ASTRA: STATISTICALLY ROBUST MODEL SELECTION FROM CROSS-VALIDATION

Wojtek Treyde & Fernanda Duarte
Chemistry Research Laboratory
Department of Chemistry
University of Oxford
Oxford, OX1 3TA, UK
{wojtek.treyde,fernanda.duartegonzalez}@chem.ox.ac.uk

ABSTRACT

Current standard practices for comparing machine learning models in low-data regimes, common in materials discovery, lack statistical rigour. We present Automated model selection using Statistical Testing for Robust Algorithms (ASTRA), which combines model training using cross-validation (CV) with statistical hypothesis testing to identify significantly better performing models. Evaluating ASTRA on hundreds of synthetic data sets and real-life drug discovery data sets from the ASAP Discovery x OpenADMET challenge shows that it selects better models than choosing the model with the best mean or median CV score, in particular when CV scores do not correlate significantly with test performance. ASTRA will make it easier to develop new approaches that significantly outperform previous models, and its modular and customisable design allows users to seamlessly integrate it into existing machine learning workflows. ASTRA is freely available in a GitHub repository.

1 INTRODUCTION

Machine learning (ML) models are increasingly being used in all domains of science, as they can make predictions quickly, cheaply, and accurately. For many applications, such as the discovery of new drugs and materials, models need to be able to make accurate predictions beyond their training domain. Achieving this generalisability is particularly challenging in low-data regimes ($< 10,000$ data points).

In low-data regimes many standard practices, such as single train–test splits, become ill-suited for developing generalisable ML models, because it will rarely be possible to construct a sufficiently unbiased and diverse test set that can provide a good estimate of a model's generalisation error. Instead, cross-validation (CV) more robustly evaluates model generalisability, as models need to repeatedly generalise to the different parts of the input space covered by each data fold.

In CV, models are evaluated on more than one test set, and thus their performance is described by a distribution of scores rather than a single score. Simply comparing mean or median scores does not take this into account. Instead, statistical hypothesis tests consider the underlying score distribution, and can be used to check if the difference between performance distributions is significant or is explicable by random chance.

Recently, Wognum et al. (2024) highlighted the need for more rigorous comparison of ML models for drug discovery applications using statistical tests, and Ash et al. (2025) proposed a set of guidelines for which tests and performance metrics to use when comparing ML models. Thus, there is a growing appreciation for the importance of integrating statistical hypothesis testing into ML benchmarking (Kamuntavičius et al., 2025; Avdiunina et al., 2025; Fooladi et al., 2025; Fischer et al., 2025). Nevertheless, existing general (Feurer et al., 2015; Erickson et al., 2020; Hernandez et al., 2025; Jin et al., 2023) and chemistry-specific (Dalmau & Alegre-Requena, 2024; Mervin et al., 2024; Haghighatlari et al., 2020) automated ML tools have not integrated statistical hypothesis testing into the model selection process.

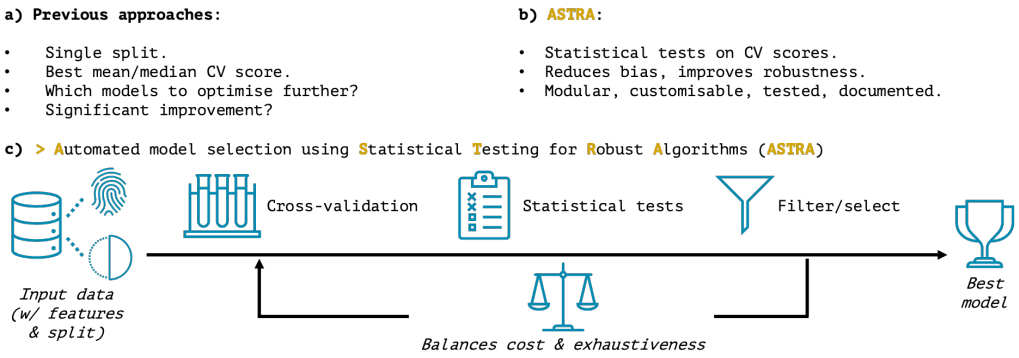

Figure 1: Overview of ASTRA. a) Previous approaches such as single train–test splits or selecting models based on best mean or median CV score are ill-suited for low-data regimes ($< 10,000$ data points) or overlook the distribution of CV scores. b) ASTRA improves upon this by integrating statistical hypothesis tests into the model selection workflow. c) High-level overview of the ASTRA workflow. Users provide featurised and split input data. Models are evaluated using CV. Filtering for optional hyperparameter tuning as well as final model selection are done using appropriate statistical tests.

To that end, we developed an automated model training and selection pipeline, termed *Automated model selection using Statistical Testing for Robust Algorithms* (ASTRA). ASTRA uses (nested) cross-validation, statistical testing, and efficient hyperparameter tuning to identify the best model architecture and featurisation scheme (*i.e.*, the model with the lowest expected generalisation error), while balancing computational cost and exhaustive exploration of the search space (Figure 1). The main innovation of ASTRA lies in using statistical testing to automate the model selection process, which improves the robustness of the final model by taking the distribution of model scores into account, and allows users to gauge whether a new model provides a statistically significant improvement. Hyperparameter tuning, if desired, is only performed for models that perform statistically equivalently, focussing computational resources.

We benchmarked ASTRA on hundreds of synthetic data sets and on the ASAP Discovery x OpenADMET Antiviral challenge (MacDermott-Opeskin et al., 2026), a blind, prospective, potency and absorption, distribution, metabolism, excretion, and toxicology (ADMET) property prediction task. Our results show that ASTRA selects better models than choosing based on the best mean or median CV score.

ASTRA will be broadly useful for ML practitioners working in low-data regimes ($< 10,000$ data points). It can be used to obtain a baseline model using domain-appropriate featurisation techniques, to benchmark more sophisticated ML approaches against that baseline, and to develop new models that significantly improve upon it. The code is tested, highly modular and customisable, it can be easily extended to include user-defined models and features, and is freely available under an MIT license in a GitHub repository.

2 Problem Statement

In many ML applications, especially in low-data regimes ($< 10,000$ data points), model selection is performed using CV. Given a finite set of candidate models $\mathcal{M} = \{M_1, \ldots, M_K\}$ and a performance metric, each model is evaluated across n CV folds, producing a set of scores

$$\mathcal{S}_k = \{s_{k,1}, \ldots, s_{k,n}\}, \quad k = 1, \ldots, K.$$

The central problem is: given these samples of performance measurements, how can one select the best model, *i.e.*, the one with the lowest expected generalisation error, while controlling for random variation?

A common practice is to select the model with the best mean (or median) CV score,

$$M^{\star}_{\text{mean}} = \arg\max_{M_k \in \mathcal{M}} \mathbb{E}[\mathcal{S}_k],$$

implicitly treating the mean CV score as a reliable estimator of downstream generalisation performance. However, this approach ignores the distributional properties of CV scores, which, in low-data settings, often exhibit high variance, skewness, and sensitivity to individual folds. As a result, differences in mean or median CV scores may arise from stochastic variation rather than from systematic differences in model generalisation behaviour.

We argue that by framing model selection as a sequence of statistical comparisons between performance distributions, one can reduce the sensitivity to outlier folds and identify models whose performance is not only high on average, but statistically superior to competing alternatives under the observed CV variability. The ASTRA framework proposed in this paper operationalises this principle.

3 Design Considerations

Several considerations and assumptions were made when developing the ASTRA workflow.

1. In low-data regimes ($< 10,000$ data points), it is usually not possible to construct an unbiased and diverse test set. Therefore, we opted for cross-validation rather than a single train–test split to more robustly evaluate model performance.
2. Users will know best how to meaningfully featurise and split their data, and many domain-specific tools exist for this. We thus start with the data already featurised and split into CV folds. For ease of integration with featurisation and splitting tools, ASTRA accepts the data set as a pandas (McKinney et al., 2010) data frame.
3. Compared to selecting models based on mean or median scores, statistical tests have the advantage of taking the distribution of scores into account and are thus more robust to outliers. Although performance estimates obtained from CV are not strictly independent because training sets overlap, this dependence can be sufficiently reduced when CV is done carefully (Ash et al., 2025). Using statistical testing, one can then automatically select the best performing model(s) based on p-values.
4. Hyperparameter tuning can have a large impact on model performance, but incurs significant computational cost because of the potentially large number of hyperparameter combinations to try. If used together with CV, it needs to be done in a nested fashion, which further drives up its computational cost. ASTRA strikes a balance between the benefit and the cost of hyperparameter tuning by selecting a subset of models that show strong baseline performance.
5. In low-data regimes, traditional ML models such as tree-based and kernel-based algorithms provide strong baseline performance and should be evaluated by default. However, users should be able to provide custom models, *e.g.*, deep learning (DL)-based. To facilitate integration with existing ML packages, ASTRA uses the widely accepted scikit-learn (Pedregosa et al., 2011) application programming interface (API) for all ML models.

4 Results and Discussion

4.1 ASTRA Workflow

Based on these considerations, we developed ASTRA as a Python package with a command line interface (CLI) with two entry points, `astra benchmark` and `astra compare` (Figure 2). However, all modules can also be imported to be used in existing Python workflows. A typical ASTRA workflow consists of running `astra benchmark` for a set of different features, resulting in one ML model per featurisation method that performs statistically significantly better than other models, followed by running `astra compare` to compare ML models using different featurisation methods, resulting in a single best-performing model, or an ensemble of statistically equivalently performing models.

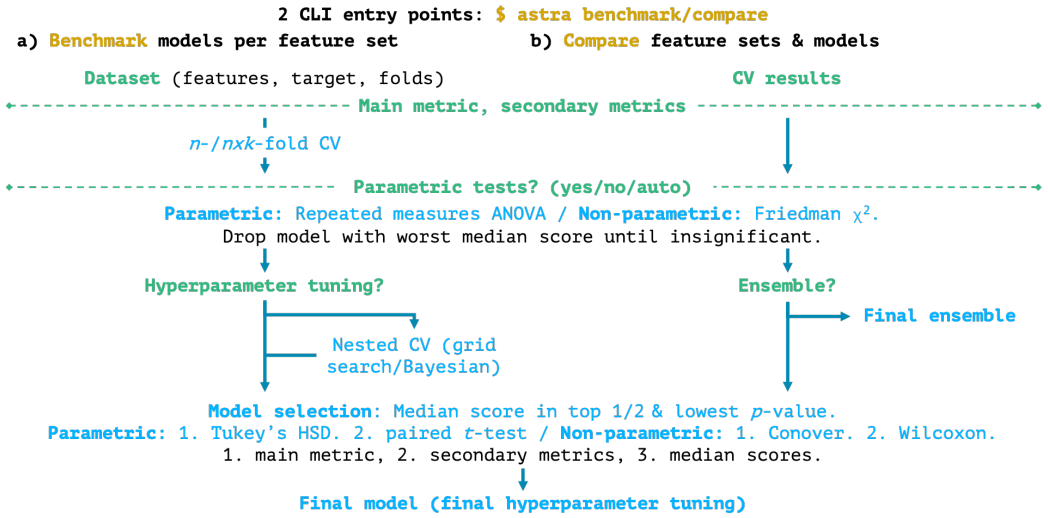

Figure 2: Details of the ASTRA workflow. ASTRA provides two command-line interface (CLI) entry points: a) `astra benchmark` and b) `astra compare`. User input is highlighted in green, ASTRA modules are blue.

For `astra benchmark`, users provide their pre-featurised and pre-split data set, a main metric to use for model comparison, and a set of secondary metrics to use if no significant differences are found using the main metric. In addition, users can specify whether to use parametric or non-parametric tests, or whether this should be automatically decided based on the distribution of scores, and whether to perform hyperparameter tuning using nested CV.

The `astra benchmark` workflow then begins by evaluating already implemented traditional ML models using n-fold or $(n \times k)$-fold cross-validation. Users can provide their own models as long as they conform to the scikit-learn API. Data processing techniques that must occur within the CV loop, such as data imputation, removal of constant or correlated features, and feature scaling, are integrated into ASTRA.

Next, to remove the worst performing models, ASTRA uses repeated measures analysis of variance (ANOVA) (Fisher, 1970), a parametric test, or the Friedman χ^2 test (Friedman, 1937), a non-parametric test, on the main metric CV scores. Both tests can be used to check for significant differences between multiple sets of model scores at the same time. ASTRA iteratively removes the model with the lowest median score until the performance differences become insignificant at a Bonferroni-corrected (Dunn, 1961) significance threshold of 0.05. If the user does not specify whether to use parametric or non-parametric tests, ASTRA will test for homogeneity of variances using Levene's test (Levene, 1960), for normality using the Shapiro-Wilk test (Shapiro & Wilk, 1965), and that the ratio of the largest and smallest variance is no larger than 9 (Blanca et al., 2018), and use parametric tests only if all of these conditions are met. If the user specified to use parametric tests but any of these conditions are violated, ASTRA will raise a warning but continue.

For the resulting models, ASTRA can, if desired, perform hyperparameter tuning and evaluate the optimised models using nested CV. In nested CV, all but one fold are used for hyperparameter optimisation, and the model is evaluated on the held-out fold, iterating across folds. Nested CV avoids choosing a model based on scores that were directly optimised during hyperparameter tuning, and which are therefore likely to be overly optimistic. The hyperparameter search can be performed via grid search or Bayesian optimisation (Akiba et al., 2019), using either default or user-defined search grids. Models are then again filtered using the ANOVA or Friedman χ^2 testing procedure described above.

Finally, (outer) CV scores are used to find the best-performing model. First, ASTRA uses Tukey's honestly significant difference (HSD) test (Tukey, 1949) or the Conover post-hoc test (Conover & Iman, 1979) with the more permissive Holm-Bonferroni adjustment (Holm, 1979) — depending on whether parametric or non-parametric tests are to be used — to identify significantly differently

performing models. Those models are sorted according to the number of models compared to which they perform significantly better. Starting from the model that has the highest number of significantly worse performing models, if its median score is in the top half, this model is selected. If multiple models perform significantly better than the same number of models, ASTRA chooses the model with the lowest sum of p-values.

If this does not yield a final model, the procedure is repeated using a paired t-test or the Wilcoxon signed-rank test (Conover, 1999), which, as non-post-hoc tests, can find significant differences where post-hoc tests do not. If that still does not yield a best model, ASTRA will repeat the procedure using the secondary metric(s), before choosing the model with the best median main score as the last resort. A final non-nested hyperparameter search is used to determine the final hyperparameters.

For `astra compare`, users provide CV results, for example as obtained by running `astra benchmark`, as well as main and secondary metrics, and whether to run parametric or non-parametric tests. The worst performing models are removed using the same ANOVA or Friedman χ^2 testing procedure as for `astra benchmark`, resulting in an ensemble of statistically equivalently performing models. If desired, a single best model will be determined using Tukey's HSD or Conover post-hoc tests, falling back to paired t or Wilcoxon signed-rank tests, or the best median score, using the main and then the secondary metric(s), analogously to `astra benchmark`. If any fallbacks are used (non-post-hoc tests, secondary metrics, or the best median score), ASTRA will issue a warning that users should consider whether the heuristic used is appropriate for their use case.

4.2 Benchmark on Synthetic Data Sets

To investigate whether the conceptual advantages of using statistical tests for ML model selection translate into practical advantages, we constructed 531 classification and 432 regression data sets of varying difficulty and size (Section 6.2), and used them to explore the central question: *Does ASTRA select a better model than choosing based on best mean or median CV score?* We ran `astra benchmark` on each data set using random 5-fold CV, and evaluated the ASTRA-selected model, as well as the models with the best mean and median scores on a withheld test set (Figure 3).

ASTRA largely chose the model with the best mean and median CV score (Figure 3a-b), although on classification data sets, it chose a better model than choosing based on mean or median CV score more often than it chose a worse one (41 vs. 38 experiments, and 62 vs. 43 experiments, respectively; Figure 3a). The area under the receiver operating characteristic curve (AUROC) of ASTRA-selected models was significantly higher than for models selected based on median CV score (Figure 3c; $p = 0.029$; one-sided Wilcoxon signed-rank test (Conover, 1999) with Pratt (Pratt, 1959) zero handling). Interestingly, for regression data sets, choosing based on mean or median CV score yielded a better model than ASTRA (33 vs. 22 experiments, and 24 vs. 13 experiments, respectively; Figure 3b), and the mean squared errors (MSEs) of models with the best median CV score were significantly lower than for ASTRA models ($p = 0.033$; Figure 3d). These results suggest that statistical hypothesis tests for model selection are particularly useful in classification settings.

4.3 Case Study on ASAP Discovery x OpenADMET Antiviral Challenge

To further investigate the practical utility of ASTRA, we deployed it on the ASAP Discovery x OpenADMET Antiviral challenge (MacDermott-Opeskin et al., 2026). The challenge consists of predicting the potency of several hundred compounds against Middle East respiratory syndrome coronavirus (MERS-CoV) and severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) main proteases (M^{pro}), as well as five ADMET endpoints. The data come from ASAP Discovery's M^{pro} drug discovery campaign. They were split by when they were tested, with compounds from early stages of the campaign serving as the training data, and compounds from later stages of the campaign serving as the test data. The test data were additionally stratified by chemical similarity, removing compounds that were deemed too chemically similar to training compounds.

This challenge therefore tests ML models retrospectively in real-life drug discovery settings, where ML models can be trained on early data to guide further drug optimisation efforts, and provides a

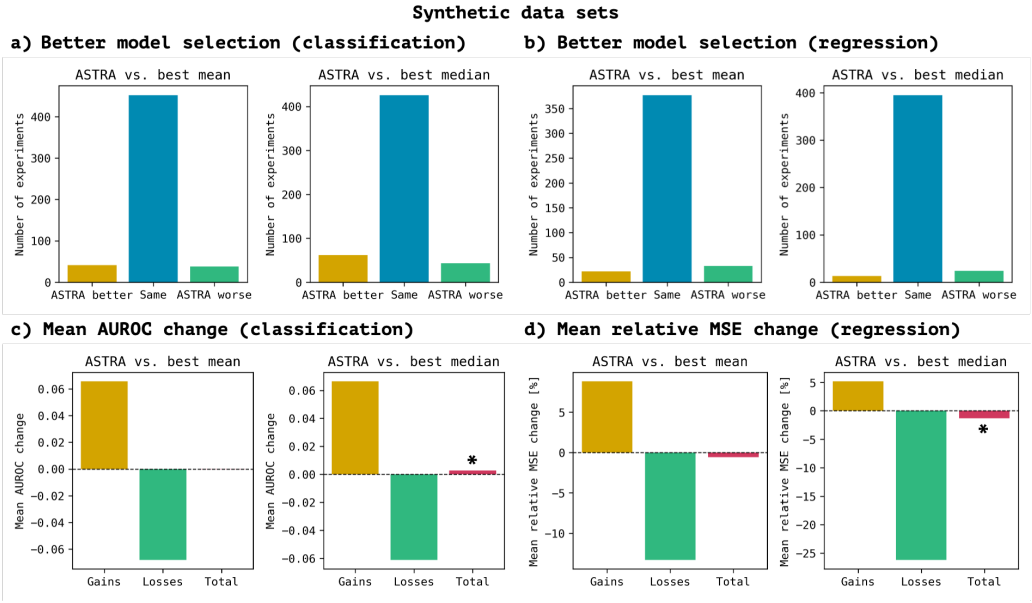

Figure 3: Benchmark results on synthetic data sets. a)-b) Number of experiments for which ASTRA chose a better (yellow), the same (blue), or worse (green) model than based on mean (left) or median (right) CV scores on a) synthetic classification data sets, and b) synthetic regression data sets. c) Mean AUROC change for classification experiments in which ASTRA chose a better (yellow) or worse (green) model than based on mean (left) or median (right) CV scores, and mean AUROC change across all classification experiments (red). d) Mean relative MSE change for regression experiments in which ASTRA chose a better (yellow) or worse (green) model than based on mean (left) or median (right) CV scores, and mean relative MSE change across all regression experiments (red). Asterisks in panels c)-d) indicate significant differences at a threshold of $p = 0.05$ (one-sided Wilcoxon signed-rank test with Pratt zero handling).

clean test set to evaluate the practical predictive power of ML models. Other common chemical benchmarks provide artificial train–test splits, *e.g.*, based on some measure of chemical similarity, and such test sets therefore only mimic real-life settings. In addition, predicting activity in drug discovery campaigns is a typical low-data scenario. This challenge thus provides an ideal test case for evaluating the practical utility of integrating statistical tests into ML model selection workflows in low-data regimes.

Because we expected the utility of ASTRA (and CV in general) to depend on the predictive power of the CV scores and, correspondingly, the data set split, we considered four splitting strategies — a random split, as well as splits based on Taylor-Butina (Butina, 1999), k-means, or Bemis-Murcko clusters (Bemis & Murcko, 1996). We featurised the data set using twelve standard cheminformatics fingerprints. We then went through a typical ASTRA workflow, consisting of running `astra benchmark` for every fingerprint and split, yielding a single model per fingerprint and split, and running `astra compare` to compare models obtained for different fingerprints on the same split, resulting in 12 (fingerprints) $\times$ 7 (data sets) $\times$ 4 (splits) $=$ 336 experiments for `astra benchmark` and 7 (data sets) $\times$ 4 (splits) $=$ 28 experiments for `astra compare`. ASTRA-selected models and the models with the best mean and median CV scores were retrained on the whole training set and evaluated on the time-split test set.

`astra benchmark` chose a better model than choosing based on best mean or median CV score in 54 and 86 experiments, respectively, 20% and 15% more often than it chose a worse model (45 and 75 experiments, respectively; Figure 4a). However, the MSEs of ASTRA models and models with the best mean or median CV score were not statistically significantly different (Figure 4c).

For `astra compare` there are much fewer experiments, and the results are therefore noisier. Nevertheless, ASTRA chose a better model than choosing based on best median CV score in 7 experi-

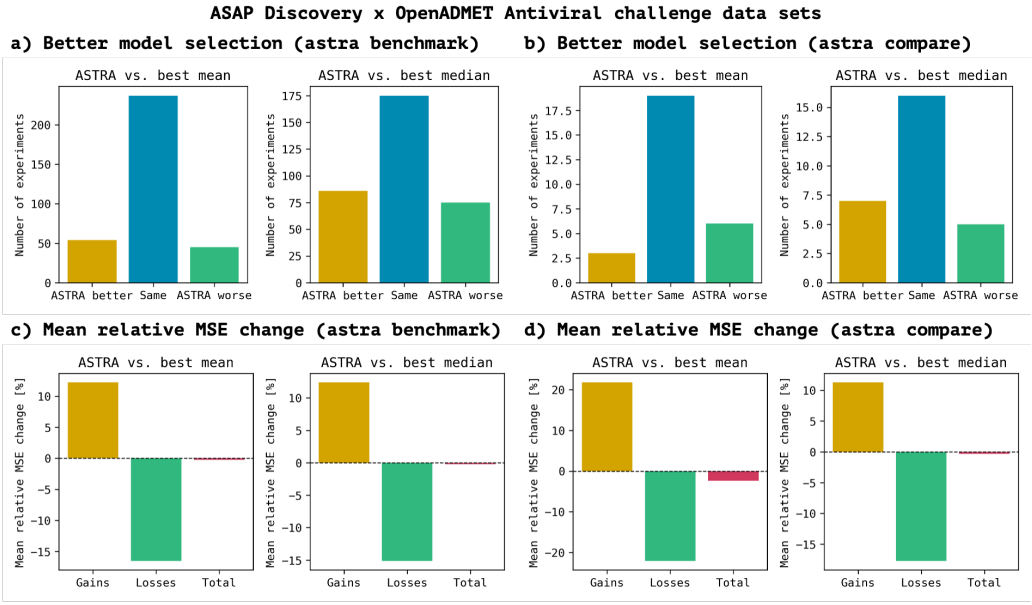

Figure 4: Results on data sets from the ASAP Discovery x OpenADMET Antiviral challenge. a)-b) Number of experiments for which ASTRA chose a better (yellow), the same (blue), or worse (green) model than based on mean (left) or median (right) CV scores for a) `astra benchmark` (across 12 different fingerpints and 4 different splits), and b) `astra compare` (across 4 different splits). c)-d) Mean relative MSE change for experiments in which c) `astra benchmark` and d) `astra compare` chose a better (yellow) or worse (green) model than based on mean (left) or median (right) CV scores, and mean relative MSE change across all experiments (red). Overall mean relative MSE changes are insignificant at a threshold of $p = 0.05$ (one-sided Wilcoxon signed-rank test with Pratt zero handling).

ments, and worse in only 5 experiments (Figure 4b). Choosing based on best mean CV score yielded better models than ASTRA in 6 experiments, minimally more often than ASTRA chose better (3 experiments). The MSEs of ASTRA models and models with the best mean or median CV score were not statistically significantly different (Figure 4d), possibly due to the small sample size.

To better understand when statistical tests select better models than mean or median scores, we separately considered the subsets of experiments for which the Spearman correlation between median CV and test scores was and was not significant at a threshold of $p = 0.05$ (Figure 5 and Figure S1).

Interestingly, for the subset of experiments for which the Spearman correlation between median CV and test scores was *insignificant*, `astra benchmark` chose better than based on mean or median scores in 38 and 61 experiments, respectively, 58% and 30% more often than it chose a worse model (24 and 47 experiments, respectively; Figure 5a), corresponding to a three- and two-fold improvement compared to when considering all experiments. Additionally, the MSEs of ASTRA-selected models were significantly lower than those of the models with the best mean CV score ($p = 0.038$; Figure 5c). For `astra compare`, results are qualitatively the same as for all experiments (Figure 5b and Figure 5d).

These results highlight the practical utility of ASTRA. It will not always be apparent which data set split will produce the most predictive CV scores, and users will therefore often encounter experiments where CV scores do not correlate well with downstream performance. This is particularly important for applications in which models need to generalise beyond their training domain. Because ASTRA outperforms model selection based on average CV scores particularly in cases where they do not correlate well with downstream performance, it is a more robust tool for model selection.

We note that ASTRA-selected model architectures and fingerprints were strongly dependent on the data set and splitting method (Figure S2), highlighting the utility of automating the model selection process.

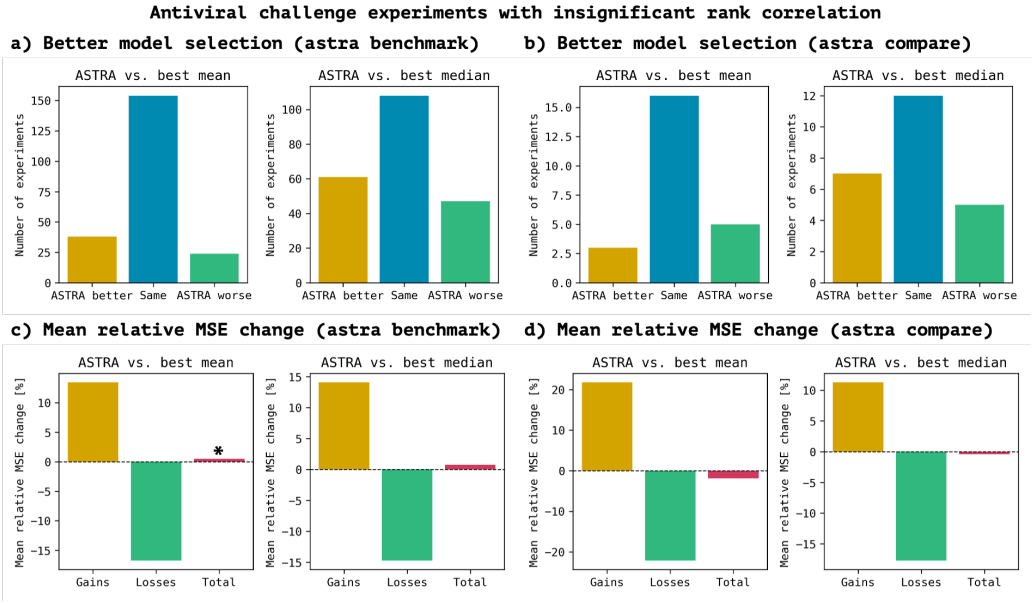

Figure 5: Results on data sets from the ASAP Discovery x OpenADMET Antiviral challenge for experiments for which the Spearman rank correlation between median CV scores and test scores was insignificant at a threshold of $p = 0.05$. a)-b) Number of experiments for which ASTRA chose a better (yellow), the same (blue), or worse (green) model than based on mean (left) or median (right) CV scores for a) `astra benchmark` (across 12 different fingerpints and 4 different splits), and b) `astra compare` (across 4 different splits). c)-d) Mean relative MSE change for experiments in which c) `astra benchmark` and d) `astra compare` chose a better (yellow) or worse (green) model than based on mean (left) or median (right) CV scores, and mean relative MSE change across all experiments (red). Asterisks in panels c)-d) indicate significant differences at a threshold of $p = 0.05$ (one-sided Wilcoxon signed-rank test with Pratt zero handling).

The combined results highlight the utility of ASTRA for choosing well-performing ML models from diverse featurisation schemes and model architectures.

5 Conclusion

ASTRA is a Python package and CLI for identifying the best ML model architecture and featurisation scheme using cross-validation and statistical testing, and will thus be broadly useful to ML practitioners working in low-data regimes ($< 10,000$ data points). Beyond the conceptual advantages of statistical tests for comparing CV scores, benchmarking on synthetic data sets and a case study on the ASAP Discovery x OpenADMET Antiviral challenge (MacDermott-Opeskin et al., 2026) show that ASTRA selects better models than choosing based on the best mean or median CV score in practice as well, in particular in classification settings and when CV scores do not correlate significantly with downstream performance.

Running ASTRA is practically no more expensive than running regular CV, as the only additional cost comes from running quick statistical tests. ASTRA implements many traditional models available in commonly used ML packages, and it allows custom models, further increasing its versatility. It can be used with domain-specific featurisation techniques to quickly obtain baseline models, and to then develop new approaches that significantly outperform them. Future computational prediction challenges should include an ASTRA baseline to evaluate whether more sophisticated models provide meaningful improvements.

ASTRA is facilitating the adoption of the guidelines on statistical testing set out by Ash et al. (2025) in ML model development workflows, and can grow into a bigger tool for the adoption of further best practices. As an open-source Python package ASTRA provides an ideal platform for incorporating

community feedback, and can easily be adjusted as the field evolves. Because of its modular nature, ASTRA can be used in conjunction with existing ML tools, fitting neatly into existing workflows.

6 METHODS

6.1 ASTRA IMPLEMENTATION DETAILS

ASTRA was implemented using scikit-learn (Pedregosa et al., 2011), Pingouin (Vallat, 2018), scikit-posthocs (Terpilowski, 2019), and SciPy (Virtanen et al., 2020). ASTRA includes all ML models implemented in scikit-learn (Pedregosa et al., 2011), XGBoost (Chen & Guestrin, 2016), LightGBM (Ke et al., 2017), and CatBoost (Prokhorenkova et al., 2018). For further implementation details, such as default model parameters, hyperparameter search grids, and practical instructions, the reader is asked to refer to ASTRA's GitHub repository, which includes extensive documentation.

6.2 SYNTHETIC DATA SETS

We constructed 531 classification and 432 regression data sets of varying size and difficulty using the `make_classification` and `make_regression` functions in scikit-learn (Pedregosa et al., 2011) (Table S1). For classification data sets, we varied the number of samples, the total number of features, the number of informative features, the number of redundant features, the number of duplicated features, and the proportions of samples assigned to the positive and negative classes. For regression data sets, we varied the number of samples, the total number of features, the number of informative features, the effective input rank, and the standard deviation of the Gaussian noise applied to the output. We randomly set aside 10% of the data for testing, and split the remaining 90% into 5 CV folds using random splitting. Classification data set splits were stratified using class labels.

For classification data sets, we trained all 22 scikit-learn (Pedregosa et al., 2011) (except for multinomial Naive Bayes), XGBoost (Chen & Guestrin, 2016), and LightGBM (Ke et al., 2017) classifiers and selected a model using `astra benchmark`. The area under the receiver operating characteristic curve (AUROC) was used as the main metric, and the area under the precision–recall curve (AUPRC) and Matthew's correlation coefficent (MCC) (Matthews, 1975) as secondary metrics.

For regression data sets, we trained all eight scikit-learn (Pedregosa et al., 2011), XGBoost (Chen & Guestrin, 2016), LightGBM (Ke et al., 2017), and CatBoost (Prokhorenkova et al., 2018) regressors and selected a model for every combination of fingerprint and split using `astra benchmark`. The mean squared error (MSE) was used as the main metric, and the coefficient of determination (R^2) and the mean absolute error (MAE) as secondary metrics.

Hyperparameter optimisation was performed using Optuna (Akiba et al., 2019). Otherwise, default settings were used. Final models were retrained on all available training data before evaluating on the test data. ASTRA-selected models were compared with the models with the best mean or median CV score based on test AUROCs and MSEs.

6.3 ASAP DISCOVERY X OPENADMET ANTIVIRAL CHALLENGE

The ASAP Discovery x OpenADMET Antiviral challenge provides time-split activity and ADMET data from ASAP Discovery's M^{pro} drug discovery campaign. The five ADMET endpoints are mouse liver microsomal stability (303 training data points, 122 test data points), human liver microsomal stability (301 training data points, 106 test data points), kinetic solubility (365 training data points, 112 test data points), the logarithm of the distribution coefficient (352 training data points, 126 test data points), and MDR1-MDCKII (a cell line) cell permeation (425 training data points, 126 test data points). Activity data is provided as negative log-transformed, half maximum inhibitory concentrations (pIC_{50}) against MERS-CoV M^{pro} (901 training data points, 297 test data points) and SARS-CoV-2 M^{pro} (842 training data points, 263 test data points).

We used `datamol` and `molfeat` to calculate a variety of fingerprints (`cats2d`, `pmapper`, `scaffoldkeys`, `desc2D`, `estate`, `rdkit`, `fcfp`, `ecfp`, `atompair`, `maccs`, `avalon`, and `topological`) (Gedeck et al., 2006; Kutlushina et al., 2018; Ertl, 2020; Kier & Hall, 1990; Rogers & Hahn, 2010; Carhart et al., 1985; Durant et al., 2002; Nilakantan et al., 1987) for all

molecules in the data set, covering a wide range of geometric, physicochemical, and pharmacophoric properties. All data sets were split into five CV folds randomly, as well as based on Taylor-Butina (Butina, 1999), k-means, and Bemis-Murcko clusters (Bemis & Murcko, 1996) using `useful_rdkit_utils` with five different random seeds, choosing the seed that led to the smallest standard deviation for the number of data points per fold. ADMET data, except LogD values, were log10-transformed.

We then trained all eight scikit-learn (Pedregosa et al., 2011), XGBoost (Chen & Guestrin, 2016), LightGBM (Ke et al., 2017), and CatBoost (Prokhorenkova et al., 2018) regressors and selected a model for every combination of fingerprint and split using `astra benchmark`. Hyperparameter optimisation was performed using Optuna (Akiba et al., 2019). We selected final combinations of fingerprints and models for every split using `astra compare`. For both `astra benchmark` and `astra compare`, the mean squared error (MSE) was used as the main metric, and the coefficient of determination (R^2) and the mean absolute error (MAE) as secondary metrics. Otherwise, default settings were used.

Final models were retrained on all available training data before evaluating on the time-split test data. ASTRA-selected models were compared with the models with the best mean or median CV score based on test MSEs.

ACKNOWLEDGEMENT

We thank Aleksy Kwiatkowski for beta testing ASTRA, and all developers of the open-source tools that ASTRA is built upon.

REPRODUCIBILITY STATEMENT

The source code for ASTRA and the benchmarks is available on GitHub.

REFERENCES

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2623–2631, 2019.

Jeremy R Ash, Cas Wognum, Raquel Rodríguez-Pérez, Matteo Aldeghi, Alan C Cheng, Djork-Arné Clevert, Ola Engkvist, Cheng Fang, Daniel J Price, Jacqueline M Hughes-Oliver, et al. Practically significant method comparison protocols for machine learning in small molecule drug discovery. *Journal of chemical information and modeling*, 65(18):9398–9411, 2025.

Polina Avdiunina, Shamieraah Jamal, Filipp Gusev, and Olexandr Isayev. All that glitters is not gold: Importance of rigorous evaluation of proteochemometric models. *Journal of Chemical Information and Modeling*, 65(19):10239–10252, 2025.

Guy W Bemis and Mark A Murcko. The properties of known drugs. 1. molecular frameworks. *Journal of medicinal chemistry*, 39(15):2887–2893, 1996.

María J Blanca, Rafael Alarcón, Jaume Arnau, Roser Bono, and Rebecca Bendayan. Effect of variance ratio on anova robustness: Might 1.5 be the limit? *Behavior Research Methods*, 50(3): 937–962, 2018.

Darko Butina. Unsupervised data base clustering based on daylight's fingerprint and tanimoto similarity: A fast and automated way to cluster small and large data sets. *Journal of Chemical Information and Computer Sciences*, 39(4):747–750, 1999.

Raymond E Carhart, Dennis H Smith, and RENGACHARI Venkataraghavan. Atom pairs as molecular features in structure-activity studies: definition and applications. *Journal of Chemical Information and Computer Sciences*, 25(2):64–73, 1985.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, 2016.

William Jay Conover. *Practical nonparametric statistics*, volume 350. John Wiley & Sons, 1999.

William Jay Conover and Ronald L Iman. On multiple-comparisons procedures. *Los Alamos Sci. Lab. Tech. Rep. LA-7677-MS*, 1:14, 1979.

David Dalmau and Juan V. Alegre-Requena. Robert: Bridging the gap between machine learning and chemistry. *WIREs Computational Molecular Science*, 14(5):e1733, 2024.

Olive Jean Dunn. Multiple comparisons among means. *J. Am. Stat. Assoc.*, 56(293):52–64, 1961.

Joseph L Durant, Burton A Leland, Douglas R Henry, and James G Nourse. Reoptimization of mdl keys for use in drug discovery. *Journal of chemical information and computer sciences*, 42(6): 1273–1280, 2002.

Nick Erickson, Jonas Mueller, Alexander Shirkov, Hang Zhang, Pedro Larroy, Mu Li, and Alexander Smola. Autogluon-tabular: Robust and accurate automl for structured data. *arXiv preprint arXiv:2003.06505*, 2020.

Peter Ertl. Identification of bioisosteric substituents by a deep neural network. *Journal of Chemical Information and Modeling*, 60(7):3369–3375, 2020.

Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. Efficient and robust automated machine learning. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

Yaëlle Fischer, Thibaud Southiratn, Dhoha Triki, and Ruel Cedeno. Deep learning vs classical methods in potency and adme prediction: Insights from a computational blind challenge. *Journal of Chemical Information and Modeling*, 2025.

Ronald Aylmer Fisher. Statistical methods for research workers. In *Breakthroughs in statistics: Methodology and distribution*, pp. 66–70. Springer, 1970.

Hosein Fooladi, Thi Ngoc Lan Vu, Miriam Mathea, and Johannes Kirchmair. Evaluating machine learning models for molecular property prediction: Performance and robustness on out-of-distribution data. *Journal of Chemical Information and Modeling*, 65(19):9871–9891, 2025.

Milton Friedman. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the american statistical association*, 32(200):675–701, 1937.

Peter Gedeck, Bernhard Rohde, and Christian Bartels. Qsar- how good is it in practice? comparison of descriptor sets on an unbiased cross section of corporate data sets. *Journal of chemical information and modeling*, 46(5):1924–1936, 2006.

Mojtaba Haghighatlari, Gaurav Vishwakarma, Doaa Altarawy, Ramachandran Subramanian, Bhargava U. Kota, Aditya Sonpal, Srirangaraj Setlur, and Johannes Hachmann. Chemml: A machine learning and informatics program package for the analysis, mining, and modeling of chemical and materials data. *WIREs Computational Molecular Science*, 10(4):e1458, 2020.

Jose Guadalupe Hernandez, Anil Kumar Saini, Attri Ghosh, and Jason H Moore. The tree-based pipeline optimization tool: Tackling biomedical research problems with genetic programming and automated machine learning. *Patterns*, 6(7), 2025.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scand. J. Stat.*, pp. 65–70, 1979.

Haifeng Jin, FranÃ§ois Chollet, Qingquan Song, and Xia Hu. Autokeras: An automl library for deep learning. *Journal of Machine Learning Research*, 24(6):1–6, 2023.

Gintautas Kamuntavičius, Tanya Paquet, Orestis Bastas, Dainius Šalkauskas, Alvaro Prat, Hisham Abdel Aty, Aurimas Pabrinkis, Povilas Norvaišas, and Roy Tal. Benchmarking ml in admet predictions: the practical impact of feature representations in ligand-based models. *Journal of Cheminformatics*, 17(1):108, 2025.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Adv. Neural Inf. Process. Syst.*, 30, 2017.

Lemont B Kier and Lowell H Hall. An electrotopological-state index for atoms in molecules. *Pharmaceutical research*, 7(8):801–807, 1990.

Alina Kutlushina, Aigul Khakimova, Timur Madzhidov, and Pavel Polishchuk. Ligand-based pharmacophore modeling using novel 3d pharmacophore signatures. *Molecules*, 23(12):3094, 2018.

Howard Levene. Robust tests for equality of variances. *Contributions to probability and statistics*, pp. 278–292, 1960.

Hugo MacDermott-Opeskin, Jenke Scheen, Cas Wognum, Joshua T Horton, Devany West, Alexander Matthew Payne, Maria A Castellanos, Sean Colby, Edward Griffen, David Cousins, et al. A computational community blind challenge on pan-coronavirus drug discovery data. *Journal of Chemical Information and Modeling*, 2026.

Brian W Matthews. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2):442–451, 1975.

Wes McKinney et al. Data structures for statistical computing in python. *scipy*, 445(1):51–56, 2010.

Lewis Mervin, Alexey Voronov, Mikhail Kabeshov, and Ola Engkvist. Qsartuna: An automated qsar modeling platform for molecular property prediction in drug design. *Journal of Chemical Information and Modeling*, 64(14):5365–5374, 2024.

Ramaswamy Nilakantan, Norman Bauman, J Scott Dixon, and R Venkataraghavan. Topological torsion: a new molecular descriptor for sar applications. comparison with other descriptors. *Journal of Chemical Information and Computer Sciences*, 27(2):82–85, 1987.

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

John W Pratt. Remarks on zeros and ties in the wilcoxon signed rank procedures. *Journal of the American Statistical Association*, 54(287):655–667, 1959.

Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. *Adv. Neural Inf. Process. Syst.*, 31, 2018.

David Rogers and Mathew Hahn. Extended-connectivity fingerprints. *Journal of chemical information and modeling*, 50(5):742–754, 2010.

Samuel Sanford Shapiro and Martin B Wilk. An analysis of variance test for normality (complete samples). *Biometrika*, 52(3-4):591–611, 1965.

Maksim A. Terpilowski. scikit-posthocs: Pairwise multiple comparison tests in python. *Journal of Open Source Software*, 4(36):1169, 2019.

John W Tukey. Comparing individual means in the analysis of variance. *Biometrics*, pp. 99–114, 1949.

Raphael Vallat. Pingouin: statistics in python. *Journal of Open Source Software*, 3(31):1026, 2018.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

Cas Wognum, Jeremy R Ash, Matteo Aldeghi, Raquel Rodríguez-Pérez, Cheng Fang, Alan C Cheng, Daniel J Price, Djork-Arné Clevert, Ola Engkvist, and W Patrick Walters. A call for an industry-led initiative to critically assess machine learning for real-world drug discovery. *Nature Machine Intelligence*, 6(10):1120–1121, 2024.

A APPENDIX

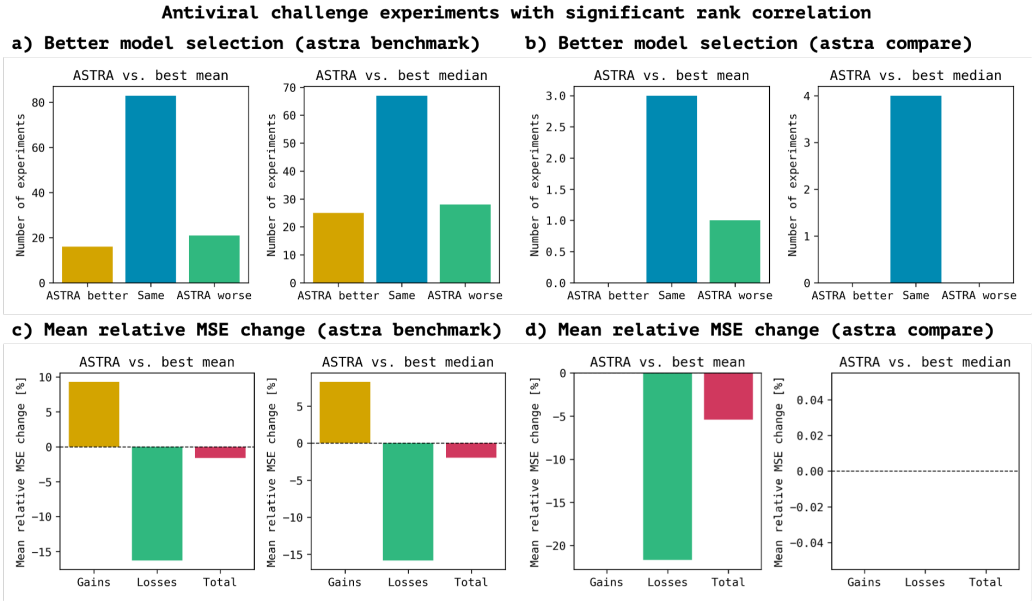

Figure S1: Results on data sets from the ASAP Discovery x OpenADMET Antiviral challenge for experiments for which the Spearman rank correlation between median CV scores and test scores was significant at a threshold of $p = 0.05$. a)-b) Number of experiments for which ASTRA chose a better (yellow), the same (blue), or worse (green) model than based on mean (left) or median (right) CV scores for a) `astra benchmark`, and b) `astra compare`. c)-d) Mean relative MSE change for experiments in which c) `astra benchmark` and d) `astra compare` chose a better (yellow) or worse (green) model than based on mean (left) or median (right) CV scores, and mean relative MSE change across all experiments (red). Overall mean relative MSE changes are insignificant at a threshold of $p = 0.05$ (one-sided Wilcoxon signed-rank test with Pratt zero handling).

Table S1: Parameters used for constructing synthetic classification and regression data sets.

Parameter	Values
Classification	
Number of samples, n_{samples}	$100,\ 1,000,\ 5,000,\ 10,000$
Total number of features, n_{features}	$0.1 \cdot n_{\text{samples}},\ 0.5 \cdot n_{\text{samples}},\ n_{\text{samples}}$
Number of informative features, $n_{\text{informative}}$	$0.25 \cdot n_{\text{features}},\ 0.5 \cdot n_{\text{features}},\ 0.75 \cdot n_{\text{features}}$
Number of redundant features, $n_{\text{redundant}}$	$0,\ 0.1 \cdot n_{\text{informative}},\ 0.2 \cdot n_{\text{informative}}$
Number of duplicated features	$0,\ 0.01 \cdot n_{\text{redundant}},\ 0.05 \cdot n_{\text{redundant}}$
Proportions of negative and positive classes	$0.5\ \&\ 0.5,\ 0.7\ \&\ 0.3,\ 0.9\ \&\ 0.1$
Regression	
Number of samples, n_{samples}	$100,\ 1,000,\ 5,000,\ 10,000$
Total number of features, n_{features}	$0.1 \cdot n_{\text{samples}},\ 0.5 \cdot n_{\text{samples}},\ n_{\text{samples}}$
Number of informative features, $n_{\text{informative}}$	$0.25 \cdot n_{\text{features}},\ 0.5 \cdot n_{\text{features}},\ 0.75 \cdot n_{\text{features}}, n_{\text{features}}$
Effective input rank	None, $0.5 \cdot n_{\text{features}},\ 0.75 \cdot n_{\text{features}}$
Standard deviation of Gaussian noise	$1,\ 5,\ 10$

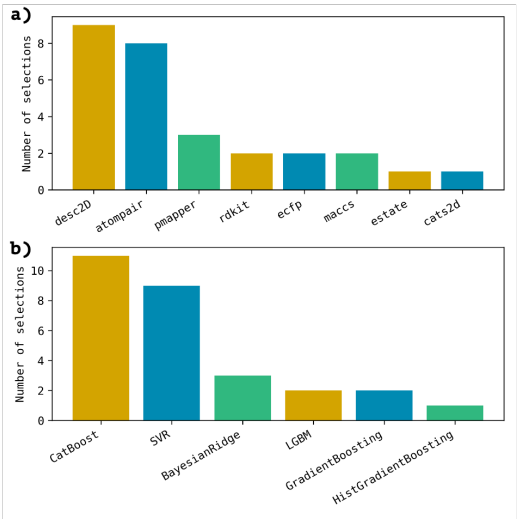

Figure S2: a) Fingerprints and b) algorithms selected by ASTRA after running `astra benchmark` and `astra compare` on every data set and split in the ASAP Discovery x OpenADMET challenge, resulting in a single model per data set–split combination.