proposed by the Ballester group.[21] Briefly, k-means clustering was applied to the two-dimensional embedding to group the molecules into seven clusters. Seven UMAP clusters were found to be optimal relative to other clustering methods. Molecules located close together in the UMAP embedding are expected to share similar features and were assigned to the same cluster. The cluster with the number of molecules closest to 20% of the dataset was selected as the test set, and the remaining six clusters were combined to form the training set. Five replicates with five different random seeds were generated for each dataset.

SPECTRA splits

In the small molecule framework, SPECTRA constructs a molecular similarity network called spectral property graph (SPG), in which molecules represented by 1024-bit Morgan fingerprints form the nodes. Edges connecting related molecules are established using a user-defined spectral property. In this study, the spectral property was based on Tanimoto similarity, such that molecules with similarity values exceeding a specific threshold were connected by an edge in SPG. Splits were generated across a range of spectral parameter (SP). As the SP increased, edges and connected nodes were gradually removed to produce increasingly distinct training and test sets. Cross-split overlap was quantified for each split at a given SP by calculating the average pairwise Tanimoto similarity between molecules in the training and test sets, as described in Equation 2.

Using this approach, a series of 60 train–test splits with an 80:20 ratio were generated, corresponding to SP values ranging from 0.00 to 1.00 in increments of 0.05, with three replicates per SP. For datasets containing a small number of molecules or highly structurally similar compounds, all molecules could be removed from the SPG before reaching an SP value of 1.00.

Models used for training

Classical models

Classical (not deep learning) machine learning models were trained on each dataset in order to assess performance on different split types. All classical models were implemented using the scikit-learn Python package.[22]

For binary classification tasks, we trained logistic regression models (LogReg), random forest models (RF), support vector machine models (SVM), and XGBoost models (XGB). The LogReg models were implemented as LASSO regression models (L1-regularlized) using the saga solver and a maximum of 1500 iterations. The SVM models were initialized with a radial basis function kernel. Both of these non-default implementation decisions were following the precedent in the MoleculeNet paper.[18] The RF and XGB models were implemented using scikit-learn defaults.

For regression tasks, we trained linear regression models (LinReg) and kernel ridge regression models (KRR). The LinReg models were implemented as LASSO regression models (L1-regularized). The KRR models were initialized with a radial basis function kernel. Both of these non-default implementation decisions were following the precedent in the MoleculeNet paper.[18] No further hyperparameter optimization was pursued.

Chemprop

We trained Chemprop v2.2.1 models on the MoleculeNet benchmark datasets.[7,8] Hyperparameters were optimized using 100 search iterations and 200 epochs to determine the optimal values for the following parameters: the number of message passing steps, the hidden size during message passing, the number of layers in the feed-forward neural network, the hidden size, dropout ratio, learning rate, warm-up period, and batch size. The random split with a ratio 80:10:10 for training, validation, and test sets respectively for each dataset was applied in this process.

To assess Chemprop v2.2.1 performance across the four major split types used in this study, we trained and evaluated models using the optimal hyperparameters identified in the previous step. Datasets were first divided into training and test sets using an 80:20 split ratio, as previously described. Then, the training set was randomly split into a new training and validation set with a 70:10 ratio, resulted in a final split of 70:10:20 for training, validation, and test sets. The validation set is required to evaluate the loss curve and enable early stopping. SPECTRA splits with validation sets containing fewer than two molecules were excluded from Chemprop performance evaluation. Models were trained and tested on each dataset and splitter using 200 epochs, an ensemble size of 5, normalization aggregation, and a patience value of 15.

Data analysis

Assessing split difficulty

The relative difficulty of a data split is often framed intuitively, with scaffold splits being considered "harder" than random splits, and UMAP splits being consider "harder" than either of those two splits, based on perceived rigorousness of the splitting process. It is assumed that the "harder" splitting methods are partitioning test data that is more dissimilar to the training data, thus approximating model generalization to unseen data.

We use two objective measurements to quantify split difficulty, rather than relying on subjective perception of rigorousness. First, we use model performance as a proxy measurement of split difficulty; here we assume that for a more difficult split, model performance should go down. This definition of split difficulty is ubiquitously used in the literature. For example, Guo *et al* claim the UMAP split is more difficult than the Butina split due to an increased hit rate (improved performance) with the latter split. However, because model performance is necessarily *model-dependent*, this strategy creates a circularity when assessing the generalizability of a particular model. A fully generalizable model should, by definition, be able to *retain good performance even for difficult splits*. Thus, defining splits as difficult based on model performance means that no model can ever attain full generalization, because to perform well is to render the split "easy."

Second, we use the cross-split overlap (Equation 2) to quantitatively measure split difficulty. Because this metric is *model agnostic*, it avoids the issues of circularity with model-dependent metrics such as performance.

Statistical analyses

To compare the performance of models on MoleculeNet datasets under different splitting strategies, we performed 2-sided Wilcoxon rank-sum tests as implemented in R. When p-value adjustment was performed to control for false discovery rate, the Bonferroni correction was used. All statistical tests were performed using R.[23]

To determine whether the median number of molecules per scaffold group was correlated to cross-split overlap of the resultant scaffold splits, we first calculated the Bemis-Murcko scaffold of each molecule in each dataset. Then, we found the median number of unique molecules in each scaffold group, where a scaffold group is comprised of the molecules that belong to a unique Bemis-Murcko scaffold. The correlation between this value and the cross-split overlap was then calculated using the rank-based Spearman's ρ, as implemented in R.[23]

Results

Duplicates and invalid SMILES are present in some MoleculeNet datasets

Before modeling, we preprocessed MoleculeNet benchmark datasets (see Experimental Section). In our study, we included five classification tasks (BACE, BBBP, Tox21, SIDER, and ClinTox) and three regression tasks (ESOL, FreeSolv, and Lipophilicity). The pre-processing strategy resulted in invalid and replicated SMILES strings being removed from the BBBP, ClinTox, Tox21, and ESOL datasets (see Table 2). In BBBP, 124 replicate SMILES were detected, comprising 6% of the dataset. Among these, 74 had incongruent labels and were removed entirely from our curated dataset. A similar situation was observed in ClinTox and ESOL, where 38 and 22 molecules were removed due to incongruent replication, respectively. Overall, fewer than 5% of molecules were filtered from BBBP, ClinTox, Tox21, and ESOL (Table 1).

No invalid or replicated molecules were detected in the BACE, SIDER, FreeSolv, and Lipophilicity datasets.

Scaffold and UMAP splits are not always "harder" than random splits as measured by model performance

For each of the eight datasets included in this analysis, Chemprop and classical models were trained on random, scaffold, and UMAP data splits (see Experimental Section). The relative performance of each model on each dataset and split type is reported in Table 3. Most dataset and model combinations were

Table 2: Results of removing invalid and replicate SMILES strings from MoleculeNet datasets. The percent filtered is shown in grey if no molecules were filtered from that dataset.

Datasets	# molecules	# invalid	# replicates filtered	# molecules (final)	Filtered (%)
BACE	1513	0	0	1513	0.00
BBBP	2050	11	74	1965	4.15
ClinTox	1484	4	38	1440	2.96
ESOL	1128	0	22	1106	1.95
FreeSolv	642	0	0	642	0.00
Lipophilicity	4200	0	0	4200	0.00
SIDER	1427	0	0	1427	0.00
Tox21	7831	8	0	7823	0.10

not statistically significantly different from random in terms of performance after adjusting for multiple hypothesis testing using a Bonferroni correction, for both scaffold splits (only 5/34 combinations tested significant) and UMAP splits (only 5/34 combinations tested significant). Furthermore, UMAP splits were not always more difficult than scaffold splits, as measured by the relative decrease in model performance compared to random splits (Full data available in Supplemental File 1). For example, in the Chemprop model trained on the FreeSolv dataset, the RMSE on the random splits was 1.23; a marked decrease in performance to a 1.83 RMSE was seen for scaffold splits; and UMAP splits resulted in a median performance of 1.12, which is non-significantly better performing than the random splits. Notably, while the deep learning model, Chemprop, is consistently among the top models tested when applied to random data splits, it is also the only model that manifests statistically significant performance decrease when applied to scaffold and UMAP splits. In 5/8 cases, both splits have significantly worse performance than random. While statistical significance was not achieved for other model types, other models nonetheless often also showed a decrease in performance across split type, accompanied by higher variance in performance across replicates.

Scaffold and UMAP splits are not always "harder" than random splits as measured by cross-split overlap

When assessing split difficulty using the cross-split overlap, we observe that for 3 out of 8 datasets (BBBP, ClinTox, and Lipophilicity), no statistically significant difference in cross-split overlap is observed between 5 random splits, 5 scaffold splits, and 5 UMAP splits (see Figure 2). For a further two datasets (SIDER and Tox21), a statistically-significant decrease in CSO is only observed for scaffold splits versus random splits, but not for UMAP versus random splits. For BACE, ESOL, and FreeSolv, a statistically-significant decrease in CSO is observed between random versus scaffold splits and between random versus UMAP splits. For FreeSolv, there is a significant *increase* in CSO when comparing UMAP splits to scaffold splits. Only for BACE is there a statistically significant decrease in CSO between scaffold and UMAP splits.

We next examined whether there was a relationship between innate properties of each dataset and the cross-split overlap observed in different splits. Bemis-Murcko scaffold splitting is sensitive to minor structural variations, such as single-atom changes in a ring, and therefore some datasets may have Bemis-Murcko scaffold assignments that result in very small numbers of molecules assigned to each given scaffold. A random split can be thought of as having each molecule assigned to its own unique scaffold group; the hope with a scaffold split is that scaffold groups will be occupied by several structurally-related molecules, but in practice this is rarely explicitly assessed.

Our hypothesis was that datasets with a low average number of molecules per scaffold group would behave more like random splits, whereas datasets with a higher average number of molecules per scaffold group would be more difficult splits. We evaluated whether the difference in cross-split overlap is related to the differences in scaffold occupancy patterns across datasets. We observe a statistically-significant Spearman correlation between the mean scaffold group occupancy and cross-split overlap (Spearman's ρ = -0.88, p-value < 0.00724). The negative value of the correlation indicates that as scaffold group occupancy goes up (*i.e.* becomes further from random) the cross-split overlap decreases, indicating a more dissimilar train and test

Table 3: The median model performance on the test set is shown for random, scaffold, and UMAP splits for each dataset and each model used to train on a particular dataset. For some datasets, the relevant metric is AUC (higher is better); for others, the relevant metric is RMSE (lower is better). The scaffold vs random column shows statistically significantly different performance between scaffold and random splits. The UMAP vs random column shows statistically significantly different performance between UMAP and random splits. Significance was assessed using the 2-sided Wilcoxon rank sum test followed by Bonferroni correction on a per-column basis. ***: < 0.001, **: < 0.01, *: < 0.05, .: not significant

Dataset	Model	Metric	Random	Scaffold	UMAP	Scaffold vs Random	UMAP vs Random
BACE	Chemprop	AUC (↑)	0.76	0.86	0.49	***	***
BACE	LogReg		0.86	0.85	0.52	.	.
BACE	RF		0.87	0.88	0.59	.	.
BACE	SVM		0.88	0.89	0.61	.	.
BACE	XGB		0.87	0.87	0.56	.	.
BBBP	Chemprop	AUC (↑)	0.92	0.90	0.77	.	***
BBBP	LogReg		0.90	0.85	0.76	.	.
BBBP	RF		0.93	0.89	0.82	.	.
BBBP	SVM		0.91	0.88	0.82	.	.
BBBP	XGB		0.92	0.86	0.77	.	.
ClinTox	Chemprop	AUC (↑)	0.85	0.89	0.84	.	.
ClinTox	LogReg		0.86	0.79	0.80	.	.
ClinTox	RF		0.79	0.78	0.71	.	.
ClinTox	SVM		0.87	0.86	0.82	.	.
ClinTox	XGB		0.82	0.76	0.70	.	.
ESOL	Chemprop	RMSE (↓)	0.66	0.83	0.83	***	***
ESOL	KRR		1.79	1.88	2.08	.	.
ESOL	LinReg		2.07	2.16	2.27	.	.
FreeSolv	Chemprop	RMSE (↓)	1.23	1.83	1.12	***	.
FreeSolv	KRR		3.42	4.78	2.82	.	.
FreeSolv	LinReg		3.86	5.04	3.40	.	.
Lipophilicity	Chemprop	RMSE (↓)	0.53	0.57	0.63	**	***
Lipophilicity	KRR		1.01	1.08	1.04	.	.
Lipophilicity	LinReg		1.21	1.24	1.16	.	.
SIDER	Chemprop	AUC (↑)	0.59	0.57	0.58	.	.
SIDER	LogReg		0.61	0.60	0.55	.	.
SIDER	RF		0.65	0.63	0.57	.	.
SIDER	SVM		0.63	0.61	0.55	.	.
SIDER	XGB		0.63	0.61	0.57	.	.
Tox21	Chemprop	AUC (↑)	0.84	0.81	0.74	***	***
Tox21	LogReg		0.79	0.74	0.66	.	.
Tox21	RF		0.81	0.76	0.67	.	.
Tox21	SVM		0.81	0.77	0.69	.	.
Tox21	XGB		0.79	0.74	0.66	.	.

set. While this trend may seem appear by high scaffold group occupancy and low cross-split overlap observed in the SIDER dataset, the correlation remains of a similar magnitude and retains statistical significance when the SIDER dataset is removed (Spearman's ρ = -0.82 , p-value < 0.03413).

SPECTRA splits cover a wider CSO range than scaffold or UMAP splits

SPECTRA splits were generated for each dataset (see Experimental Section). These SPECTRA splits are each associated with an internal spectral parameter (SP) that defines properties of the spectral graph used

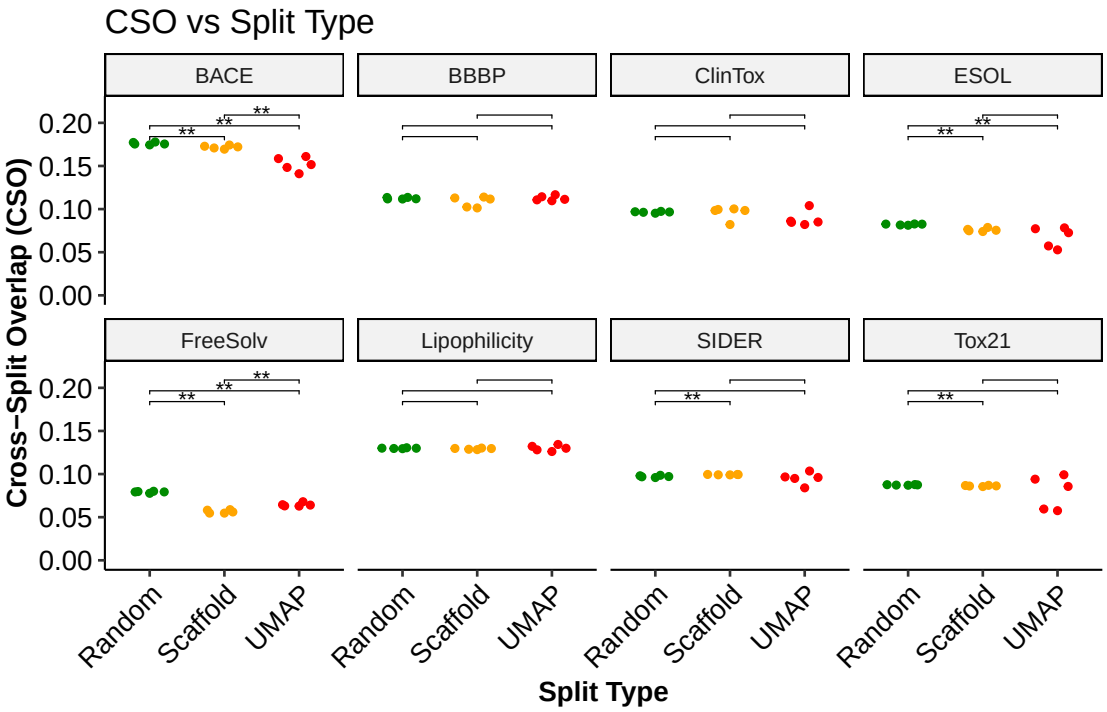


Figure 2: For each dataset and split type, the cross-split overlap is plotted. Statistical significance is determined by the 2-sided Wilcoxon rank sum test, with no p-value adjustment within or between figure panels. ***: < 0.001, **: < 0.01, *: < 0.05, blank: not significant

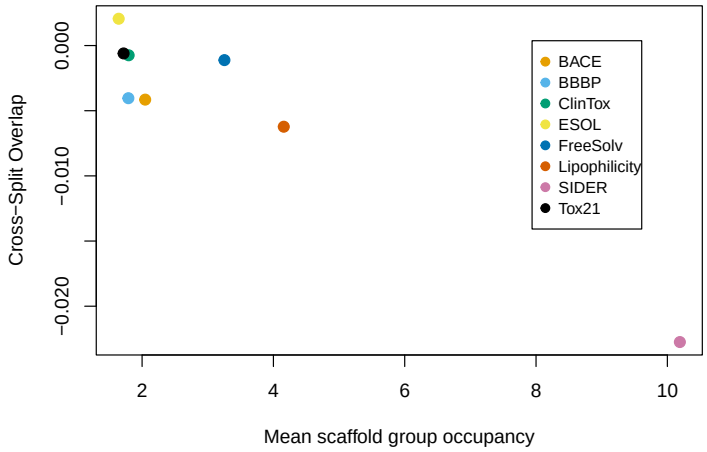


Figure 3: For each MoleculeNet dataset, the x-axis value is the mean scaffold group occupancy (mean number of molecules with a particular Bemis-Murcko scaffold) and the y-axis value is the mean cross-split overlap of the scaffold splits generated for that dataset.

to create the data split. The SPECTRA cross-split overlap series curve, which relates the CSO to SP, was then generated, and the CSO values of random, scaffold and UMAP were mapped onto this curve to identify their corresponding spectral parameters (SP) and range on the SPECTRA split spectrum (see Fig 4).

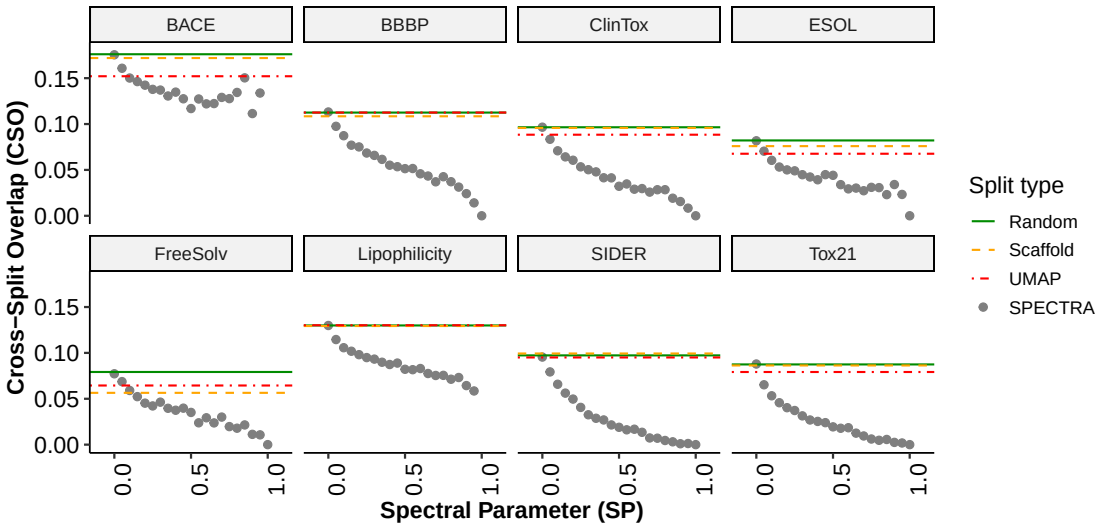


Figure 4: The relationship between CSO and SP is shown for all SPECTRA splits across all eight datasets; the mean CSO for each of random, scaffold, and UMAP splits is also shown as a horizontal line across each plot.

In these eight MoleculeNet datasets, we observe that as the spectral parameter is tuned upwards from 0.0 to 1.0, the cross-split overlap decreases and thus the similarity between the train and test set decreases (see Figure 4). Mapping the mean cross-split overlap of random, scaffold, and UMAP onto the SPECTRA cross-split overlap series curve reveals their relatively high cross-split overlap. Random splitting was expected to yield the highest cross-split overlap, followed by scaffold and UMAP with a decrease from random to scaffold and UMAP. However, the observed differences between these methods was typically small. In the BBBP, SIDER and Lipophilicity datasets, the cross-split overlap value of random, scaffold, and UMAP nearly coincide at SP = 0.0, indicating that scaffold and UMAP performance are approximately the same as random in these datasets, and still produce highly similar train and test splits within the highest overlap regime defined by SPECTRA. In the remaining datasets, although the separation between three splitting strategies is larger, they still fall within spectral parameter from 0.0 to 0.2, corresponding to the high overlapping region of SPECTRA. Across these datasets, random splits consistently exhibit the highest cross-split overlap, followed by scaffold and finally UMAP, except in FreeSolv. The scaffold split on FreeSolv shows the lowest cross-split overlap among the three strategies, corresponding to an SP of approximately 0.2.

Cross-split overlap is an effective predictor of model performance

Next we assessed the relationship between model performance and cross-split overlap across random, scaffold, UMAP, and SPECTRA splits for our datasets (see Figures 5 and 6). We hypothesized that splits with higher difficulty (lower CSO) will result in lower model performance. Figures 5 and 6 show Chemprop model performance, as it is the current state-of-the-art benchmark deep learning model, but we make plots for all models available in Supplemental File 2. The number of SPECTRA splits varied among datasets because some datasets did not reach an SP value of 1.00, and splits containing one or no molecules in the validation set were excluded from model performance analysis.

For binary classification tasks, AUC increases as CSO increases, indicating that models perform better when the train and test sets have higher relative similarity (Figure 5). We also observe that the performance of all three non-SPECTRA split types falls in the region of the curve that has relatively high CSO and overall performance. The BACE dataset is the only one where the UMAP split strategy resulted in a CSO that was lower than random (with non-overlapping error bars), and we observe a corresponding lower AUC.

For regression tasks, RMSE decreases as CSO increases, indicating the same trend: models perform better when train and test sets have higher relative similarity. We observe that random, scaffold, and UMAP performance falls in the region of the curve that has relatively high CSO and overall performance

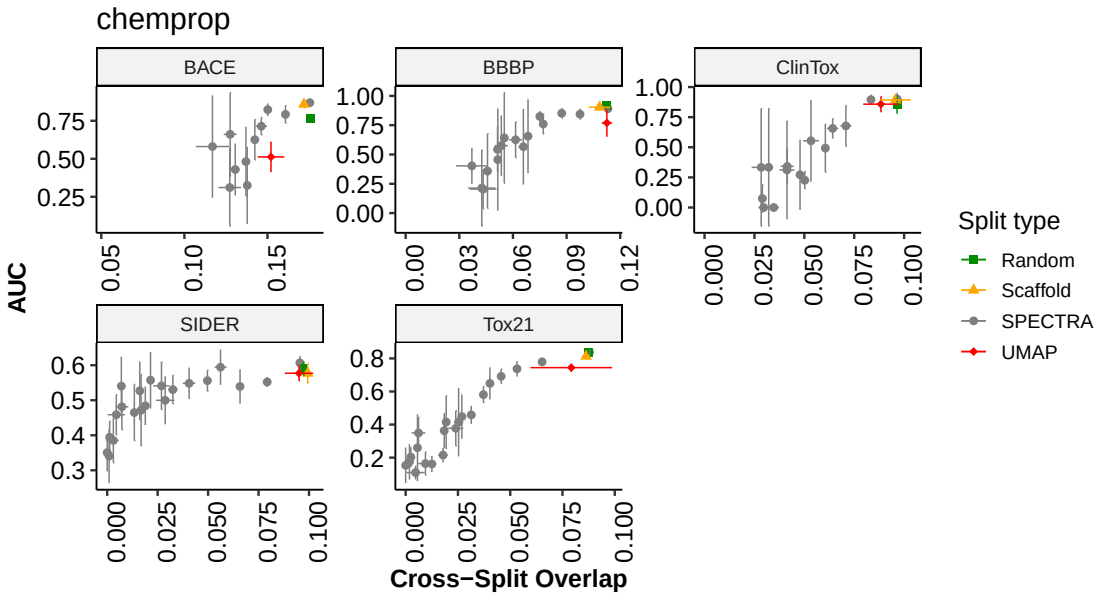


Figure 5: Model performance (AUC) is shown with its relationship to the cross-split overlap (CSO). SPECTRA splits are grouped by their spectral parameter (SP). Each point represents a mean value, with standard deviations given both for the AUC (y-axis) and for the CSO (x-axis), as each individual split that is part of the aggregation has its own individual AUC and CSO. In addition to showing model performance for the SPECTRA splits (grey circles), we also show performance for random (green squares), scaffold (yellow triangles), and UMAP (red diamonds) splits.

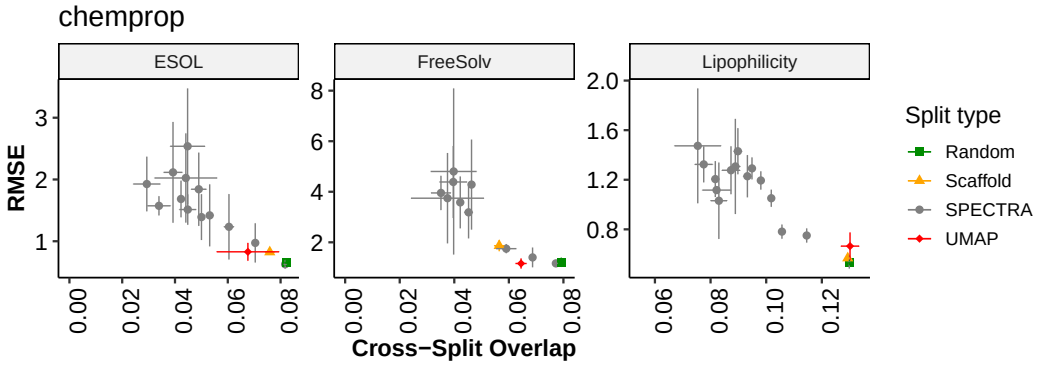


Figure 6: Model performance (RMSE) is shown with its relationship to the cross-split overlap (CSO). SPECTRA splits are grouped by their spectral parameter (SP). Each point represents a mean value, with standard deviations given both for the RMSE (y-axis) and for the CSO (x-axis), as each individual split that is part of the aggregation has its own individual RMSE and CSO. In addition to showing model performance for the SPECTRA splits (grey circles), we also show performance for random (green squares), scaffold (yellow triangles), and UMAP (red diamonds) splits.

(Fig. 6). In the FreeSolv dataset, model performance on the scaffold splits was significantly different from performance on the random and UMAP splits, which we can see explained by the model-agnostic cross-split overlap metric.

For the SPECTRA splits, Chemprop model performance exhibited greater variability in the low overlap region than in the high overlap region, which is likely due to the small number of molecules available for model training and evaluation and the difficulty of predicting highly dissimilar test molecules at very low cross-split overlap (see Supplemental File 3). In all datasets, UMAP performance tends to have higher

standard deviation compared to random and scaffold splits.

Discussion

Evaluating the generalizability of molecular property prediction models depends on two major factors: the quality of the datasets used for training and testing, and the rigor of the splitting strategy employed. In theory, a more rigorous splitting strategy creates a more dissimilar test set, such that model performance on the harder split is a better indicator of the generalizability of a model. In this study, we systematically examined both factors across eight widely used MoleculeNet benchmark datasets, employing random, scaffold, UMAP, and SPECTRA-based data splits in combination with Chemprop and classical machine learning models.

Our identification of invalid and replicate SMILES strings in several MoleculeNet benchmark datasets underscores the importance of rigorous pre-processing. The proportion of removed molecules was relatively modest (the largest being BBBP with 4.15% filtered and ClinTox with 2.96% filtered), but the conflicting labels among replicate entries could reasonably be expected to artificially affect performance metrics in publications that report model performance on MoleculeNet datasets used without pre-processing. While we are not the first to report such concerns,[24] our findings emphasize the need for care in the creation, curation, and use of common cheminformatics benchmark datasets.

We also report that, contrary to common assumption, scaffold and UMAP splits do not create reliably more challenging splits, either when assessed by resultant model performance or when assessed by a model-agnostic metric, the cross-split overlap. Across the vast majority of dataset–model combinations examined, no statistically significant difference in model performance was observed between scaffold or UMAP splits and random splits following Bonferroni correction, although we note that variability in model performance was high, especially for classical models. Even considering only Chemprop models, where performance was most sensitive to split type, in the ClinTox, FreeSolv, and SIDER datasets, no statistically-significant difference was observed between UMAP splits and random splits. Moreover, UMAP splits did not consistently produce lower performance than scaffold splits. These findings show that the choice of split strategy alone is insufficient to guarantee a rigorous out-of-distribution evaluation. For Bemis-Murcko scaffold splits, our preliminary results suggest that examining the mean scaffold group occupancy may be one way to assess the expected quality of the scaffold splits.

In contrast to the unreliable difficulty levels of splits generated with commonly-used splitting strategies, SPECTRA splits were demonstrated to generate data splits with a range of cross-split overlaps. The ability to tune the cross-split overlap by varying the spectral parameter allows researchers to deliberately probe model generalizability using varying degrees of train/test dissimilarity, rather than relying on a fixed splitting strategy.

A consistent and monotonic relationship between CSO and model performance was observed across both classification and regression tasks: AUC increased with increasing CSO in classification tasks, while RMSE decreased with increasing CSO in regression tasks. This relationship held across Chemprop and classical machine learning models (see Figs 5 and 6, as well as Supplemental File 2), and across all eight datasets examined. The elevated performance variability observed at low cross-split overlap in SPECTRA splits likely reflects both the reduced training set size at high spectral parameter values and the inherent difficulty of extrapolating to chemically dissimilar test molecules.

Taken together, these results support the use of CSO as a more effective, model-agnostic metric for quantifying split difficulty. It is also an effective predictive metric for model performance, without relying on the circularity of using model performance as a proxy for split difficulty. We suggest the use of CSO when comparing performance results on common benchmark datasets as a further method to quantify whether a particular set of splits happened to be easier, resulting in better model performance. CSO measurements can be used to determine the relative difficulty of splits generated by known methods, and we also suggest them as a means of assessing newly introduced splitting strategies, such as the "spectral splitting" method (named for its relationship to spectral graph theory) recently proposed by Klarner *et al.*[25]

Further, the area under the spectral performance curve (AUSPC), as first described in Ektefaie *et al*,[17] is a strong contender for a useful measure of generalizability of a model, as different models will create different curves with different AUSPC: a high AUSPC indicates that model performance is well-retained even under increasingly dissimilar train and test sets.

Some work has already been done evaluating the robustness of molecular property prediction models to out-of-distribution data.[26] Fooladi *et al* report a nuanced relationship between performance of models on in-distribution vs out-of-distribution data. Our work adds a valuable perspective: some splitting strategies which are commonly thought to produce out-of-distribution test sets are not actually achieving that goal when empirically measured using the cross-split overlap. This difference between assumption and reality may be driving some of the observed nuances in the relationship between split strategy and performance, as we report in our study.

Finally, several limitations of the present study merit consideration. Our analysis was restricted to eight MoleculeNet datasets and a small suite of splitting methods, and it remains to be determined whether the observed relationships between CSO, split strategy, and model performance generalize to larger or more structurally diverse chemical spaces. This limitation is especially relevant as the field expands into the use of foundation models, which are typically pre-trained on huge corpora of molecular data.[27] Additionally, while we selected several classical machine learning models as well as Chemprop for our initial evaluation, the behavior of deep learning architectures with different inductive biases, such as transformer-based models or pre-trained models, also warrants further investigation. SPECTRA splitting also involves a compromise: to achieve iteratively lower cross-split overlap, SPECTRA produces iteratively smaller datasets. Future work could examine whether a splitting strategy can be devised that ameliorates this considerable drawback.

Conclusions

When scientists are assessing molecular property prediction models, datasets are typically partitioned into training and test sets using various splitting strategies to simulate out-of-distribution conditions. Beyond random splitting, scaffold and UMAP-based methods have been widely adopted with the rationale that they will reduce similarity between the train and test set and thereby improve the assessment of model generalizability. Although these approaches are commonly expected to substantially reduce cross-split overlap compared to random splitting, our results show that this expectation does not consistently hold across all datasets, either as measured by model performance or by cross-split overlap. Instead, these three splitting strategies tend to occupy a narrow, high-similarity region of chemical space partitioning, as revealed by their mapping onto the SPECTRA cross-split overlap series curve.

Taken together, these findings motivate a re-examination of prevailing benchmarking practices in molecular machine learning and highlight cross-split overlap as a quantitative, model-agnostic metric for characterizing the stringency of splitting strategies. Our results elucidate the relationship between cross-split overlap and model performance, and provide an understanding of where current state-of-the-art splitting strategies, including random, scaffold, and UMAP splits, fall within the performance and cross-split overlap scheme. We report that scaffold and UMAP splits are closer to random than has commonly been thought, and propose the use of SPECTRA as a principled evaluation strategy for examining the generalizability of a model.

Acknowledgments

The authors thank members of the SAGE lab at University of Massachusetts Amherst for helpful commentary and feedback throughout this project.

The computational resources for this work were provided by the University of Massachusetts Amherst's partnership with the Unity Research Computing Platform, a multi-institutional cluster led by the University of Massachusetts and the University of Rhode Island.

Conflicts of interest

B.M.R. is a former employee of Novartis Biomedical Research.

Supporting information

The code used to produce these analyses is available at `https://github.com/vynkdo/SPECTRA_evaluation`. The following supplemental files are also made available.

- Supplemental File 1: Contains the raw, unadjusted p-values for comparing scaffold to random and UMAP to random splitting across models and datasets, as well as visualizations of the same data.
- Supplemental File 2: Contains plots that show model performance and its relationship to CSO across all models not showcased in the main text.
- Supplemental File 3: Contains a summary figure and tables reporting the number of molecules in the training, validation, and test sets generated by SPECTRA for all eight MoleculeNet datasets.

References

(1) Lipinski, C. A. *Journal of Pharmacological and Toxicological Methods* **2000**, *44*, 235–249.

(2) Ertl, P. *Journal of Chemical Information and Computer Sciences* **2003**, *43*, 374–380.

(3) Bohacek, R. S.; McMartin, C.; Guida, W. C. *Medicinal Research Reviews* **1996**, *16*, 3–50.

(4) Polishchuk, P. G.; Madzhidov, T. I.; Varnek, A. *Journal of Computer-Aided Molecular Design* **2013**, *27*, 675–679.

(5) Formanek, A.; Vincze, A.; Bicsak, R.; Balogh, G. T.; Moreau, Y.; Arany, Á. *Journal of Chemical Information and Modeling* **2026**, *66*, 5508–5518.

(6) Yang, K.; Swanson, K.; Jin, W.; Coley, C.; Eiden, P.; Gao, H.; Guzman-Perez, A.; Hopper, T.; Kelley, B.; Mathea, M.; Palmer, A.; Settels, V.; Jaakkola, T.; Jensen, K.; Barzilay, R. *Journal of Chemical Information and Modeling* **2019**, *59*, 3370–3388.

(7) Heid, E.; Greenman, K. P.; Chung, Y.; Li, S.-C.; Graff, D. E.; Vermeire, F. H.; Wu, H.; Green, W. H.; McGill, C. J. *Journal of Chemical Information and Modeling* **2024**, *64*, 9–17.

(8) Graff, D. E.; Morgan, N. K.; Burns, J. W.; Doner, A. C.; Li, B.; Li, S.-C.; Manu, J.; Menon, A.; Pang, H.-W.; Wu, H.; Zalte, A. S.; Zheng, J. W.; Coley, C. W.; Green, W. H.; Greenman, K. P. *Journal of Chemical Information and Modeling* **2026**, *66*, 28–33.

(9) Lionta, E.; Spyrou, G.; Vassilatis, D.; Cournia, Z. *Current Topics in Medicinal Chemistry* **2014**, *14*, 1923–1938.

(10) MLPDS – Machine Learning for Pharmaceutical Discovery and Synthesis Consortium, en-US.

(11) Kretschmer, F.; Seipp, J.; Ludwig, M.; Klau, G. W.; Böcker, S. *Nature Communications* **2025**, *16*, 554.

(12) Chen, L.; Cruz, A.; Ramsey, S.; Dickson, C. J.; Duca, J. S.; Hornak, V.; Koes, D. R.; Kurtzman, T. *PLOS ONE* **2019**, *14*, ed. by Zhang, Y., e0220113.

(13) Bemis, G. W.; Murcko, M. A. *Journal of Medicinal Chemistry* **1996**, *39*, 2887–2893.

(14) Landrum, G. RDKit.

(15) Guo, Q.; Hernandez-Hernandez, S.; Ballester, P. J. *Journal of Cheminformatics* **2025**, *17*, 94.

(16) Orlov, A. A.; Akhmetshin, T. N.; Horvath, D.; Marcou, G.; Varnek, A. *Molecular Informatics* **2025**, *44*, e202400265.

(17) Ektefaie, Y.; Shen, A.; Bykova, D.; Marin, M. G.; Zitnik, M.; Farhat, M. *Nature Machine Intelligence* **2024**, *6*, 1512–1524.

(18) Wu, Z.; Ramsundar, B.; Feinberg, E. N.; Gomes, J.; Geniesse, C.; Pappu, A. S.; Leswing, K.; Pande, V. *Chemical Science* **2018**, *9*, 513–530.

(19) Bajusz, D.; Rácz, A.; Héberger, K. *Journal of Cheminformatics* **2015**, *7*, 20.

(20) Ramsundar, B.; Eastman, P.; Walters, P.; Pande, V., *Deep learning for the life sciences: applying deep learning to genomics, microscopy, drug discovery, and more*, First edition, OCLC: 1303366952; O'Reilly: Beijing, 2019.

(21) Guo, Q.; Hernandez-Hernandez, S.; Ballester, P. J. Scaffold Splits Overestimate Virtual Screening Performance, arXiv:2406.00873 [q-bio], 2024.

(22) Pedregosa, F. et al. *Journal of Machine Learning Research* **2011**, *12*, 2825–2830.

(23) Team, R. C. R: A Language and Environment for Statistical Computing, Vienna, Austria, 2024.

(24) Walters, P. We Need Better Benchmarks for Machine Learning in Drug Discovery, 2023.

(25) Klarner, L.; Rudner, T. G. J.; Reutlinger, M.; Schindler, T.; Morris, G. M.; Deane, C.; Teh, Y. W. In *Proceedings of the 40th International Conference on Machine Learning*, PMLR: 2023; Vol. 202, pp 17176–17197.

(26) Fooladi, H.; Vu, T. N. L.; Mathea, M.; Kirchmair, J. *Journal of Chemical Information and Modeling* **2025**, *65*, 9871–9891.

(27) Choi, J.; Nam, G.; Choi, J.; Jung, Y. *JACS Au* **2025**, *5*, 1499–1518.

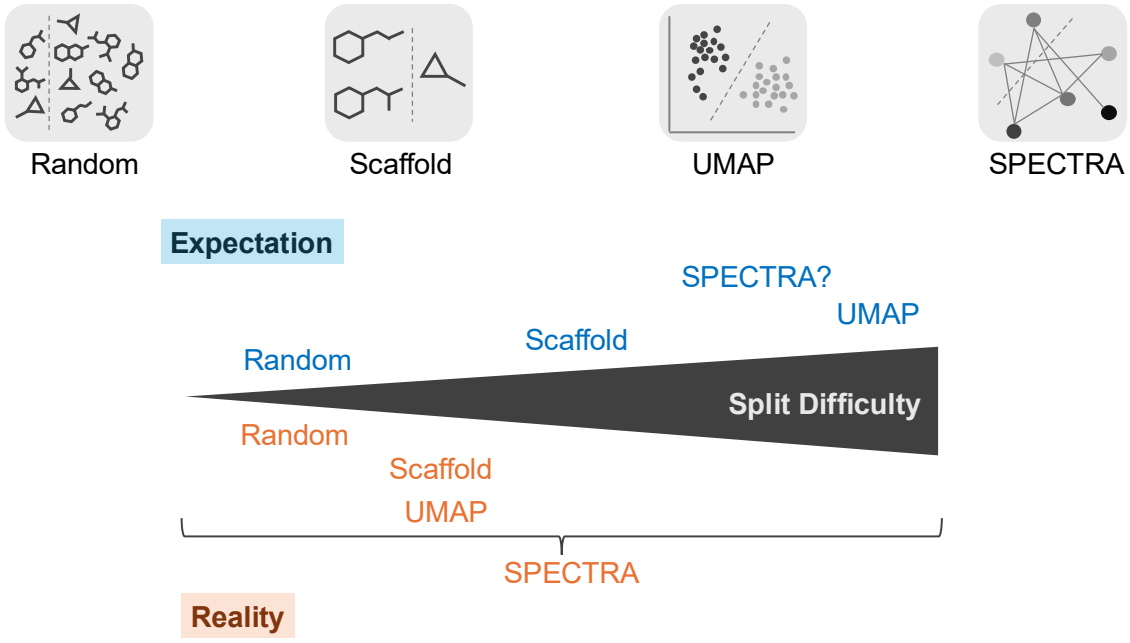


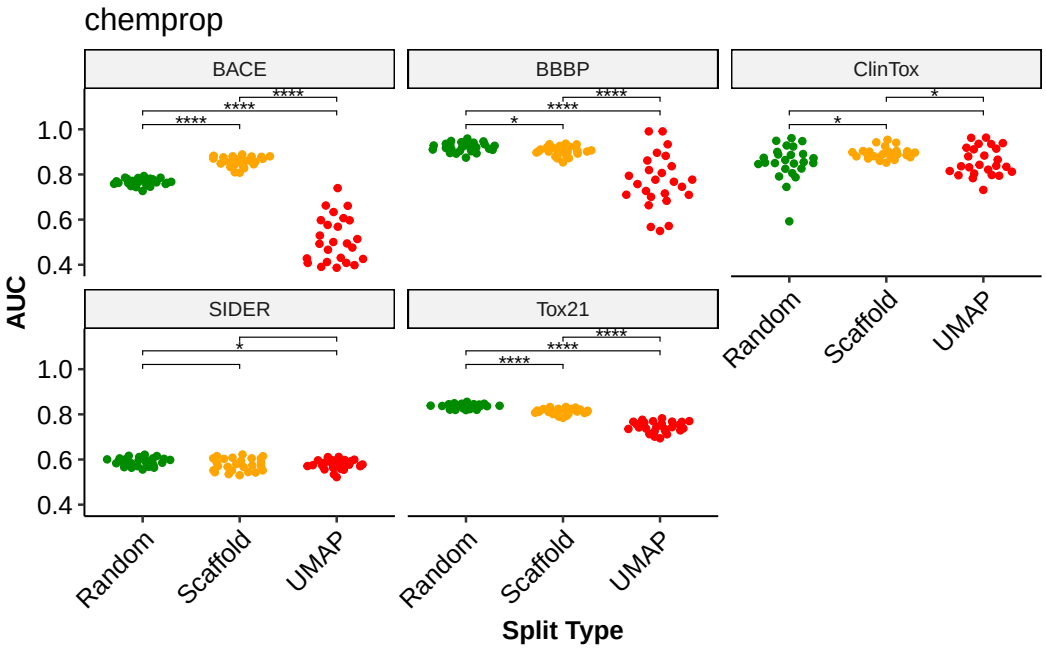
Figure 7: Graphical Abstract

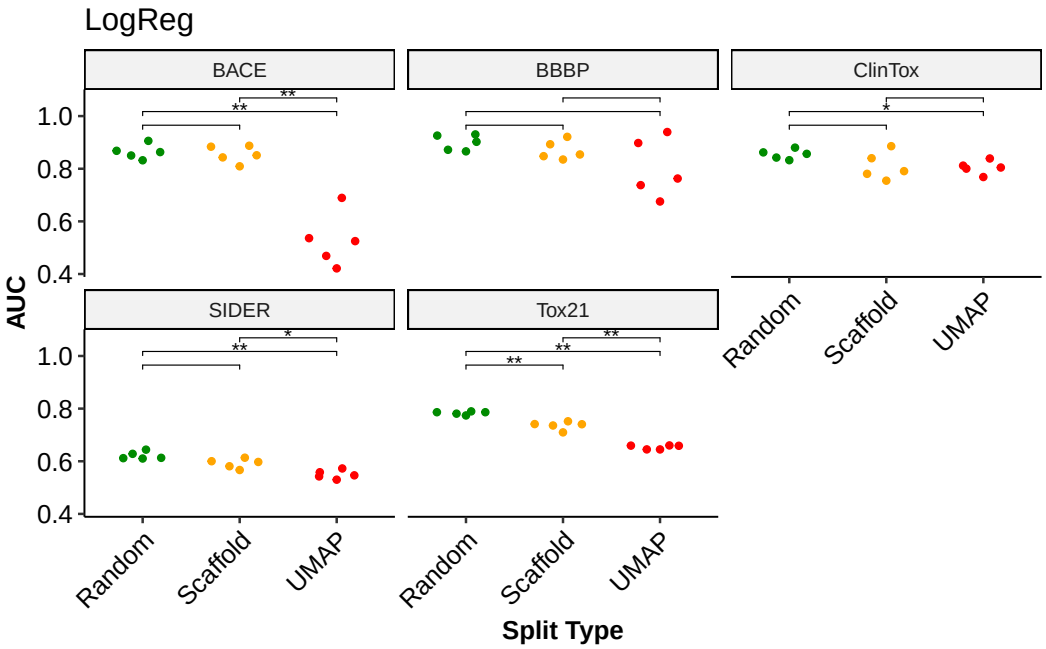
Supplemental File 1

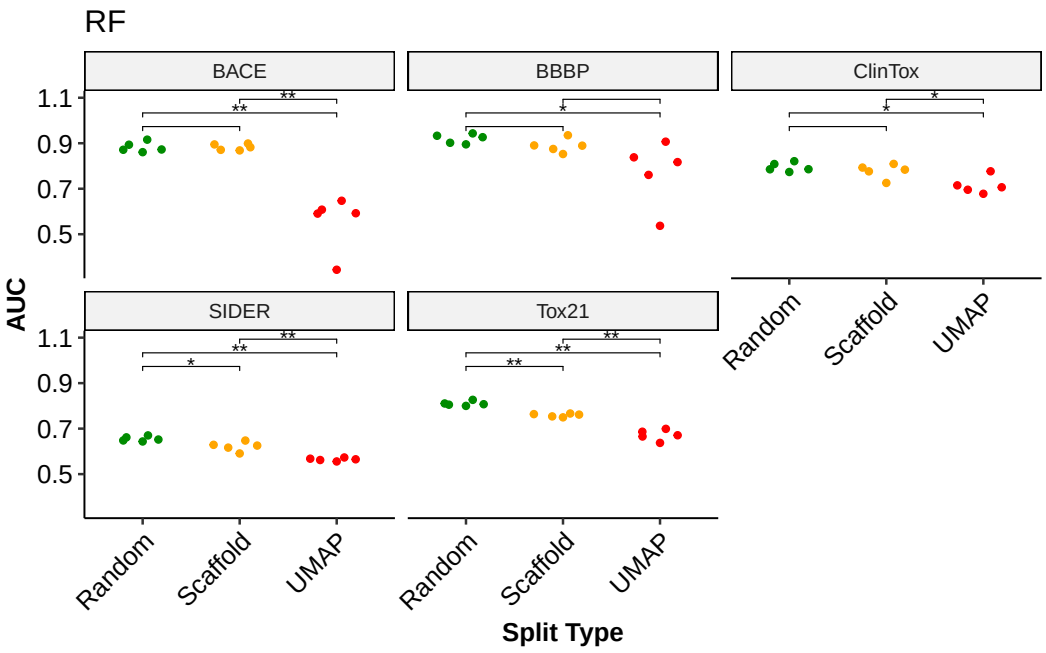
The following table reports the unadjusted 2-sided Wilcoxon rank sum p-values that underlie Table 3 in the main text. The median model performance on the test set is shown for random, scaffold, and UMAP splits for each dataset and each model used to train on a particular dataset. For some datasets, the relevant metric is AUC (higher is better); for others, the relevant metric is RMSE (lower is better). The scaffold vs random column shows statistically significantly different performance between scaffold and random splits. The UMAP vs random column shows statistically significantly different performance between UMAP and random splits.

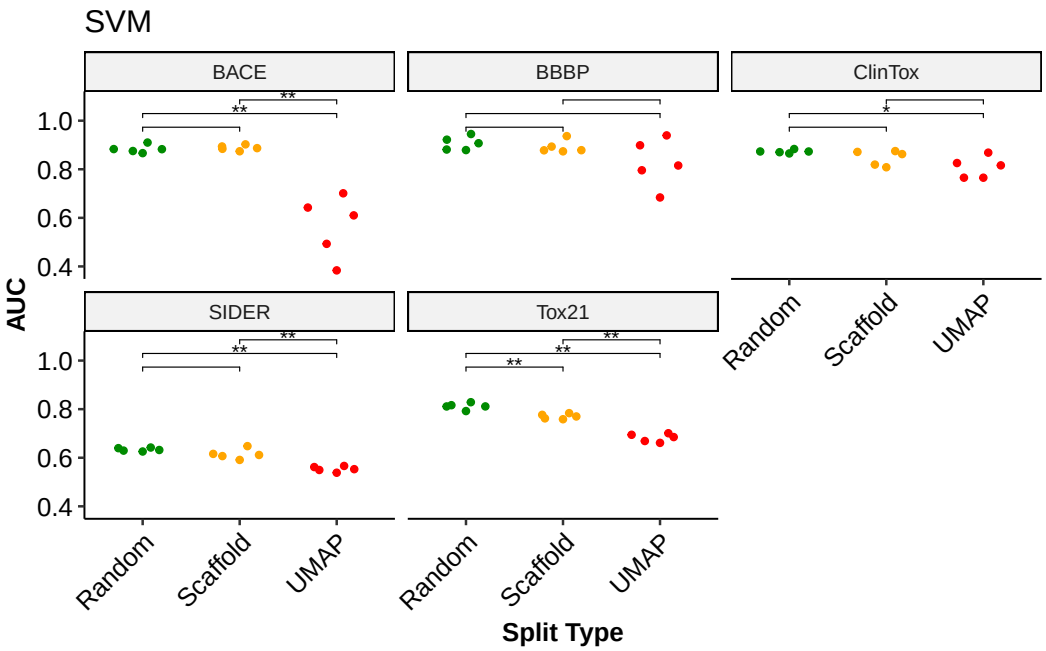
Dataset	Model	Metric	Random	Scaffold	UMAP	Scaffold vs Random	UMAP vs Random
BACE	Chemprop	AUC (↑)	0.76	0.86	0.49	0.00	0.00
BACE	LogReg		0.86	0.85	0.52	0.84	0.01
BACE	RF		0.87	0.88	0.59	1.00	0.01
BACE	SVM		0.88	0.89	0.61	0.42	0.01
BACE	XGB		0.87	0.87	0.56	1.00	0.01
BBBP	Chemprop	AUC (↑)	0.92	0.90	0.77	0.01	0.00
BBBP	LogReg		0.90	0.85	0.76	0.15	0.31
BBBP	RF		0.93	0.89	0.82	0.10	0.03
BBBP	SVM		0.91	0.88	0.82	0.22	0.22
BBBP	XGB		0.92	0.86	0.77	0.15	0.15
ClinTox	Chemprop	AUC (↑)	0.85	0.89	0.84	0.01	0.69
ClinTox	LogReg		0.86	0.79	0.80	0.22	0.02
ClinTox	RF		0.79	0.78	0.71	0.55	0.02
ClinTox	SVM		0.87	0.86	0.82	0.22	0.02
ClinTox	XGB		0.82	0.76	0.70	0.03	0.02
ESOL	Chemprop	RMSE (↓)	0.66	0.83	0.83	0.00	0.00
ESOL	KRR		1.79	1.88	2.08	0.10	0.15
ESOL	LinReg		2.07	2.16	2.27	0.31	0.15
FreeSolv	Chemprop	RMSE (↓)	1.23	1.83	1.12	0.00	0.20
FreeSolv	KRR		3.42	4.78	2.82	0.02	0.01
FreeSolv	LinReg		3.86	5.04	3.40	0.03	0.02
Lipophilicity	Chemprop	RMSE (↓)	0.53	0.57	0.63	0.00	0.00
Lipophilicity	KRR		1.01	1.08	1.04	0.22	1.00
Lipophilicity	LinReg		1.21	1.24	1.16	0.69	0.69
SIDER	Chemprop	AUC (↑)	0.59	0.57	0.58	0.14	0.04
SIDER	LogReg		0.61	0.60	0.55	0.06	0.01
SIDER	RF		0.65	0.63	0.57	0.02	0.01
SIDER	SVM		0.63	0.61	0.55	0.15	0.01
SIDER	XGB		0.63	0.61	0.57	0.06	0.01
Tox21	Chemprop	AUC (↑)	0.84	0.81	0.74	0.00	0.00
Tox21	LogReg		0.79	0.74	0.66	0.01	0.01
Tox21	RF		0.81	0.76	0.67	0.01	0.01
Tox21	SVM		0.81	0.77	0.69	0.01	0.01
Tox21	XGB		0.79	0.74	0.66	0.01	0.01

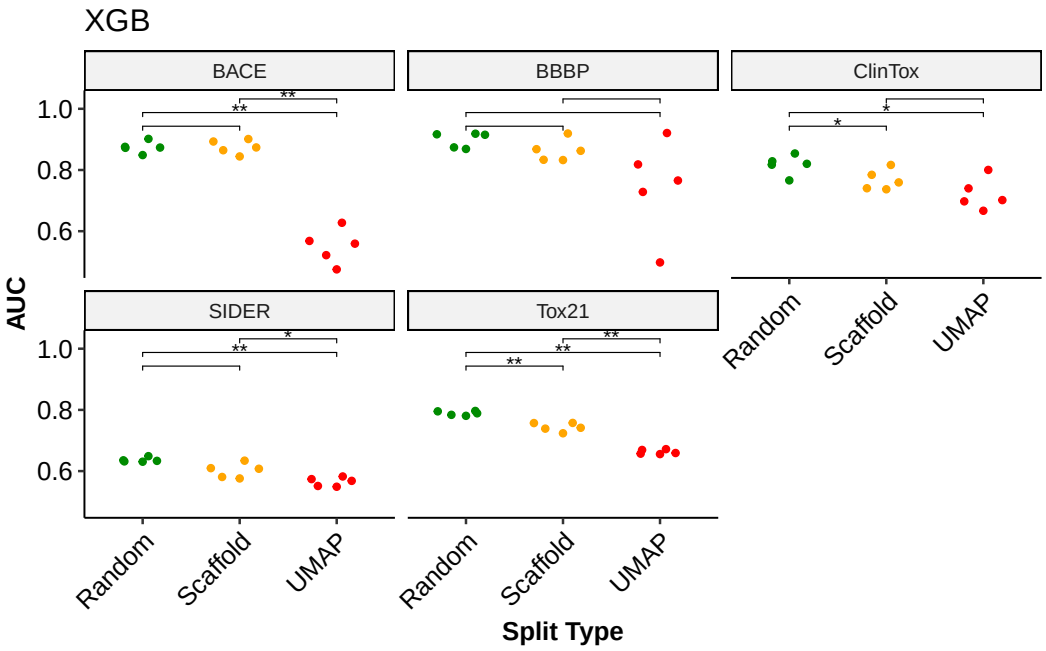
This supplemental file also contains plots that show model performance and its relationship to split type for random, scaffold, and UMAP splits, the contents of which were summarized in Table 3 in the main text and in the table above.

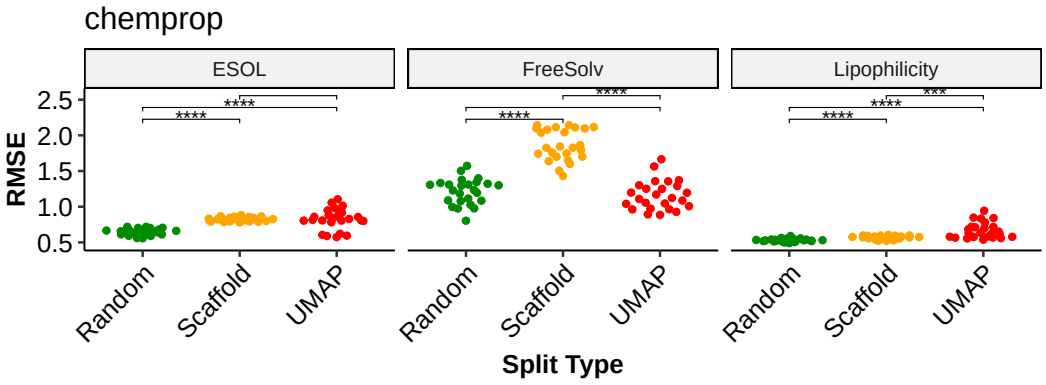

chemprop
BACE
BBBP
ClinTox
SIDER
Tox21
AUC
1.0
0.8
0.6
0.4
Random
Scaffold
UMAP
Split Type

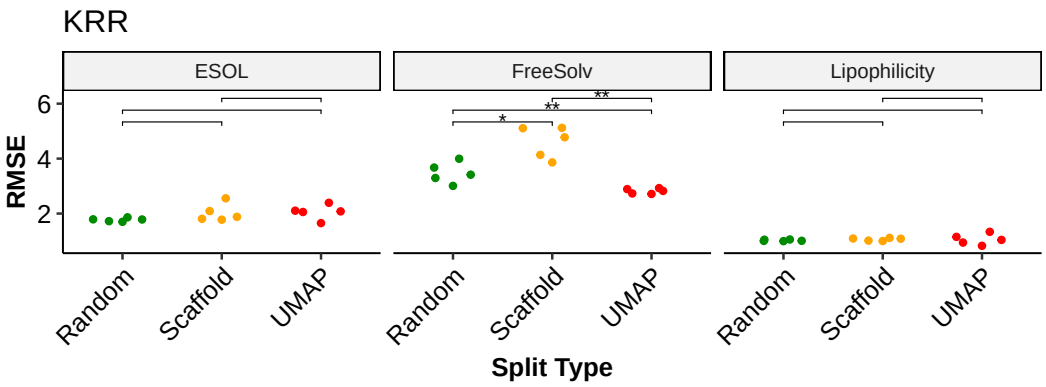

LogReg
BACE
BBBP
ClinTox
SIDER
Tox21
AUC
1.0
0.8
0.6
0.4
Random
Scaffold
UMAP
Split Type

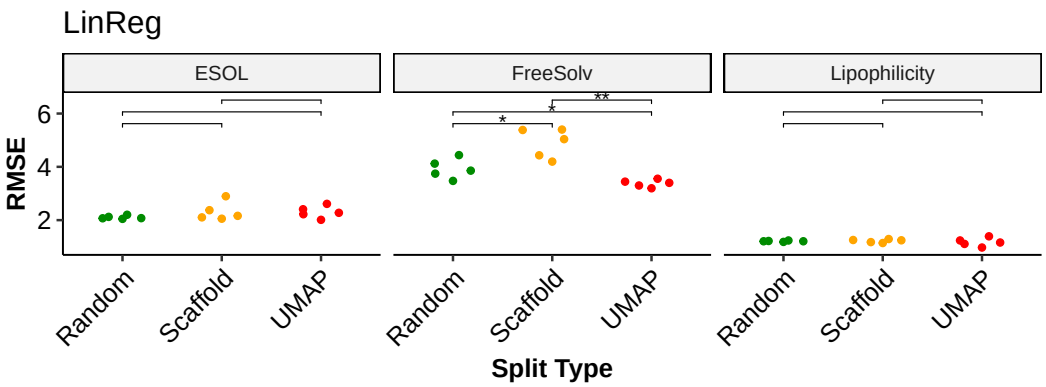

RF
BACE
BBBP
ClinTox
SIDER
Tox21
AUC
1.1
0.9
0.7
0.5
Random
Scaffold
UMAP
Split Type


SVM
BACE
BBBP
ClinTox
SIDER
Tox21
AUC
1.0
0.8
0.6
0.4
Random
Scaffold
UMAP
Split Type


XGB
BACE
BBBP
ClinTox
SIDER
Tox21
AUC
1.0
0.8
0.6
Random
Scaffold
UMAP
Split Type

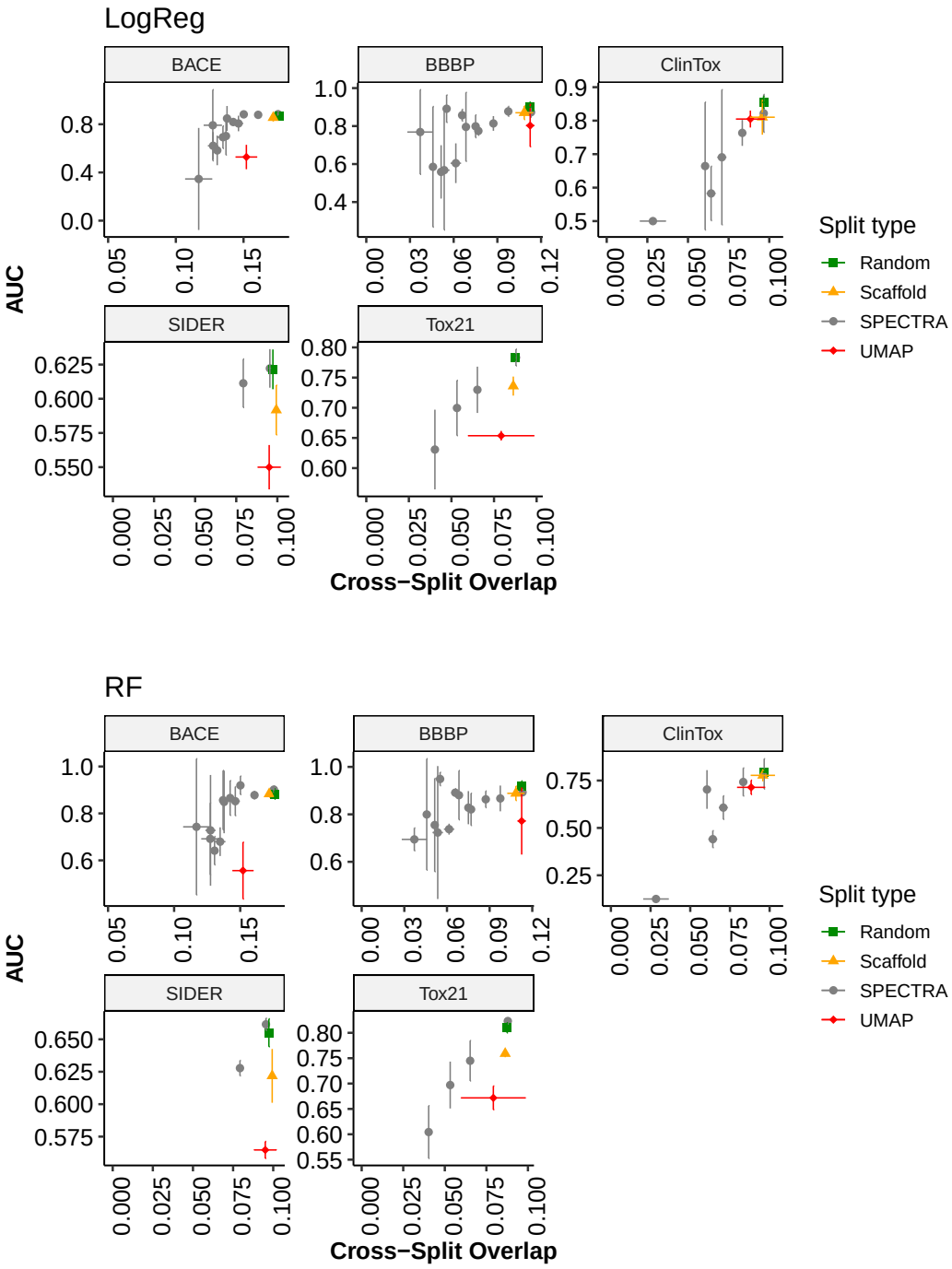

chemprop
ESOL
FreeSolv
Lipophilicity
RMSE
2.5
2.0
1.5
1.0
0.5
Random
Scaffold
UMAP
Split Type

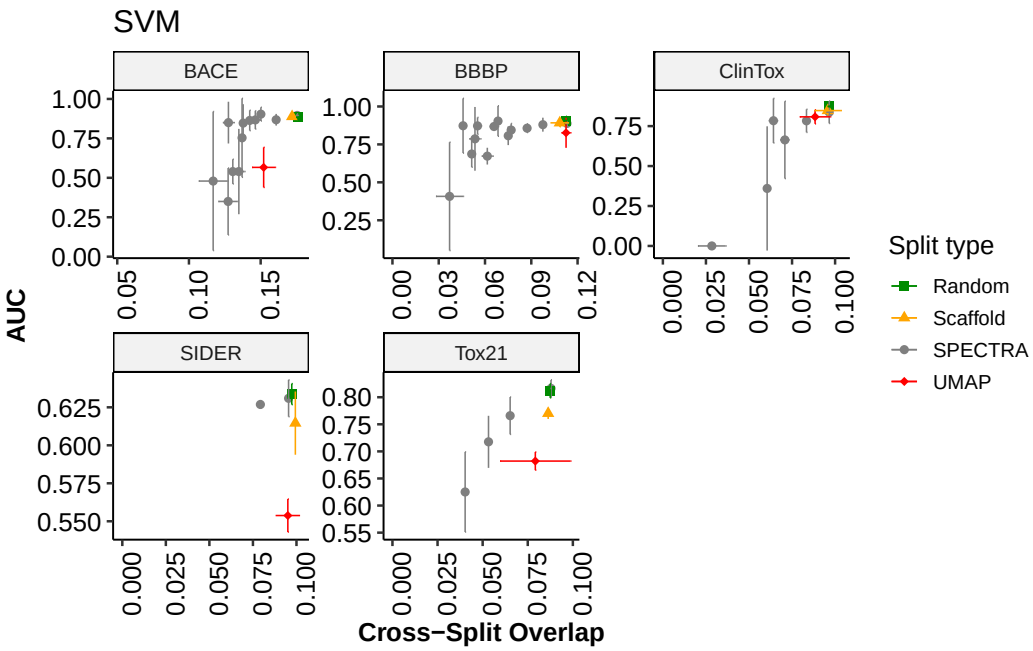

KRR
ESOL
FreeSolv
Lipophilicity
RMSE
6
4
2
Random
Scaffold
UMAP
Split Type

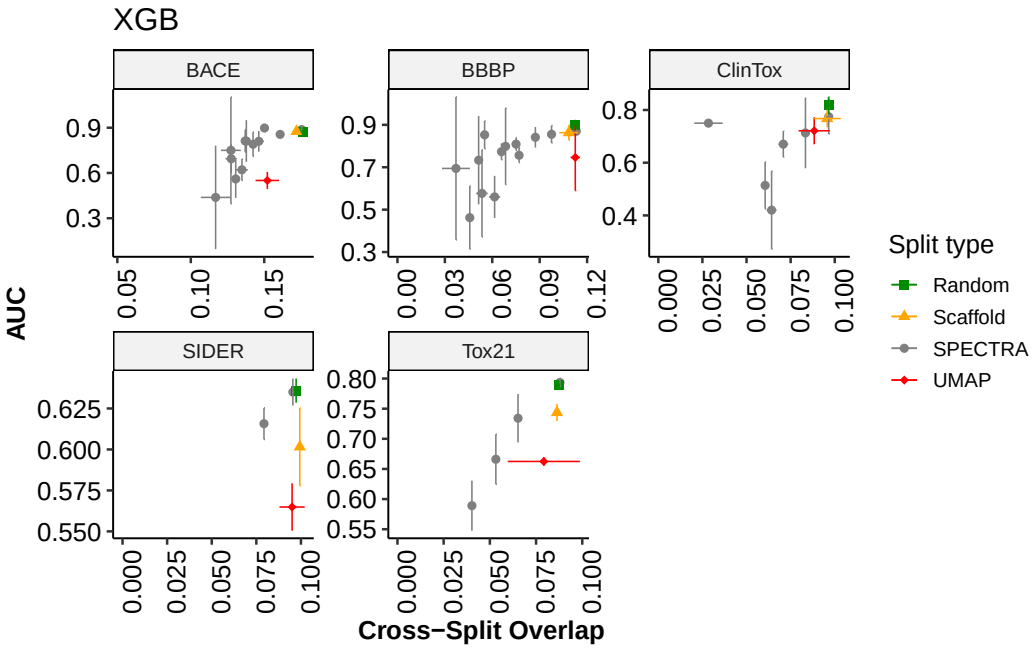

LinReg
ESOL
FreeSolv
Lipophilicity
RMSE
6
4
2
*
*
**
Random
Scaffold
UMAP
Split Type

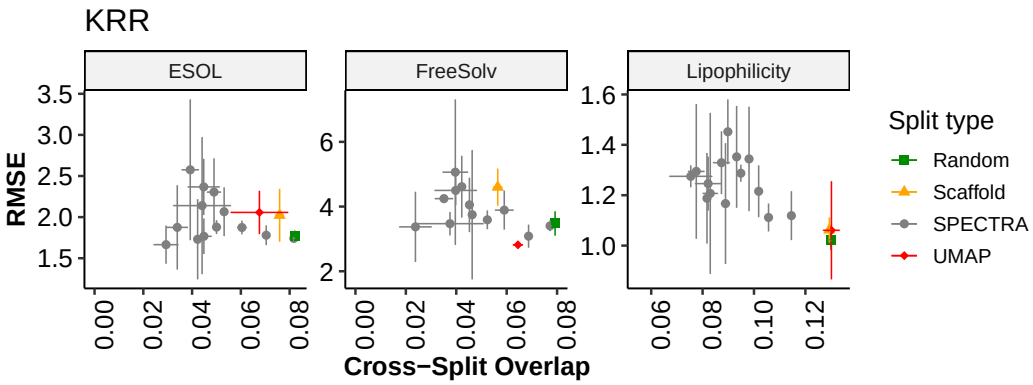
Supplemental File 2

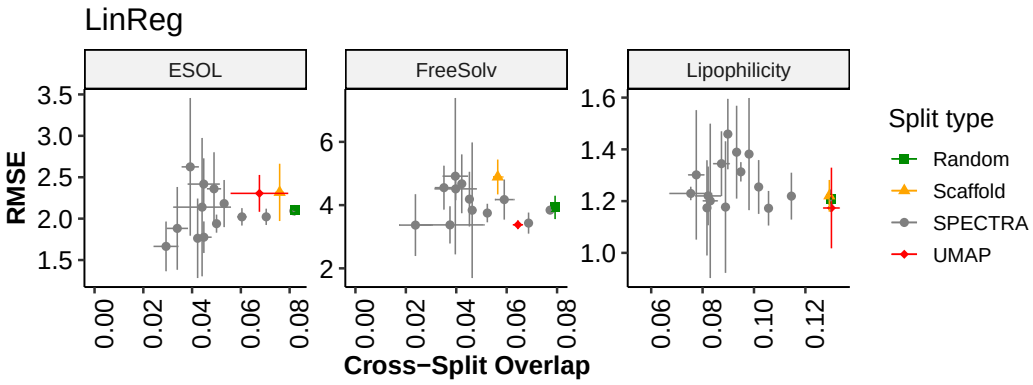
This supplemental file contains plots that show model performance and its relationship to CSO across all models assessed as part of this study. Model performance (AUC or RMSE) is shown with its relationship to the cross-split overlap (CSO). SPECTRA splits are grouped by their spectral parameter (SP). Each point represents a mean value, with standard deviations given both for the AUC (y-axis) and for the CSO (x-axis), as each individual split that is part of the aggregation has its own individual AUC and CSO. In addition to showing model performance for the SPECTRA splits (grey circles), we also show performance for random (green squares), scaffold (yellow triangles), and UMAP (red diamonds) splits.




SVM
BACE
BBBP
ClinTox
SIDER
Tox21
AUC
Cross−Split Overlap
Split type
Random
Scaffold
SPECTRA
UMAP

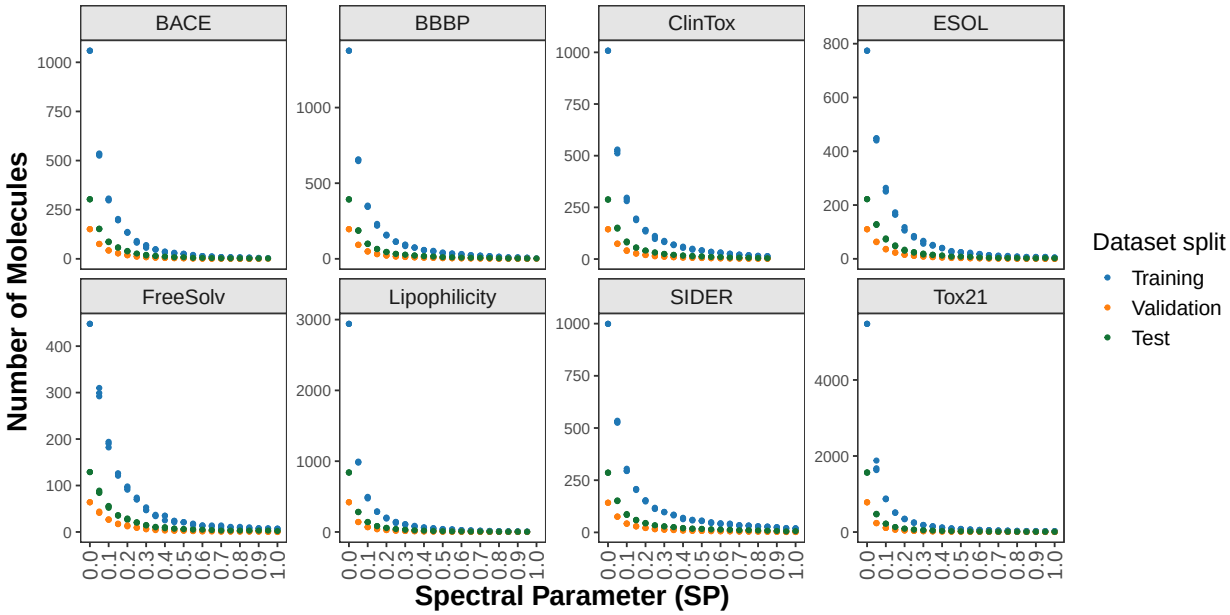

XGB
BACE
BBBP
ClinTox
SIDER
Tox21
AUC
Cross−Split Overlap
Split type
Random
Scaffold
SPECTRA
UMAP


KRR
ESOL
FreeSolv
Lipophilicity
RMSE
Cross−Split Overlap
Split type
Random
Scaffold
SPECTRA
UMAP


LinReg
ESOL
FreeSolv
Lipophilicity
RMSE
Cross−Split Overlap
Split type
Random
Scaffold
SPECTRA
UMAP

Supplemental File 3

This supplementary file contains one summary figure and eight tables reporting the sizes of the training, validation, and test sets generated by SPECTRA across the spectral parameter range for the eight MoleculeNet datasets. Three independent replicates were generated at each spectral parameter, and the number of molecules assigned to each partition is reported.



dataset	spectral_parameter	training_size	val_size	test_size
BACE	0.00_0	1059	151	303
BACE	0.00_1	1059	151	303
BACE	0.00_2	1059	151	303
BACE	0.05_0	525	75	151
BACE	0.05_1	536	76	153
BACE	0.05_2	532	76	153
BACE	0.10_0	301	43	86
BACE	0.10_1	297	42	85
BACE	0.10_2	307	43	88
BACE	0.15_0	197	28	57
BACE	0.15_1	203	28	58
BACE	0.15_2	196	28	57
BACE	0.20_0	133	18	38
BACE	0.20_1	137	19	40
BACE	0.20_2	133	19	38
BACE	0.25_0	85	12	25
BACE	0.25_1	82	11	24
BACE	0.25_2	91	13	27
BACE	0.30_0	70	9	20
BACE	0.30_1	66	9	19
BACE	0.30_2	56	8	17
BACE	0.35_0	49	7	15
BACE	0.35_1	48	6	14
BACE	0.35_2	46	6	14
BACE	0.40_0	33	4	10
BACE	0.40_1	36	5	11
BACE	0.40_2	37	5	11
BACE	0.45_0	29	4	9
BACE	0.45_1	31	4	9
BACE	0.45_2	28	3	8
BACE	0.50_0	26	3	8
BACE	0.50_1	25	3	8
BACE	0.50_2	25	3	8
BACE	0.55_0	20	2	6
BACE	0.55_1	18	2	5
BACE	0.55_2	15	2	5
BACE	0.60_0	14	1	4
BACE	0.60_1	14	1	4
BACE	0.60_2	14	2	5
BACE	0.65_0	11	1	3
BACE	0.65_1	12	1	4
BACE	0.65_2	12	1	4
BACE	0.70_0	9	1	3
BACE	0.70_1	9	1	3
BACE	0.70_2	9	1	3
BACE	0.75_0	7	1	2
BACE	0.75_1	7	1	2
BACE	0.75_2	9	1	3
BACE	0.80_0	7	0	2
BACE	0.80_1	6	0	2
BACE	0.80_2	8	1	3
BACE	0.85_0	7	0	2
BACE	0.85_1	5	0	2
BACE	0.85_2	5	0	2
BACE	0.90_0	4	0	2
BACE	0.90_1	5	0	2
BACE	0.95_2	4	0	2

dataset	spectral_parameter	training_size	val_size	test_size
BBBP	0.00_0	1376	196	393
BBBP	0.00_1	1376	196	393
BBBP	0.00_2	1376	196	393
BBBP	0.05_0	656	93	188
BBBP	0.05_1	657	93	188
BBBP	0.05_2	646	92	185
BBBP	0.10_0	352	50	101
BBBP	0.10_1	343	48	98
BBBP	0.10_2	346	49	99
BBBP	0.15_0	219	31	63
BBBP	0.15_1	231	32	66
BBBP	0.15_2	222	31	64
BBBP	0.20_0	156	22	45
BBBP	0.20_1	160	22	46
BBBP	0.20_2	154	22	44
BBBP	0.25_0	116	16	33
BBBP	0.25_1	112	16	32
BBBP	0.25_2	112	15	32
BBBP	0.30_0	84	12	25
BBBP	0.30_1	93	13	27
BBBP	0.30_2	94	13	27
BBBP	0.35_0	76	10	22
BBBP	0.35_1	73	10	21
BBBP	0.35_2	74	10	21
BBBP	0.40_0	59	8	17
BBBP	0.40_1	57	8	17
BBBP	0.40_2	58	8	17
BBBP	0.45_0	49	7	14
BBBP	0.45_1	53	7	15
BBBP	0.45_2	50	7	15
BBBP	0.50_0	42	5	12
BBBP	0.50_1	38	5	11
BBBP	0.50_2	39	5	11
BBBP	0.55_0	35	4	10
BBBP	0.55_1	35	5	10
BBBP	0.55_2	33	4	10
BBBP	0.60_0	31	4	9
BBBP	0.60_1	32	4	10
BBBP	0.60_2	25	3	8
BBBP	0.65_0	21	3	6
BBBP	0.65_1	25	3	7
BBBP	0.65_2	24	3	7
BBBP	0.70_0	20	2	6
BBBP	0.70_1	21	2	6
BBBP	0.70_2	21	3	7
BBBP	0.75_0	18	2	5
BBBP	0.75_1	18	2	5
BBBP	0.75_2	18	2	6
BBBP	0.80_0	12	1	4
BBBP	0.80_1	14	1	4
BBBP	0.80_2	14	1	4
BBBP	0.85_0	9	1	3
BBBP	0.85_1	10	1	3
BBBP	0.85_2	8	1	3
BBBP	0.90_0	7	1	2
BBBP	0.90_1	7	1	2
BBBP	0.90_2	11	1	3
BBBP	0.95_0	7	1	3
BBBP	0.95_1	7	0	2
BBBP	0.95_2	7	1	2
BBBP	1.00_0	4	0	2

dataset	spectral_parameter	training_size	val_size	test_size
ClinTox	0.00_0	1008	144	288
ClinTox	0.00_1	1008	144	288
ClinTox	0.00_2	1008	144	288
ClinTox	0.05_0	530	75	152
ClinTox	0.05_1	522	74	150
ClinTox	0.05_2	511	73	147
ClinTox	0.10_0	290	41	83
ClinTox	0.10_1	297	42	85
ClinTox	0.10_2	280	39	80
ClinTox	0.15_0	188	26	54
ClinTox	0.15_1	196	28	56
ClinTox	0.15_2	192	27	55
ClinTox	0.20_0	140	20	41
ClinTox	0.20_1	133	19	38
ClinTox	0.20_2	141	20	41
ClinTox	0.25_0	97	13	28
ClinTox	0.25_1	112	15	32
ClinTox	0.25_2	106	15	31
ClinTox	0.30_0	82	11	24
ClinTox	0.30_1	88	12	25
ClinTox	0.30_2	84	12	24
ClinTox	0.35_0	70	9	20
ClinTox	0.35_1	67	9	19
ClinTox	0.35_2	70	9	20
ClinTox	0.40_0	54	7	16
ClinTox	0.40_1	60	8	17
ClinTox	0.40_2	56	7	16
ClinTox	0.45_0	50	7	15
ClinTox	0.45_1	48	6	14
ClinTox	0.45_2	45	6	13
ClinTox	0.50_0	41	5	12
ClinTox	0.50_1	42	6	12
ClinTox	0.50_2	42	5	12
ClinTox	0.55_0	35	4	10
ClinTox	0.55_1	33	4	10
ClinTox	0.55_2	35	5	10
ClinTox	0.60_0	32	4	9
ClinTox	0.60_1	28	3	8
ClinTox	0.60_2	31	4	9
ClinTox	0.65_0	25	3	7
ClinTox	0.65_1	26	3	8
ClinTox	0.65_2	25	3	7
ClinTox	0.70_0	21	2	6
ClinTox	0.70_1	21	3	6
ClinTox	0.70_2	21	3	7
ClinTox	0.75_0	18	2	5
ClinTox	0.75_1	19	2	6
ClinTox	0.75_2	18	2	6
ClinTox	0.80_0	15	2	5
ClinTox	0.80_1	16	2	5
ClinTox	0.80_2	14	2	5
ClinTox	0.85_0	14	2	4
ClinTox	0.85_1	13	1	4
ClinTox	0.85_2	14	2	4

dataset	spectral_parameter	training_size	val_size	test_size
ESOL	0.00_0	774	110	222
ESOL	0.00_1	774	110	222
ESOL	0.00_2	774	110	222
ESOL	0.05_0	449	64	129
ESOL	0.05_1	440	62	126
ESOL	0.05_2	447	63	128
ESOL	0.10_0	250	35	72
ESOL	0.10_1	264	37	76
ESOL	0.10_2	254	36	73
ESOL	0.15_0	165	23	48
ESOL	0.15_1	164	23	47
ESOL	0.15_2	173	24	50
ESOL	0.20_0	109	15	31
ESOL	0.20_1	118	16	34
ESOL	0.20_2	105	15	30
ESOL	0.25_0	83	11	24
ESOL	0.25_1	85	12	25
ESOL	0.25_2	79	11	23
ESOL	0.30_0	67	9	19
ESOL	0.30_1	64	9	19
ESOL	0.30_2	56	8	16
ESOL	0.35_0	52	7	15
ESOL	0.35_1	50	7	15
ESOL	0.35_2	49	6	14
ESOL	0.40_0	41	5	12
ESOL	0.40_1	39	5	11
ESOL	0.40_2	40	5	12
ESOL	0.45_0	28	4	9
ESOL	0.45_1	29	4	9
ESOL	0.45_2	25	3	8
ESOL	0.50_0	25	3	8
ESOL	0.50_1	24	3	7
ESOL	0.50_2	25	3	8
ESOL	0.55_0	19	2	6
ESOL	0.55_1	19	2	6
ESOL	0.55_2	23	3	7
ESOL	0.60_0	17	2	5
ESOL	0.60_1	18	2	5
ESOL	0.60_2	18	2	5
ESOL	0.65_0	12	1	4
ESOL	0.65_1	14	1	4
ESOL	0.65_2	14	2	4
ESOL	0.70_0	12	1	4
ESOL	0.70_1	11	1	4
ESOL	0.70_2	10	1	3
ESOL	0.75_0	11	1	3
ESOL	0.75_1	10	1	3
ESOL	0.75_2	7	1	3
ESOL	0.80_0	7	0	2
ESOL	0.80_1	9	1	3
ESOL	0.80_2	8	1	3
ESOL	0.85_0	6	0	2
ESOL	0.85_1	7	1	2
ESOL	0.85_2	7	0	2
ESOL	0.90_0	7	1	2
ESOL	0.90_1	6	0	2
ESOL	0.90_2	7	1	2
ESOL	0.95_0	6	0	2
ESOL	0.95_1	7	0	2
ESOL	0.95_2	7	1	3
ESOL	1.00_0	5	0	2
ESOL	1.00_1	4	0	2
ESOL	1.00_2	5	0	2

dataset	spectral_parameter	training_size	val_size	test_size
FreeSolv	0.00_0	448	64	129
FreeSolv	0.00_1	448	64	129
FreeSolv	0.00_2	448	64	129
FreeSolv	0.05_0	310	44	89
FreeSolv	0.05_1	292	41	84
FreeSolv	0.05_2	299	42	86
FreeSolv	0.10_0	194	27	56
FreeSolv	0.10_1	182	26	52
FreeSolv	0.10_2	190	27	55
FreeSolv	0.15_0	126	18	36
FreeSolv	0.15_1	121	17	35
FreeSolv	0.15_2	126	17	36
FreeSolv	0.20_0	91	12	26
FreeSolv	0.20_1	95	13	28
FreeSolv	0.20_2	98	14	29
FreeSolv	0.25_0	70	9	20
FreeSolv	0.25_1	74	10	21
FreeSolv	0.25_2	70	9	20
FreeSolv	0.30_0	47	6	14
FreeSolv	0.30_1	53	7	15
FreeSolv	0.30_2	47	6	14
FreeSolv	0.35_0	37	5	11
FreeSolv	0.35_1	34	4	10
FreeSolv	0.35_2	35	5	10
FreeSolv	0.40_0	35	5	10
FreeSolv	0.40_1	35	4	10
FreeSolv	0.40_2	25	3	8
FreeSolv	0.45_0	24	3	7
FreeSolv	0.45_1	24	3	7
FreeSolv	0.45_2	21	3	7
FreeSolv	0.50_0	21	3	6
FreeSolv	0.50_1	21	3	6
FreeSolv	0.50_2	21	3	6
FreeSolv	0.55_0	17	2	5
FreeSolv	0.55_1	18	2	6
FreeSolv	0.55_2	15	2	5
FreeSolv	0.60_0	14	2	4
FreeSolv	0.60_1	13	1	4
FreeSolv	0.60_2	14	2	5
FreeSolv	0.65_0	14	2	4
FreeSolv	0.65_2	12	1	4
FreeSolv	0.70_0	14	1	4
FreeSolv	0.70_1	12	1	4
FreeSolv	0.70_2	8	1	3
FreeSolv	0.75_0	10	1	3
FreeSolv	0.75_1	11	1	4
FreeSolv	0.75_2	10	1	3
FreeSolv	0.80_0	9	1	3
FreeSolv	0.80_1	11	1	3
FreeSolv	0.80_2	9	1	3
FreeSolv	0.85_0	8	1	3
FreeSolv	0.85_1	8	1	3
FreeSolv	0.85_2	10	1	3
FreeSolv	0.90_0	7	1	3
FreeSolv	0.90_1	8	1	3
FreeSolv	0.90_2	8	1	3
FreeSolv	0.95_0	7	0	2
FreeSolv	0.95_1	7	1	3
FreeSolv	0.95_2	8	1	3
FreeSolv	1.00_0	7	1	2
FreeSolv	1.00_1	7	0	2
FreeSolv	1.00_2	7	0	2

dataset	spectral_parameter	training_size	val_size	test_size
Lipophilicity	0.00_0	2940	420	840
Lipophilicity	0.00_1	2940	420	840
Lipophilicity	0.00_2	2940	420	840
Lipophilicity	0.05_0	980	140	280
Lipophilicity	0.05_1	994	142	285
Lipophilicity	0.05_2	986	140	282
Lipophilicity	0.10_0	473	67	136
Lipophilicity	0.10_1	496	70	142
Lipophilicity	0.10_2	486	69	139
Lipophilicity	0.15_0	292	41	84
Lipophilicity	0.15_1	284	40	82
Lipophilicity	0.15_2	284	40	81
Lipophilicity	0.20_0	199	28	57
Lipophilicity	0.20_1	203	28	58
Lipophilicity	0.20_2	189	27	55
Lipophilicity	0.25_0	126	18	36
Lipophilicity	0.25_1	140	20	40
Lipophilicity	0.25_2	143	20	41
Lipophilicity	0.30_0	109	15	32
Lipophilicity	0.30_1	110	15	32
Lipophilicity	0.30_2	103	14	30
Lipophilicity	0.35_0	78	11	23
Lipophilicity	0.35_1	87	12	25
Lipophilicity	0.35_2	78	11	23
Lipophilicity	0.40_0	61	8	18
Lipophilicity	0.40_1	60	8	17
Lipophilicity	0.40_2	66	9	19
Lipophilicity	0.45_0	55	7	16
Lipophilicity	0.45_1	49	7	14
Lipophilicity	0.45_2	46	6	14
Lipophilicity	0.50_0	37	5	11
Lipophilicity	0.50_1	37	5	11
Lipophilicity	0.50_2	42	5	12
Lipophilicity	0.55_0	32	4	9
Lipophilicity	0.55_1	39	5	11
Lipophilicity	0.55_2	30	4	9
Lipophilicity	0.60_0	26	3	8
Lipophilicity	0.60_1	25	3	8
Lipophilicity	0.60_2	25	3	8
Lipophilicity	0.65_0	22	3	7
Lipophilicity	0.65_1	21	2	6
Lipophilicity	0.65_2	21	3	6
Lipophilicity	0.70_0	14	2	5
Lipophilicity	0.70_1	18	2	5
Lipophilicity	0.70_2	18	2	6
Lipophilicity	0.75_0	14	2	4
Lipophilicity	0.75_1	14	1	4
Lipophilicity	0.75_2	14	2	5
Lipophilicity	0.80_0	10	1	3
Lipophilicity	0.80_1	12	1	4
Lipophilicity	0.80_2	11	1	3
Lipophilicity	0.85_0	7	1	3
Lipophilicity	0.85_1	8	1	3
Lipophilicity	0.85_2	10	1	3
Lipophilicity	0.90_0	7	0	2
Lipophilicity	0.90_1	7	1	2
Lipophilicity	0.90_2	8	1	3
Lipophilicity	0.95_1	5	0	2
Lipophilicity	0.95_2	6	0	2

dataset	spectral_parameter	training_size	val_size	test_size
SIDER	0.00_0	999	142	286
SIDER	0.00_1	999	142	286
SIDER	0.00_2	999	142	286
SIDER	0.05_0	525	75	150
SIDER	0.05_1	529	75	151
SIDER	0.05_2	535	76	153
SIDER	0.10_0	294	41	84
SIDER	0.10_1	294	42	84
SIDER	0.10_2	305	43	88
SIDER	0.15_0	207	29	59
SIDER	0.15_1	208	29	60
SIDER	0.15_2	203	29	58
SIDER	0.20_0	154	22	45
SIDER	0.20_1	153	21	44
SIDER	0.20_2	148	21	43
SIDER	0.25_0	114	16	33
SIDER	0.25_1	112	15	32
SIDER	0.25_2	118	16	34
SIDER	0.30_0	98	13	28
SIDER	0.30_1	98	14	29
SIDER	0.30_2	96	13	28
SIDER	0.35_0	81	11	24
SIDER	0.35_1	84	12	25
SIDER	0.35_2	85	12	25
SIDER	0.40_0	67	9	20
SIDER	0.40_1	70	10	21
SIDER	0.40_2	63	9	19
SIDER	0.45_0	60	8	18
SIDER	0.45_1	57	8	17
SIDER	0.45_2	60	8	17
SIDER	0.50_0	55	7	16
SIDER	0.50_1	57	8	17
SIDER	0.50_2	54	7	16
SIDER	0.55_0	45	6	13
SIDER	0.55_1	49	7	14
SIDER	0.55_2	46	6	14
SIDER	0.60_0	40	5	12
SIDER	0.60_1	44	6	13
SIDER	0.60_2	42	6	13
SIDER	0.65_0	36	5	11
SIDER	0.65_1	41	5	12
SIDER	0.65_2	42	6	12
SIDER	0.70_0	35	5	10
SIDER	0.70_1	34	4	10
SIDER	0.70_2	32	4	10
SIDER	0.75_0	33	4	10
SIDER	0.75_1	32	4	10
SIDER	0.75_2	32	4	10
SIDER	0.80_0	28	4	8
SIDER	0.80_1	30	4	9
SIDER	0.80_2	28	4	9
SIDER	0.85_0	27	3	8
SIDER	0.85_1	28	4	8
SIDER	0.85_2	27	3	8
SIDER	0.90_0	25	3	7
SIDER	0.90_1	25	3	7
SIDER	0.90_2	23	3	7
SIDER	0.95_0	20	2	6
SIDER	0.95_1	21	2	6
SIDER	0.95_2	17	2	5
SIDER	1.00_0	20	2	6
SIDER	1.00_1	19	2	6
SIDER	1.00_2	18	2	6

dataset	spectral_parameter	training_size	val_size	test_size
Tox21	0.00_0	5476	782	1565
Tox21	0.00_1	5476	782	1565
Tox21	0.00_2	5476	782	1565
Tox21	0.05_0	1630	232	466
Tox21	0.05_1	1677	239	479
Tox21	0.05_2	1880	235	471
Tox21	0.10_0	864	108	217
Tox21	0.10_1	883	110	221
Tox21	0.10_2	866	108	217
Tox21	0.15_0	504	63	126
Tox21	0.15_1	511	63	128
Tox21	0.15_2	517	64	130
Tox21	0.20_0	336	42	85
Tox21	0.20_1	348	43	87
Tox21	0.20_2	350	43	88
Tox21	0.25_0	254	31	64
Tox21	0.25_1	248	31	62
Tox21	0.25_2	242	30	61
Tox21	0.30_0	188	23	47
Tox21	0.30_1	182	22	46
Tox21	0.30_2	185	23	47
Tox21	0.35_0	150	18	38
Tox21	0.35_1	147	18	37
Tox21	0.35_2	153	19	39
Tox21	0.40_0	120	15	31
Tox21	0.40_1	114	14	29
Tox21	0.40_2	118	14	30
Tox21	0.45_0	96	12	24
Tox21	0.45_1	88	11	22
Tox21	0.45_2	98	12	25
Tox21	0.50_0	83	10	21
Tox21	0.50_1	77	9	20
Tox21	0.50_2	81	10	21
Tox21	0.55_0	65	8	17
Tox21	0.55_1	66	8	17
Tox21	0.55_2	67	8	17
Tox21	0.60_0	56	7	14
Tox21	0.60_1	55	6	14
Tox21	0.60_2	57	7	15
Tox21	0.65_0	51	6	13
Tox21	0.65_1	48	6	12
Tox21	0.65_2	49	6	13
Tox21	0.70_0	41	5	11
Tox21	0.70_1	39	4	10
Tox21	0.70_2	39	4	10
Tox21	0.75_0	37	4	10
Tox21	0.75_1	35	4	9
Tox21	0.75_2	36	4	10
Tox21	0.80_0	28	3	7
Tox21	0.80_1	32	4	8
Tox21	0.80_2	30	3	8
Tox21	0.85_0	27	3	7
Tox21	0.85_1	27	3	7
Tox21	0.85_2	24	3	6
Tox21	0.90_0	23	2	6
Tox21	0.90_1	23	2	6
Tox21	0.90_2	28	3	7
Tox21	0.95_0	21	2	6
Tox21	0.95_1	21	2	6
Tox21	0.95_2	25	3	7
Tox21	1.00_0	22	2	6
Tox21	1.00_1	22	2	6
Tox21	1.00_2	24	3	6