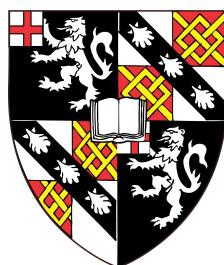


Using Machine Learning to Predict Protein/Ligand Interactions



Vikram Sundar

July 16, 2019

Department of Chemistry

Churchill College

University of Cambridge

Supervisor: Dr. Lucy Colwell

This dissertation is submitted for the degree of Master of Philosophy.

Contents

| | |
|---|-------------|
| Declaration | xi |
| Abstract | xiii |
| Acknowledgments | xv |
| 1 Introduction | 1 |
| 1.1 Drug Discovery and Protein/Ligand Binding | 1 |
| 1.2 Machine Learning for Virtual Screening | 2 |
| 1.3 Incorporating Multiple Proteins | 4 |
| 1.4 Generalisation and Attribution | 5 |
| 1.5 Outline of this Thesis | 7 |
| 2 Debiasing Algorithms and Generalisation | 9 |
| 2.1 Introduction | 9 |
| 2.2 Methods | 11 |
| 2.3 Results | 14 |
| 2.4 Discussion | 19 |
| Appendix: Supporting Information | 23 |
| 2.A Change in Far-AUC | 23 |
| 2.B Tracking Far-AUC during debiasing | 23 |
| 2.C AVE Bias and Dataset Size | 24 |
| 3 Generalisation and Attribution | 27 |
| 3.1 Introduction | 27 |
| 3.2 Methods | 29 |
| 3.2.1 Datasets | 29 |

Contents

| | |
|--|-----------|
| 3.2.2 Models | 30 |
| 3.2.3 Evaluation Methods | 31 |
| 3.3 Results | 33 |
| 3.3.1 Comparison of Model Performance | 33 |
| 3.3.2 Generalisation | 34 |
| 3.3.3 Attribution | 35 |
| 3.4 Conclusions | 51 |
| Appendix: Supporting Information | 53 |
| 3.A Supplementary Methods | 53 |
| 3.A.1 Toy Dataset Construction | 53 |
| 3.A.2 Random Matrix Theory Based Methods | 53 |
| 3.B Supplementary Results | 54 |
| 3.B.1 Additional Adversarial Examples | 54 |
| 3.B.2 Top Features Correlated with Activity for Other Datasets | 61 |
| 3.B.3 Toy Model AUC as a Function of Distance | 61 |
| 4 DTI Models and Generalisation to Unseen Proteins | 69 |
| 4.1 Introduction | 69 |
| 4.2 Methods | 71 |
| 4.2.1 Dataset Construction and Problem Set-up | 71 |
| 4.2.2 Models | 73 |
| 4.3 Results | 74 |
| 4.3.1 On the Unseen Target and Drug Subproblem | 74 |
| 4.3.2 On the Other Subproblems | 76 |
| 4.3.3 Proposed Explanations | 77 |
| 4.4 Conclusions | 79 |
| Appendix: Supporting Information | 81 |
| 4.A Supplementary Methods | 81 |
| 4.A.1 Dataset Statistics | 81 |
| 4.A.2 Model Descriptions | 82 |
| 4.A.3 Hyperparameter Selection | 84 |

Contents

| | |
|---|------------|
| 4.B Supplementary Results | 85 |
| 4.B.1 Additional Datasets | 85 |
| 4.B.2 Trial-by-Trial Comparisons | 85 |
| 4.B.3 Failure to Eliminate Ligand Similarity | 90 |
| 4.B.4 Probabilities Predicted in Training Submatrix | 93 |
| 5 Conclusion | 99 |
| Bibliography | 101 |

List of Figures

| | |
|--|----|
| 1.2.1 Generating ECFP Fingerprints | 3 |
| 2.1.1 Definition of AVE and MUV Bias | 10 |
| 2.2.1 The Far-AUC and Generalisation | 12 |
| 2.2.2 Impact of Debiasing on the Far-AUC | 14 |
| 2.3.1 Change in Far-AUC in Relation to Other Factors | 16 |
| 2.3.2 Standard AUC and Far-AUC After Debiasing | 17 |
| 2.3.3 AVE Debiasing without Deletion | 18 |
| 2.A.1 Change in Far-AUC and Decoys Added | 23 |
| 2.B.1 MUV Bias and Far-AUC | 24 |
| 2.B.2 AVE Bias and Far-AUC | 24 |
| 2.C.1 AVE Bias and Dataset Size | 25 |
| 2.C.2 Debiasing without Deletion and Number of Active Ligands | 25 |
| 3.2.1 Evaluation Methods | 32 |
| 3.3.1 Comparison of Validation AUCs for Various ML Models. | 36 |
| 3.3.2 Far-AUC and Standard AUC for Various ML Models | 37 |
| 3.3.3 Comparison of Far-AUCs for Various ML Models | 38 |
| 3.3.4 Attribution Scores on Toy Datasets | 39 |
| 3.3.5 Attribution False Positives and False Negatives | 40 |
| 3.3.6 Adversarial Examples | 41 |
| 3.3.7 Top Features Correlated with Activity | 43 |
| 3.3.8 Top Features Correlated with Activity with Fragment-Matched Decoys | 44 |
| 3.3.9 Attribution Scores After Adding Fragment-Matched Decoys | 45 |
| 3.3.10 Adversarial Examples with Fragment-Matched Decoys | 46 |
| 3.3.11 Background and Toy Dataset Correlation Matrices | 48 |

List of Figures

| | |
|--|----|
| 3.3.12 Comparative Attribution Scores on Toy and Real Data | 50 |
| 3.B.1 Adversarial Examples for ThreeElem0 | 55 |
| 3.B.2 Adversarial Examples for ThreeElem2 | 56 |
| 3.B.3 Adversarial Examples for ThreeElem3 | 57 |
| 3.B.4 Adversarial Examples for ThreeElem0 with Fragment-Matched Decoys | 58 |
| 3.B.5 Adversarial Examples for ThreeElem2 with Fragment-Matched Decoys | 59 |
| 3.B.6 Adversarial Examples for ThreeElem3 with Fragment-Matched Decoys | 60 |
| 3.B.7 Top Features Correlated with Activity for ThreeElem0 | 62 |
| 3.B.8 Top Features Correlated with Activity for ThreeElem0 with Fragment-Matched Decoys | 63 |
| 3.B.9 Top Features Correlated with Activity for ThreeElem2 | 64 |
| 3.B.10 Top Features Correlated with Activity for ThreeElem2 with Fragment-Matched Decoys | 65 |
| 3.B.11 Top Features Correlated with Activity for ThreeElem3 | 66 |
| 3.B.12 Top Features Correlated with Activity for ThreeElem3 with Fragment-Matched Decoys | 67 |
| 3.B.13 AUC as a Function of Distance from Training Set for Toy Data | 68 |
| 4.2.1 Setup for the DTI Problem and Our Proposal | 72 |
| 4.3.1 Impact of Logistic Regression Preprocessing on Unseen Target and Drug AUCs | 75 |
| 4.3.2 Unseen Target and Drug AUCs for Logistic Regression and Random Forest Preprocessing | 75 |
| 4.3.3 Impact of Logistic Regression Preprocessing on Unseen Drug AUCs . . | 76 |
| 4.3.4 Impact of Logistic Regression Preprocessing on Unseen Target AUCs . | 77 |
| 4.3.5 Information Added by Our Method for DTI Models | 79 |
| 4.B.1 Impact of Logistic Regression Preprocessing on Unseen Target and Drug AUCs (All Data) | 86 |
| 4.B.2 Unseen Target and Drug AUCs for Logistic Regression and Random Forest Preprocessing (All Data) | 87 |
| 4.B.3 Impact of Logistic Regression Preprocessing on Unseen Drug AUCs (All Data) | 88 |

List of Figures

| | | |
|--------|--|----|
| 4.B.4 | Impact of Logistic Regression Preprocessing on Unseen Target AUCs (All Data) | 89 |
| 4.B.5 | Trial-by-Trial Impact of Logistic Regression Preprocessing on Unseen Drug and Target AUCs | 90 |
| 4.B.6 | Trial-by-Trial Impact of Logistic Regression Preprocessing on Unseen Target AUCs | 91 |
| 4.B.7 | Trial-by-Trial Impact of Logistic Regression Preprocessing on Unseen Drug AUCs | 92 |
| 4.B.8 | Trial-by-Trial Impact of Random Forest Preprocessing on Unseen Drug and Target AUCs | 93 |
| 4.B.9 | Trial-by-Trial Impact of Random Forest Preprocessing on Unseen Target AUCs | 94 |
| 4.B.10 | Trial-by-Trial Impact of Random Forest Preprocessing on Unseen Drug AUCs | 95 |
| 4.B.11 | Failure to Eliminate Ligand Similarity from RLS | 96 |
| 4.B.12 | Information Added by Preprocessing for DTI Models (All Data) | 97 |

Declaration

This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration except where specifically indicated in the text. This dissertation is under 15000 words in length, including summary/abstract, tables and footnotes, but excluding table of contents, photographs, diagrams, figure captions, list of figures/diagrams, list of abbreviations/acronyms, bibliography, appendices, and acknowledgements.

Abstract

Identification of ligands that tightly bind to given protein targets is a crucial step in drug discovery. Traditional high-throughput screening approaches to this problem are expensive and prone to failure while physics-based computational approaches like docking and molecular dynamics have proven inaccurate. Many standard machine learning (ML) approaches have achieved outstanding success on virtual screening (identifying active ligands for a given protein) on benchmark datasets. This thesis has three parts related to problems arising with ML algorithms in virtual screening. In the first part, we demonstrate that many common ML models fail to generalise well and that debiasing algorithms, a common means of improving generalisability by eliminating bias in chemical data, fail to consistently improve generalisation. In the second part, we develop methods of attribution for ML virtual screening models and show that many common ML models fail to learn the right binding logic on toy data; we trace their attribution errors to spurious correlations in background data that confound the models. In the third part, we develop models for the multiple protein/multiple ligand problem that can generalise to previously unseen proteins by using a two-step process involving preprocessing with robust single protein models. Our work points to common flaws in standard ML approaches to virtual screening while proposing several potential solutions.

Acknowledgments

I would like to thank a number of people that made this thesis possible. First, I thank my mentor and supervisor, Dr. Lucy Colwell, for her support and guidance throughout this project. She provided me with an introduction to the field, inspired several of the ideas that became key in the project, and helped me overcome the many challenges I encountered. My research this year would not have been possible without her. I would also like to thank Dr. Carl Poelking and research scientists at Astex Pharmaceuticals for helpful discussions that shaped some of my research.

My year at Cambridge would not have been possible without the financial support of the Winston Churchill Foundation of the United States. Thank you for granting me the Churchill Scholarship; I hope to have lived up to the prestige of the award you provided me with. Thank you also to Churchill College and all the staff there for their support throughout the research process this year. Thank you to the wonderful community of students in Churchill College who made this year one of the most enjoyable and most productive years of my life. Most importantly, thank you to my parents who have always supported me, no matter how far from home I am.

Chapter 1

Introduction

1.1 Drug Discovery and Protein/Ligand Binding

The first step in drug discovery today is identification of a ligand that binds tightly to a given protein target, carrying out a desired function [27, 30]. It is therefore crucial that pharmaceutical scientists possess efficient and reliable methods of identifying ligands that tightly bind to a given protein target. Unfortunately, many methods commonly available today have serious limitations.

The predominant experimental approach is high-throughput screening, which is guided trial-and-error using a screening library of millions of compounds [30]. However, chemical space, the set of all possible chemical compounds, consists of orders of magnitude more compounds. Screening experiments are prone to failure because they cannot explore a large enough fraction of chemical space to identify the best ligand for a given protein [60, 45, 50]. Computational approaches would ideally explore regions of chemical space that screening experiments could not explore for maximum utility.

A variety of computational approaches to this problem have been developed. There are 3 distinct computational problems that could be considered: virtual screening, which predicts whether or not a given ligand is active, i.e. tightly bound to the protein in question, affinity prediction, which predicts binding affinity, and pose prediction, which predicts the structure of the bound protein-ligand complex [13]. This thesis focuses on the virtual screening problem. We may also consider two broad classes of approaches: physical models or statistical models [13].

Physical models for virtual screening have relied on either docking, where both the

protein and ligand are treated as rigid and fitted together using some scoring function, or Molecular Dynamics, where the trajectories of the ligand and protein interacting are simulated to predict activity [27, 13]. Both approaches require a scoring function or force field to estimate the energy of a given protein-ligand interaction, since an exact computation is not computationally feasible today [27, 10, 13]. Inaccuracies in the force field or other physical model and further coarse-graining for speed optimization have hindered the overall utility of these methods [27, 10, 4, 13]. Often, these approaches fail to predict many known interactions; when they are successful, the force fields need to be hand-tuned to the given system and are largely not easily generalisable [13].

The emergence of large quantities of screening data in databases such as ChEMBL [21] has encouraged the use of statistical models for this problem instead. Machine learning techniques have been introduced and applied to the virtual screening problem with qualified success. In this thesis, we focus on identifying and rectifying some of the problems with common ML approaches to virtual screening.

1.2 Machine Learning for Virtual Screening

Machine learning (ML) treats the virtual screening problem as a supervised classification problem. Here, we are given data about a training set of ligands with known activity and are asked to make predictions about a test set. A given model extracts a feature vector for each ligand that describes its structure; it then fits a function on a given high-dimensional vector space that predicts activity of a ligand. We test a model's accuracy by measuring its performance on the test set [13]. Models are expected to output probabilities that a given molecule is active; a chemist can then set a threshold to split the predicted molecules into actives and inactives. The standard measure of performance used throughout this work is the Area under the Receiver Operator Characteristic Curve (AUC); the AUC does not consider choice of threshold. An AUC of 0.5 corresponds to random predictions and of 1 to perfect accuracy.

There are two key choices to be made in this process: the choice of molecular representation, or how to convert a ligand structure into a feature vector, and the choice of model. A large number of molecular representations have been developed [46, 8]. The simplest possible representation is to count the number of heavy atoms, hydrogen

bond donors and acceptors, ring systems and other important features [59, 72]. More sophisticated features called chemical fingerprints also include information about the chemical structure of the ligand; examples of fingerprints include MACCS keys [16] and ECFP fingerprints [58]. A recent, more powerful approach is the graph convolutional neural network, which takes as input the chemical structure as a combinatorial graph and learns the appropriate molecular representation alongside the activity function [17, 35, 2, 23]. It is still unclear whether the increased complexity of these descriptions consistently leads to improved performance [72, 47].

This work focuses exclusively on ECFP6 fingerprints as a feature description for ligands. These fingerprints are generated by a three-step process. First, circles of up to a given diameter, in this case 6, are drawn around each atom to produce a list of substructures of the given molecule. Duplicate substructures are eliminated and each substructure is assigned a numerical identifier that encodes its connectivity and atom types. Lastly, each identifier is hashed to a fixed-length binary representation via a given hash function [58]. A summary of the first two steps of this process may be found in Fig. 1.2.1. This process provides a representation that accounts for atom type and connectivity, but could potentially have a high rate of bit collision [28].

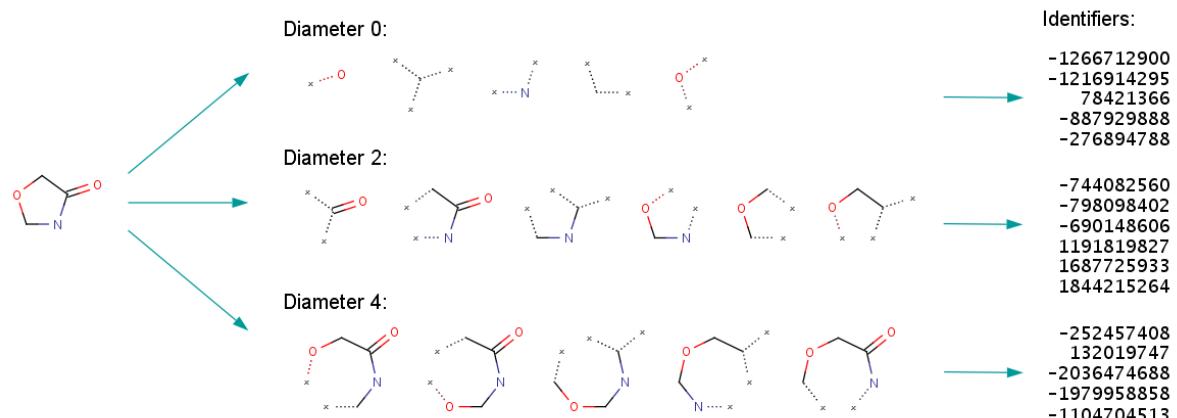


Figure 1.2.1: Generating ECFP Fingerprints, taken from the ChemAxon documentation [58]. To generate these fingerprints, we first generate substructures of up to a given radius and then compute numerical identifiers for each substructure. Finally (not shown), we hash these identifiers into a fixed-length bit string.

An equally wide variety of models have been applied to the virtual screening problem. Traditional approaches include support vector machines, Gaussian processes,

and random forests [13, 65, 77, 3, 4]. The relative simplicity and interpretability of these approaches provides some guarantee to the user that they will not drastically overfit. Over the past few years, there has been an explosion of interest in the use of deep neural networks for virtual screening [13, 44, 17, 35, 2, 23, 66, 71, 55]. Deep neural networks have the advantage of being highly expressive, i.e. able to fit essentially any function, but at the cost of decreased interpretability and much greater likelihood of overfitting. This work primarily focuses on simpler models, but many of the issues we identify are also applicable to deep neural networks [47, 72]. Many of these models, whether more traditional or deep, have performed outstandingly well on validation sets, with reported AUCs exceeding 0.9 [66, 54, 17, 71, 35, 53, 55, 24].

1.3 Incorporating Multiple Proteins

All of the methods outlined thus far rely on the existence of screening data about the protein of interest, and therefore do not obviate the need for a prior successful experimental screen. The problem of predicting ligands that tightly bind to a protein without prior screening data is much more difficult [19, 57]. The most common approach to this problem is to expand the training set to include information about multiple proteins, for some of which we have available screening data. This problem is now known as either proteochemometric modeling (PCM) or drug-target interactions (DTI) [19, 57, 52, 14].

Creating a DTI model requires a representation of both the protein and the ligand in question. If structural data is provided, a wide variety of structural representations can be used. For families of closely related proteins, images of the 3D protein structure or of the binding site can be directly encoded; if crystal structures of the bound protein-ligand complex are available, those can also be used in a similar fashion [57, 38]. Without structural data, the protein sequence can be translated either into a simple amino acid or dipeptide frequency count [19, 57, 56] or physico-chemical descriptors can be used for individual amino acids [57, 68]. Another popular approach is the use of sequence or structural similarity in lieu of explicit vector representations [19, 57, 63]; these methods require kernel-based models.

Unlike the situation for virtual screening, many DTI models were designed specif-

ically for the DTI problem, instead of being specific cases of more general models. This is partially due to the popularity of similarity-based representations of the target protein. One of the most popular model classes today is based on matrix factorization, more commonly used in recommender systems [19]. In this approach, the matrix of protein/ligand interactions is factorized as a product of two low-rank matrices with constraints from protein and ligand similarities [25, 12, 76, 20, 19]. A number of specialized loss functions and models have been developed along this general principle. However, many of these models show poor performance, especially on issues of generalising to completely unknown proteins and ligands [19]. The specifics of these approaches and some other popular ones will be discussed in Chapter 4.

Just as with virtual screening, deep neural networks have also emerged as a popular approach in this field [73, 31, 42, 22, 44, 55]. A variety of architectures for the deep neural networks have been proposed, including the standard feedforward neural network [55], the restricted Boltzmann machine [73], standard convolutional neural networks [71], and the graph convolutional neural network [35]. These neural networks also can take in more complex representations of the target protein, which may include the complete structure or the structure of the binding pocket, and complete structural information about the ligand. The greater expressivity of the deep neural network works well with the larger quantity of data available in the DTI problem. Deep neural networks have often shown higher performance due to these advantages [57], but they still do not perform well enough to overcome the generalisability problem.

1.4 Generalisation and Attribution

One important problem in the field today is generalisation across all of chemical space. Many papers have reported a discrepancy between high performance on validation sets and poor performance in generalising across all of chemical space [70, 11, 59, 61, 72, 43, 64]. One of the primary contributing factors to this discrepancy is dataset bias. Molecules in chemical space tend to be clustered around scaffolds, since similar molecules are easier for chemists to synthesize; this issue is known as analogue bias [70, 59, 72]. Complicating these clustering issues is the fact that active ligands for a given protein are likely to be similar and therefore clustered together even with

Chapter 1 Introduction

perfectly unbiased data. In addition, if, due to inadequate sampling, a simple feature like number of heavy atoms correlates strongly with activity, an ML model can perform very well while learning a completely useless rule; this is the problem of artificial enrichment [59, 70]. It is therefore possible for a model to perform very well on a not carefully designed validation set by simply memorising information about a particular cluster or scaffold and be unable to generalise to any other scaffold [59, 72]. Generalisation is both crucial for model performance *in silico* and most useful for experimentalists, who would like to make predictions about ligands dissimilar to ones they have produced in the laboratory [50].

Related to issues of generalisation is the problem of attribution and interpretability of virtual screening models. Attribution has emerged as an increasingly important problem in ML in general, both to make ML models more trustworthy for users and to prevent the generation of adversarial examples that may harm model performance [69, 37]. For chemistry in particular, interpretable models can help guide medicinal chemists in the drug discovery process in making refinements to active ligands to improve other properties [41]. Ideally, attribution could be used for *de novo* drug design, wherein models identify fragments predicted to be important for binding and construct a molecule that is optimally active for a given protein target. However, accurate attribution is difficult for virtual screening models. The hashing step in the fingerprint process means that a given feature does not necessarily correspond to only one molecular fragment [58], making standard attribution methods largely useless. Further, even if attribution is performed successfully, high-performing models are often found to have learned the wrong binding logic [47].

Generalising in protein space poses yet another challenge. As described above, the DTI models today do not perform well even on randomly split validation sets. Our current methods of representing protein targets and training models for them fail to consistently make predictions in protein space. These results do not account for clustering in chemical space as described previously or for phylogenetic clustering in protein space. It is crucial that computational scientists tackle this problem so our models can be truly useful to experimentalists; a model that requires prior identification of active ligands can only be of limited utility.

1.5 Outline of this Thesis

The main results of this thesis consist of three distinct chapters, each of which is an independent manuscript either under review or being prepared for publication. All three chapters are linked by the overall theme of addressing the problems discussed in the previous section.

Chapter 2 focuses on the generalisation problem in chemical space. We define a new metric for generalisation ability of a given model and demonstrate that standard ML virtual screening models fail to generalise well. We then examine debiasing algorithms, algorithms designed to reduce clustering and hopefully improve generalisation while also providing a better metric for measuring generalisation ability. We demonstrate that surprisingly, common debiasing algorithms fail to improve generalisation of our models and do not do a particularly good job of measuring their generalisation ability. Our work suggests that debiasing algorithms should be used more carefully and other approaches may be preferred for resolving the generalisation issue. This chapter has been independently published on ChemRXiv and is currently under review [64].

Chapter 3 examines the generalisation and attribution questions more closely. We examine the performance of random-matrix-theory (RMT) methods that claim to eliminate noise and generalise better with more accurate attributions when compared to standard models; we find that they do not live up to their promises. Further, we develop a method of attribution based on toy datasets that can be used on any fingerprint-based model and use it to show serious flaws in standard virtual screening models. We identify the source of these flaws as spurious correlations in the background screening library, suggesting that the diversity of the screening library plays an important role in model attribution.

Chapter 4 looks at the DTI problem and proposes a new framework for developing DTI models to improve generalisation in protein space. We show that using robust single protein virtual screening models can significantly improve the performance of DTI models. In particular, our new models are able to generalise in protein and ligand space simultaneously, consistently attaining AUCs greater than 0.9 on our test sets. While issues related to clustering remain, our results suggest a new framework for developing DTI models.

Chapter 2

Debiasing Algorithms for Protein/Ligand Binding Data do not Improve Generalisation

2.1 Introduction

Many recent ML approaches to virtual screening have achieved outstanding success on benchmark datasets that are randomly partitioned into train and validation sets, with AUCs (area under the Receiver Operator Characteristic curve) routinely exceeding 0.9 [66, 54, 17, 71, 35, 53, 55, 24]. However, it is unclear whether this impressive performance indicates that a model that can truly generalise across chemical space, or instead simply overfits the training data [70, 11, 59, 61, 72, 43]. Since chemical space contains clusters of molecules around scaffolds, memorizing the properties of a few scaffolds can be sufficient to perform well, masking the fact that the model may not generalise beyond close analogues [29, 26]. Further, molecules tested experimentally are generally designed by humans and therefore likely to be easy to synthesize and similar to previously known binders [11, 33]. Finally, if, due to properties of the dataset, it is possible to classify the ligands based only on simple properties like number of hydrogen bonds, classifiers may perform outstandingly well on training data but be completely unable to make predictions in other parts of chemical space [59].

To counter this limitation, data bias definitions and corresponding debiasing algorithms have been introduced [59, 74, 5, 72]. Two popular bias measures, Maximum

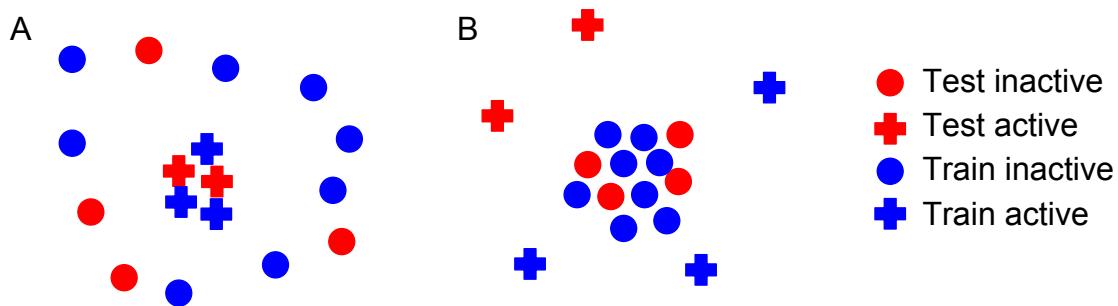


Figure 2.1.1: Definition of AVE and MUV Bias. (a) This set is considered biased by MUV because actives are clustered and not uniformly embedded in the inactive decoys. (b) This set is considered biased by AVE, because the inactives are tightly clustered relative to the active-inactive distance.

Unbiased Validation (MUV) and Asymmetric Validation Embedding (AVE), are illustrated in Fig. 2.1.1 [59, 72]. For AVE bias, the bias measure is used by a genetic algorithm to rearrange the train/validation split such that the bias is reduced [72]; MUV bias is used by a similar genetic algorithm to select a benchmark set with minimal bias, from which random train/validation splits can be generated [59]. Specifically, MUV ensures that active ligands are uniformly embedded among inactive ligands according to some distance metric, while AVE adds the requirement that inactive ligands are not tightly clustered. Across benchmark datasets for multiple protein targets, both bias metrics were shown to correlate to model performance, suggesting that heavily biased datasets provide a falsely optimistic picture of the predictive ability of the trained model. Furthermore, debiasing was shown to decrease classification accuracy for the debiased validation set, presumably because the resulting models could not perform well by simply memorising the training data [72, 59].

Further studies have established that debiasing with simple descriptors, like molecular weight and number of hydrogen bonds, can be used to obtain a more predictive metric of model performance [62]. This work focused on descriptors that were known not to be correlated with activity, so any clustering in the space of simple descriptors had to be a result of bias. We will instead focus on fingerprints like the original AVE study [72]. Since fingerprints may be correlated with activity even in the true binding function, we expect to see some clustering that is not a result of bias, which makes debiasing more difficult.

Debiasing algorithms were originally designed to provide a more realistic metric

for measuring real-world model performance [72, 59]. However, another purported advantage of models trained using debiased data splits is that they overfit less, so generalise better to make accurate predictions for novel candidate ligands. Despite seeming reasonable, the argument for this supposed advantage has a logical flaw: a model that overfits less will not always perform better at generalising to novel ligands. In this chapter, we develop a framework to measure generalisation ability, and explicitly test these hypotheses. We assemble data for 189 targets with greater than 500 reported active ligands, and split each dataset into a train set and a distant held-out test set used to define the far-AUC, a metric of model generalisation (see Fig. 2.2.1a). We then randomly split the train set to produce a random held-out validation set, which is used to measure the standard AUC. Despite achieving state-of-the-art AUCs on the held-out validation sets, our trained ML models struggle to generalise effectively when challenged with the distant held-out test sets. We then apply MUV and AVE debiasing to our 189 datasets, and use the resulting debiased train sets to build new models. We find that the debiased models make less accurate predictions for novel candidate ligands, as illustrated by their performance on the distant held-out test sets. Further, counter to the stated aim, the performance of our models on the distant held-out test sets is not well-predicted by their performance on the debiased validation sets, suggesting that debiasing does not yield a particularly realistic metric for measuring model performance.

2.2 Methods

We filter protein-ligand binding data from ChEMBL 24.1 [15, 21], acquiring active ligands (IC_{50} , K_i , K_d , or EC_{50} of less than 1 μM) for each protein target. We acquired data for inactive ligands from PubChem indexed by UniProt Protein ID [48, 36]. The handful of cases where ligands were marked both active and inactive by different assays were eliminated. Some targets have fewer inactives than actives, in which case we randomly drew inactives from ChEMBL to achieve an even split for every target. This procedure was repeated every time the algorithm was run, contributing to the error bars shown in the figures.

We use ECFP6 fingerprints with 2048 bits as the feature set for all models [58]. For

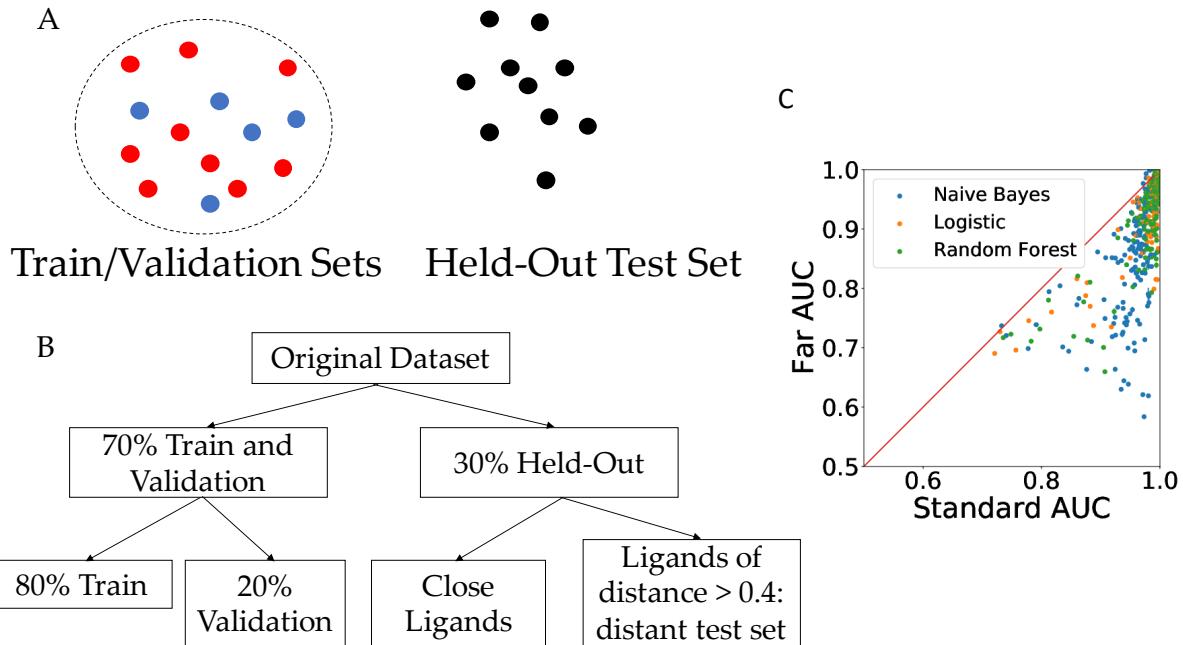


Figure 2.2.1: (a) Definition of the far-AUC, the AUC using a held-out test set with minimum distance of 0.4 from all elements of the training or validation set. (b) A flowchart depicting the splitting process to generate the far-AUC. (c) Comparison of the standard model AUC and the far-AUC, a measure of generalisation. Model predictions are less accurate for the distant held-out test set than for the validation set, showing that these models do not generalise.

consistency we use Jaccard distance as the distance metric throughout, computed from Scipy [34] (note this differs from the specific metric used in MUV [59]). We first randomly split both the actives and inactives for each protein target into a 70% set and a 30% set. Each 30% set is filtered to build distant held-out test-sets , wherein all ligands are at least 0.4 from every ligand in the 70% set. We observe empirically that these distant held-out test sets typically have approximately 10% of the total active and 25% of the total inactive ligands, respectively. Our far-AUC metric measured the performance of models on this distant held-out test set, which only contains molecules distant from any molecules (active or inactive) that the model has previously seen. Fig. 2.2.1b summarizes the procedure of generating this held-out test set.

We further randomly split the remaining 70% set for each target into train (80%) and validation (20%) sets and train Naive Bayes, Logistic Regression, and Random Forest models using these data splits. All models used were implemented with scikit-learn [51]. We use no prior for Naive Bayes, $C = 1$ for Logistic Regression, and 100

2.2 Methods

trees with a maximum depth of 10 for Random Forest. We do not tune model hyperparameters since the focus of our work is the debiasing algorithms. The 3-way train/validation/test split and the requirement for data points far from the training set restricted us to the 189 protein targets with greater than 500 active ligands. All AUCs are measured over 50 replicates to determine error bars. Randomness comes from the train/validation/test split and (where needed) the selection of inactives. All error bars are SEM and indicated to 1σ precision.

We follow the original definitions of AVE and MUV bias [72, 59]. Specifically, given two sets V, T of molecules (which may be the same) and a similarity threshold $d \in [0, 1]$, we define a nearest-neighbor function

$$S_{(V,T,d)} = \frac{1}{||V||} \sum_{v \in V} I_d(v, T) \quad (2.1)$$

where $I_d(v, T) = 1$ if the distance from the molecule v to its nearest neighboring molecule in T that is not itself is smaller than d . We then define a distance function on sets

$$H_{(V,T)} = \frac{1}{||D||} \sum_{d \in D} S(V, T, d) \quad (2.2)$$

with $D = \{0, 0.02, \dots, 1\}$. For AVE bias, let V_a, V_i, T_a, T_i be the sets of validation actives, validation inactives, test actives, and test inactives, respectively. Then the AVE bias is defined as

$$B_{\text{AVE}} = H_{(V_a, T_a)} - H_{(V_a, T_i)} + H_{(V_i, T_i)} - H_{(V_i, T_a)}. \quad (2.3)$$

For MUV bias, let the benchmark actives be B_a and the benchmark inactives be B_i . Then, the MUV bias is defined as

$$B_{\text{MUV}} = H_{(B_a, B_a)} - H_{(B_a, B_i)}. \quad (2.4)$$

This formulation is equivalent to the original definition of MUV bias.[59] Note that $H_{(B_a, B_a)} \neq 1$ since we do not count the original molecule in Equation 1.

To minimise MUV or AVE bias we use the implementation developed in the original AVE paper [72], which largely follows that in the original MUV paper [59]. A number of random initial train/validation splits are generated; then the mutation phase of the genetic algorithm involves randomly merging two train/validation splits, moving

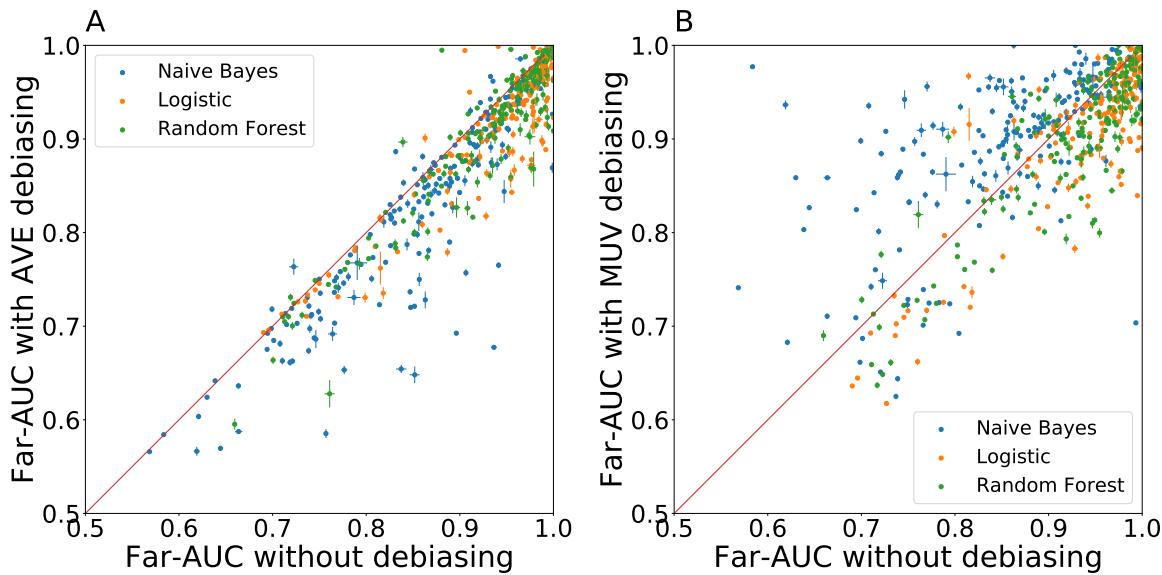


Figure 2.2.2: Impact of debiasing on the far-AUC; points below the diagonal indicate targets for which debiasing decreased the far-AUC. (a) AVE debiasing reduces the ability of the models to generalise. (b) MUV debiasing does not consistently help the models to generalise. These results suggest that debiasing does not improve the ability of the models to generalise to novel candidate ligands.

compounds between the training and validation sets, and deleting compounds from either set. The algorithm ran for 300 iterations or until bias was < 0.01 , whichever occurred first, and then produced the least biased split. For MUV bias, the train/validation splits are replaced by benchmark/non-benchmark splits, and we then perform a random train/validation split with 20% validation within the benchmark dataset.

2.3 Results

We first use our framework to test the extent to which ML models are able to generalise and make accurate predictions for novel candidate ligands. To assess this we compare the ability of each trained model to accurately classify ligands in (i) the random held-out validation set, reported by the standard AUC and (ii) the distant held-out test set, reported by the far-AUC. The distant held-out test sets mimic the real-world need to make accurate predictions for novel candidate ligands that are distinct from the training data. The results of our analysis for Naive Bayes, Logistic Regression and Random Forest models are shown in Fig. 2b. We find that the far-AUC is significantly lower

than the standard AUC for all models tested across all 189 target datasets, indicating that our models do not generalise well. This confirms the hypothesis that generalisation is a challenge for ML models trained on protein/ligand binding datasets. Although we have not tested more complex models, it is likely that as the complexity of the ML model increases, overfitting to the training data will increase.

We next assess the extent to which MUV and AVE debiasing alleviate this issue. The key question is whether these algorithms remove data biases that prevent the models from generalising or instead remove useful information that is necessary for the model to learn. This is a nontrivial question, as both MUV and AVE rely on the inherent assumption that the metrics being used to measure distances between ligands do not correlate strongly with the binding activity of the ligand. The ultimate goal of a ML model is to learn a function of the data features that distinguishes ligands that bind to a given protein from those that don't. In contrast, the goal of debiasing is to ensure that actives and inactives are well-mixed according to some distance metric, which is itself a function of the data features. If this distance metric happens to be correlated with the physico-chemical criteria required for binding, then debiasing will remove important information from the training data, potentially harming the performance of the model. Unlike the situation for simple descriptors [62], we do expect some correlation between fingerprint distance and true binding activity.

To address this question we use the distant held-out test sets to compare the existing models with versions trained using debiased train/validation splits. We evaluate the effect of debiasing by computing the change in the far-AUC score between the models trained on the debiased data, and the models trained on the original data for each target. As shown in Fig. 2.2.2, neither AVE nor MUV debiasing improves the generalisation ability of the trained models. On average, AVE decreases the far-AUC of Logistic Regression by 0.024 ± 0.030 (mean \pm standard deviation) and that of Random Forest by 0.021 ± 0.029 (both significant at a $p < 0.01$ level), while MUV debiasing decreases the far-AUC of Logistic Regression by 0.028 ± 0.040 and of Random Forest by 0.024 ± 0.040 (both significant at a $p < 0.01$ level).

To check whether the far-AUC depends on the extent to which the data were debiased by either MUV or AVE, Figs. 2.3.1a and 2.3.1b show the change in far-AUC as a function of the final dataset bias achieved by AVE and MUV, respectively. We find no

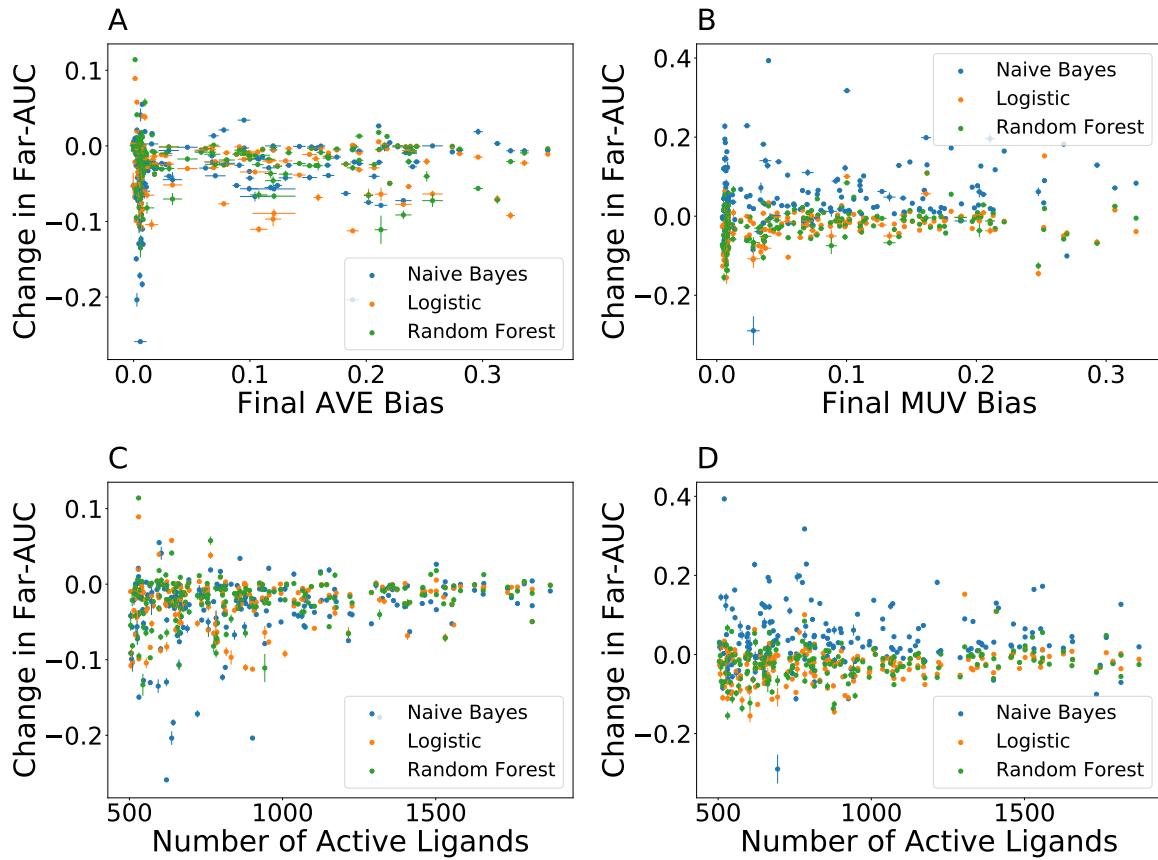


Figure 2.3.1: Relationship between the change in far-AUC achieved by debiasing and either (a) the final AVE bias or (b) the final MUV bias; we find no correlation. Furthermore we see no improvement in the resulting generalisability as a function of the number of active ligands per target for (c) AVE or (d) MUV debiasing. These results suggest that neither approach has much effect for large datasets.

correlation in each case.

We further probe if the number of active ligands for a target indicates whether debiasing will prove effective. Figs. 2.3.1c and 2.3.1d show no correlation between the number of active ligands and the change in far-AUC achieved by debiasing. However, the magnitude of the change in far-AUC decreases slightly as the number of active ligands increases, suggesting that debiasing has a smaller effect for large datasets. In addition, we checked whether there was any relationship between the change in the far-AUC and the number of decoy molecules added for a given target to achieve equal numbers of active and inactive ligands. Fig. 2.A.1 confirms that there is no correlation.

Our results also allow us to examine the hypothesis that the validation AUC produced from a debiased dataset is a better predictor of real-world model performance

and generalisation ability. In order to do this, we can compare the validation AUC produced by the debiasing algorithm to the far-AUC on the distant held-out test set. Fig. 2.3.2a demonstrates that there is significant disagreement between the standard AUC and far-AUC with AVE debiasing, with an RMS error of 0.19, an increase from the RMS error of 0.15 between standard AUC and far-AUC without debiasing. There is some correlation between the validation AUC and the far-AUC with AVE debiasing, with a correlation coefficient of 0.72, but this is comparable to the correlation coefficient of the standard AUC and far-AUC without debiasing, which is 0.69. Similarly, in Fig. 2.3.2b MUV has an RMS error of 0.13 and a correlation of 0.79, a slight improvement over the results for the original standard AUC. These results suggest that validation AUCs from debiasing algorithms are in significant disagreement with the far-AUC, and thus likely not good measures of generalisation ability of a given model.

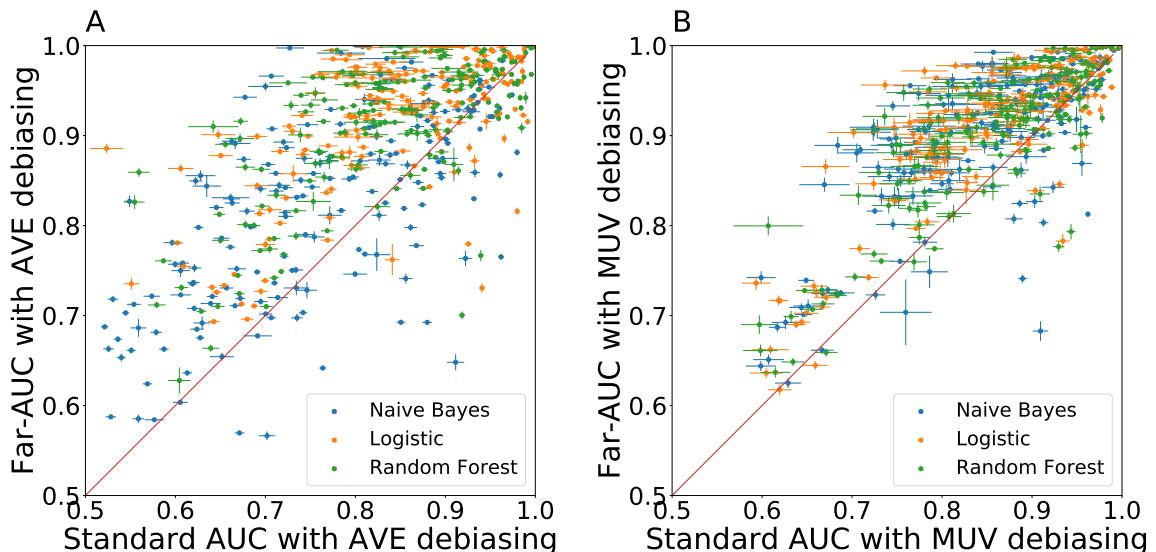


Figure 2.3.2: Comparison of standard AUC and far-AUC after (a) AVE and (b) MUV debiasing. Standard AUCs on the validation set are not particularly representative of the far-AUC, and therefore of model generalisation ability, after debiasing.

To better understand these findings, and ask whether there are situations in which debiasing does improve the ability to generalise, we examine the data generated by our experiment in more detail. We first examine far-AUC trajectories measured during debiasing. Figs. 2.B.1 and 2.B.2 suggest that there is a positive correlation between the far-AUC achieved and reduction of either the MUV or AVE bias. The jagged behaviour

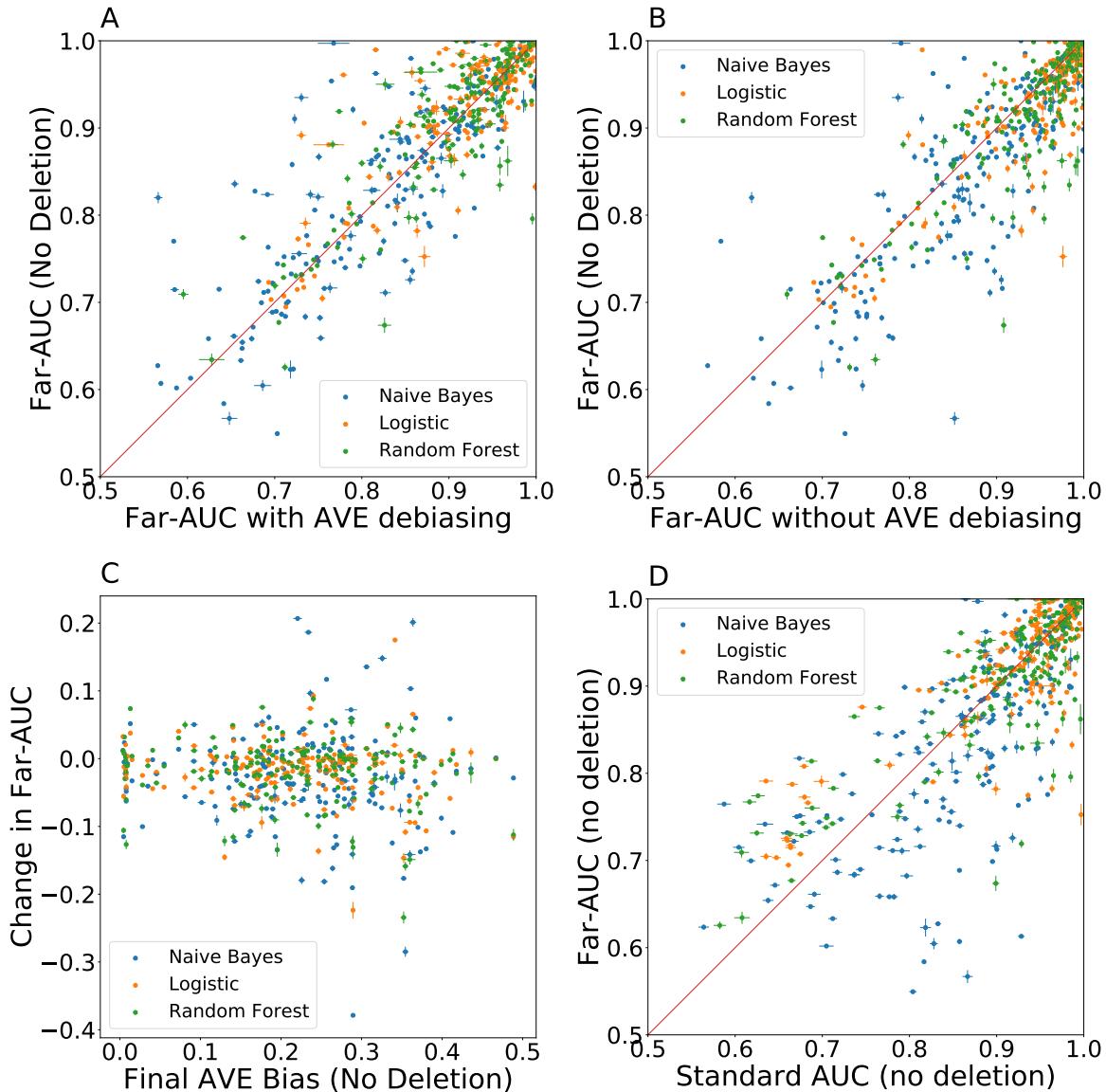


Figure 2.3.3: AVE debiasing without deletion. (a) When deletion is forbidden for the AVE debiasing algorithm, performance improves slightly over that obtained with the standard version. (b) However, debiasing without deletion is worse, on average, than the performance obtained by the models before debiasing. (c) There is no clear relationship between the change in far-AUC achieved when deletion during debiasing is forbidden and the final AVE bias. (d) Standard AUCs on the validation set are not representative of the far-AUC after AVE debiasing when deletion is forbidden.

seen for both algorithms suggests that certain moves that significantly decrease the far-AUC are sometimes preferred.

In the case of AVE debiasing, we hypothesised that moves in these trajectories that decrease the far-AUC correspond to data deletion from both the training and vali-

dation sets, a move allowed by the genetic algorithm. For MUV debiasing, the only allowed move is deletion from the benchmark dataset, so this is irrelevant. For example, Fig. 2.C.1 shows how the size of the dataset for CHEMBL5508 decreases during the debiasing process. To evaluate debiasing in the absence of data deletion, we modified the genetic algorithm to prevent data from being deleted. Fig. 2.C.1 shows that the modified version of AVE debiasing still succeeds at reducing the bias of the dataset. Could this provide an approach that better enables the models to generalise? To test this we repeated our earlier analysis with the modified debiasing algorithms.

As shown in Fig. 2.3.3, we find that forbidding deletion does slightly improve the generalisation ability of the resulting debiased models for AVE compared to the versions with deletion. However, even without deletion, AVE decreases the far-AUC of Logistic Regression by 0.016 ± 0.041 and of Random Forest by 0.017 ± 0.044 on average across the 189 datasets (both significant at a $p < 0.01$ level). As in the case where deletion is allowed, Fig. 2.3.3c confirms that there is no relationship between the extent to which the data were debiased and the change in far-AUC obtained.

We also examined whether the debiasing algorithm proves more capable of measuring the generalisation ability of a model when deletion is forbidden. Our results in Fig. 2.3.3d show a marginal improvement. The RMS error between the standard and far-AUC in AVE debiasing drops to 0.12; the correlations between the two also increase to 0.76. However, our results suggest that AVE debiasing does not consistently measure generalisation ability even when deletion is forbidden.

Overall, our results indicate that it is difficult to predict when AVE or MUV debiasing will improve generalisation and whether performances measured under AVE or MUV debiasing are representative of real-world model performance. Further work is needed to determine the conditions under which debiasing has the potential to improve the ability of a model to generalise and make accurate predictions for novel candidate ligands.

2.4 Discussion

In this chapter we develop a simple far-AUC metric that measures the ability of a protein-ligand binding model to generalise and make accurate predictions for novel

candidate ligands. We use this metric to evaluate the AVE and MUV debiasing algorithms that were designed to reduce overfitting to the training data, and thus potentially improve the ability of models to generalise. Our analysis for both AVE and MUV debiasing in Fig. 2.2.2 finds that debiasing does not systematically improve the ability of the trained models to generalise, despite the fact that Fig. 2b shows there is significant room for improvement. Further, our results in Fig. 2.3.2 suggest that validation AUCs measured after debiasing are not particularly representative of the far-AUC, suggesting that they do not correlate with real-world model performance or generalisation ability.

This suggests that debiasing algorithms are not able to accurately distinguish signal, or features correlated with activity in the true binding function, from bias, or features correlated with activity due to the limited data we have, and in many cases remove relevant information from the training data. Our analysis of the debiasing trajectories suggests that the deletion operation used by the genetic algorithm in both MUV and AVE debiasing may exacerbate this loss of useful information or signal from the data. To address this we implemented versions that did not allow data points to be deleted. This did not result in models that were better able to generalise compared to those built without debiasing.

Dataset bias is clearly an important issue, particularly in chemistry; however, current debiasing approaches need to be applied carefully to ensure that they do not eliminate relevant information. Indeed, some clustering among actives is to be expected in fingerprint space, since active ligands for a given protein can have structurally similar features. It is important to distinguish between this clustering and artificial clustering that may result from the fact that only portions of chemical space have been explored by synthetic chemists, or other potential sources of bias.

Another approach is to better understand the regions of chemical space in which protein-ligand binding models trained using a particular dataset are able to make accurate predictions. Generalisation is challenging for ML models across many contexts, even when trained with unbiased datasets, so given the highly biased nature of chemical data, expecting protein-ligand binding models to generalise may be ambitious. Methods that establish the domain of applicability for trained models need to be developed to provide confidence in those predictions that fall within this domain. This

2.4 Discussion

approach would have the advantage of avoiding the information/bias distinguishability problem described above while still allowing the resulting models to generalise to some degree.

Appendix: Supporting Information

2.A Change in Far-AUC

We also examined whether there was a relationship between the change in far-AUC and the number of decoys randomly added as inactives. Fig. 2.A.1 reveals that there is no such relationship.

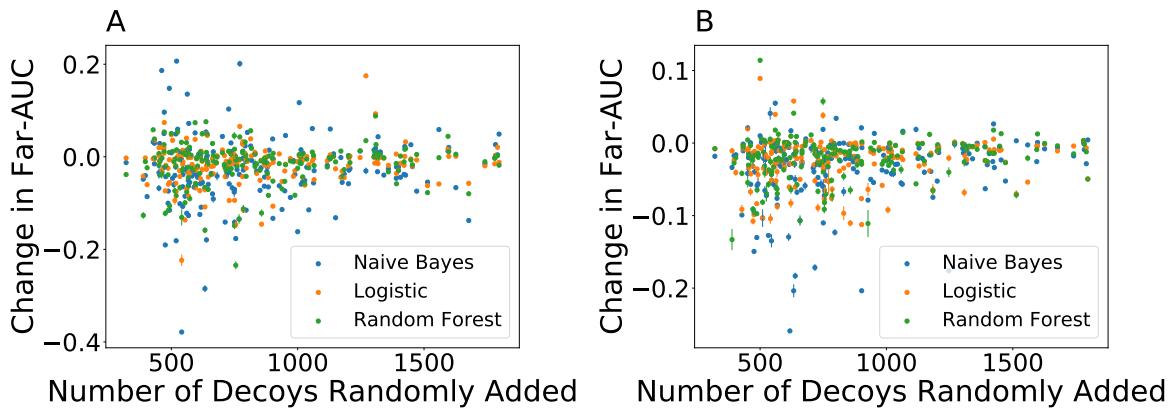


Figure 2.A.1: There is no clear relationship between the change in far-AUC achieved and number of decoy molecules added to increase the number of inactive ligands for AVE debiasing either (a) with or (b) without deletion.

2.B Tracking Far-AUC during debiasing

To better understand how the debiasing algorithms operate, we tracked the far-AUC at each step of the debiasing algorithm for both MUV and AVE bias. The results are shown in Figs. 2.B.1 and 2.B.2. We see a fairly clear positive correlation in the case of AVE bias, suggesting that on average biased sets are actually better at generalisation. This correlation is clearly dominated by overall jagged behavior, suggesting that there

are certain sequences of moves that dramatically affect generalisation ability and gradually reduce overall AVE or MUV bias. We suspected that some of these moves were deletions, leading to our analysis of the performance of the AVE debiasing algorithm without deletion. The overall shape of the trajectory however suggests no particular way to modify the debiasing algorithm to improve the resulting measured far-AUC.

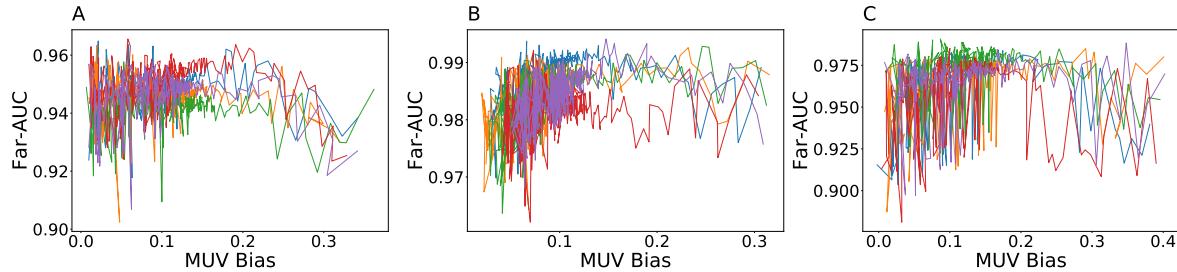


Figure 2.B.1: Correlation between MUV Bias and logistic regression far-AUC tracked for 5 iterations at each step of the debiasing algorithm for 3 targets. The jagged behavior observed in Fig. 2.B.1c suggests that certain moves that significantly worsen model generalisability, but this behavior is not that common.

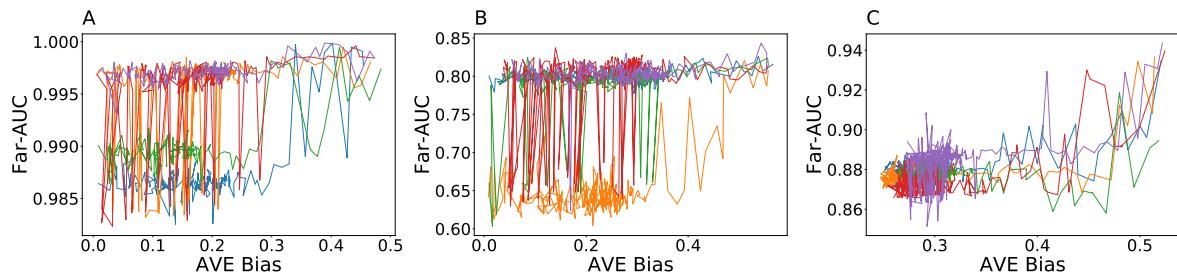


Figure 2.B.2: Correlation between AVE Bias and logistic regression far-AUC tracked for 5 iterations at each step of the debiasing algorithm for 3 targets. The clear general positive correlation suggests that biased datasets are better at generalising. The jagged behavior suggests that certain moves that significantly worsen model generalisability

2.C AVE Bias and Dataset Size

We examined the relationship between AVE bias and dataset (training + validation) size after noting that the debiasing procedure consistently preferred to remove points from both training and validation sets. There is no particularly strong relationship between a random train/validation split's AVE bias and the overall dataset size, but

there is a strong relationship between these two variables for splits produced by the debiasing algorithm as shown in Fig. 2.C.1a .

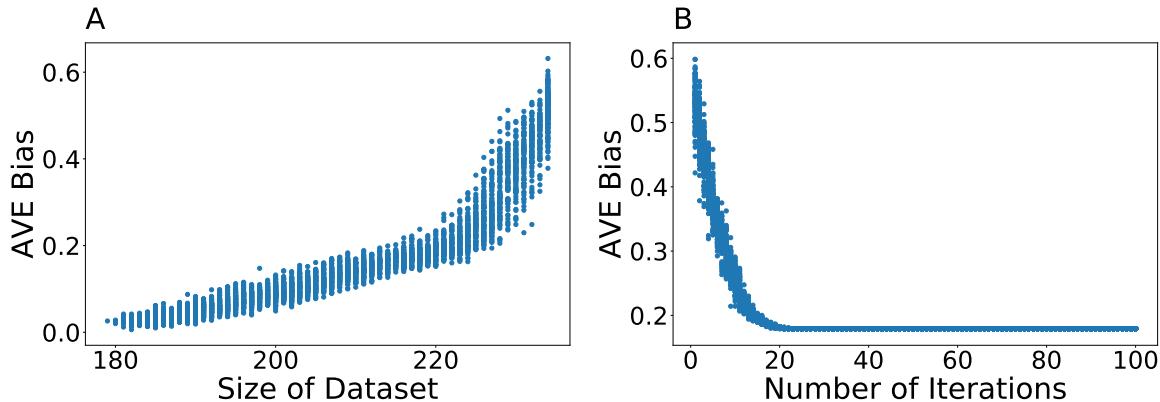


Figure 2.C.1: Analysis of the Relationship between AVE Bias and Dataset Size for CHEMBL5508. In (a), we observe that as the algorithm proceeds, the AVE bias shrinks by shrinking the size of the overall dataset (training + validation). In (b), we observe that it is still possible to debias the dataset using AVE without deleting points.

Preventing our algorithm from deleting data points was partially successful in debiasing the dataset, as shown in Fig. 2.C.1b. These results were consistent across all the examples we tried. However, as discussed in the main paper, they did not significantly improve generalisability of our models.

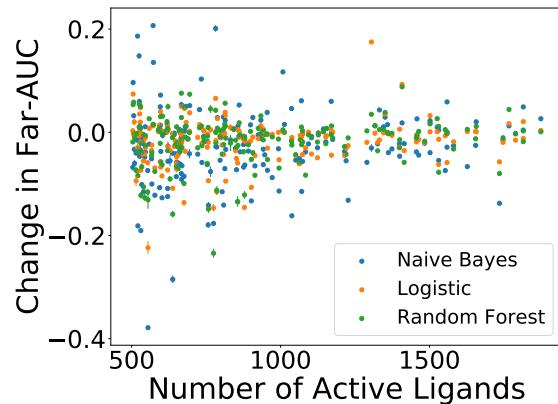


Figure 2.C.2: There is no clear relationship between the change in far-AUC achieved by debiasing without deletion and the number of active ligands per target for AVE debiasing without deletion.

Chapter 3

Evaluating Generalisation and Attribution Ability of Fingerprint-Based Protein/Ligand Binding Models

3.1 Introduction

Generalisation in ligand space and attribution are two key challenges for single protein virtual screening models. We have already established that many high-performing ML models for virtual screening struggle to generalise well far from their training data. Further, recent studies analyzing how neural network models in particular attribute their results suggest that even high-performing models often do not learn the correct rule [47]. Attribution of virtual screening models is particularly important because models that consistently misattribute can be fooled by adversarial examples and can mislead medicinal chemists trying to refine potential drug candidates using predictions from the model [47].

Some methods have been proposed to mitigate these issues of generalisation and attribution. In particular, random matrix theory (RMT) has been used to eliminate noise in chemical data [40]. These methods note that a randomly sampled set of chemical ligands has a fingerprint covariance matrix whose eigenvalues follow the Marchenko-Pastur distribution, implying that the covariance matrix would be the identity were it not for noise due to finite sampling in a large multidimensional space [40]. As such, any signal found in a set of active ligands would be found in eigenvectors with eigen-

values above the Marchenko-Pastur cutoff [40]. This observation can be used to developed a number of RMT-based models, ranging from simple ones with no parameters to using the method as feature selection prior to running some other model [40]. These models claim to improve the generalisation ability of models and be more interpretable than standard models, allowing medicinal chemists to make refinements using their predictions [40, 41].

In this chapter, we evaluate the ability of virtual screening models to generalise and attribute their results. We begin by proposing simple, general frameworks for evaluating the generalisation ability and attribution of a particular model. Our framework for generalisation is based on a distance-based split; our attribution scores are based on results on toy datasets and simply identify a subset of the fingerprint bits necessary for a model to obtain a particular score, thus bypassing the need for a model to be differentiable. We then evaluate a number of standard and RMT-based models on these metrics. Our results clearly establish that high-performing models by AUC on randomly held-out validation sets often do not generalise well and models that perform and generalise well may still not learn the correct binding rule. In particular, contrary to previous results, we find that RMT-based models perform worse and are consistently worse at generalisation than standard counterparts, suggesting that the RMT framework is not functioning as expected.

We then examine why models consistently misattribute on toy datasets despite performing well and propose some solutions that can help improve model performance overall. Specifically, we demonstrate that the addition of fragment-matched decoys, decoys that have similar fragments as active molecules but are inactive, can help reduce rates of misattribution. Further, we show that a number of misattributions can be traced to spurious correlations in the dataset that originate in the background, or the area of chemical space accessible to screens. We suggest some potential approaches for models to account for these spurious background correlations or for better screening libraries to be developed that may have fewer correlations. Our results provide a better understanding of why virtual screening models fail to generalise and attribute correctly and what steps can be taken to mitigate these issues.

3.2 Methods

3.2.1 Datasets

We used the same protein-ligand binding data as in our previous work [64]. Specifically, we acquired active ligands for each protein target from ChEMBL 24.1 [15, 21]; our criterion for activity was an IC_{50} , K_i , K_d , or EC_{50} of less than $1 \mu\text{M}$. We obtained inactive ligands from PubChem indexed by UniProt Protein ID [48, 36] and eliminated the small number of conflicts between the two. Since there were often fewer inactives than actives, we randomly drew additional decoys from ChEMBL to achieve an even split for every target. This procedure was repeated for every repetition of the algorithm, contributing to the error bars computed.

Our work on attribution also required the construction of a number of toy datasets. Our method of constructing these toy datasets was inspired by but not identical to previous work done on neural networks [47]. Each toy protein had a specified binding logic requiring that up to 3 fragments be present (called one-element, two-element, and three-element toy datasets); the fragments that were required were selected randomly from the set of possible fragments in Table 3.A.1. We identified 600 active and 600 inactive ligands from the ZINC12 database [32] that were active per our toy binding logic to generate the toy dataset. The binding logic we used for each dataset, along with their names, can be found in Table 3.2.1; the exact SMARTS code used for each fragment can be found in Table 3.A.1.

For the two-element and three-element toy datasets, we also generated toy datasets with fragment-matched decoys. These datasets were intended to improve attribution results by providing our algorithms with decoys that were more closely matched to the actives. Specifically, for the two-element datasets, we included 200 generic inactives as before, 200 inactives with only the first fragment, and 200 inactives with only the second fragment. For the three-element datasets, we included 150 generic inactives, 150 inactives with the first two fragments, 150 inactives with the first and third fragments, and 150 inactives with the second two fragments.

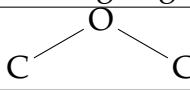
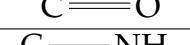
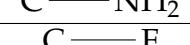
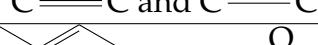
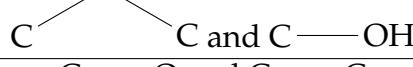
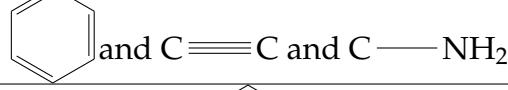
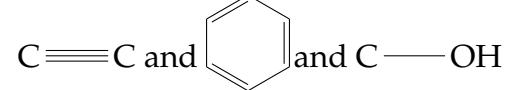
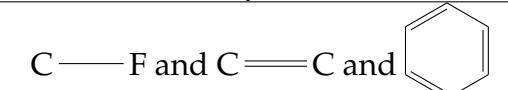
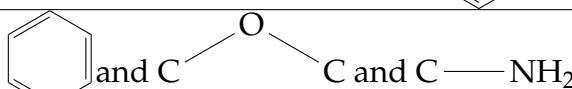
| Dataset | Binding Logic |
|------------|---|
| OneElem0 |  |
| OneElem1 |  |
| OneElem2 |  |
| OneElem3 |  |
| TwoElem0 |  |
| TwoElem1 |  |
| TwoElem2 |  |
| TwoElem3 |  |
| ThreeElem0 |  |
| ThreeElem1 |  |
| ThreeElem2 |  |
| ThreeElem3 |  |

Table 3.2.1: Binding Logics Used for Construction of the Toy Datasets. These were used to perform our attribution analysis.

3.2.2 Models

The feature set for all our models was ECFP6 fingerprints with 2048 bits [58]. When we required a distance metric, we used Jaccard distance as the distance metric computed from Scipy [34]. We initially tested 7 different models: 4 relatively standard ones and 3 based on random-matrix-theory-related approaches. Details of the models tested follow:

1. 1-Nearest Neighbor (1-NN): a baseline model that simply picks the nearest ligand.
2. Naive Bayes (NB): implemented using scikit-learn [51] with no prior.
3. Logistic Regression (LR): implemented using scikit-learn with $C = 1$ as the regularization constant.
4. Random Forest (RF): implemented using scikit-learn with 100 trees and a maxi-

mum depth of 25.

5. Random Matrix Theory (RMT): this is the simplest random matrix theory model that makes predictions based on distance to the hyperplane spanned by the significant eigenvectors in the active correlation matrix [40]. It has no parameters and completely ignores inactives.
6. Random Matrix Discriminant (RMD): this model upgrades the initial random matrix theory model by also incorporated correlations in the inactives and using two distance cutoffs [41].
7. RMT + Logistic (RMT + LR): this model uses random matrix theory as a feature selection method, picking the significant eigenvectors from the overall correlation matrix of both actives and inactives, and then runs logistic regression with $C = 1$ on the ensuing features.

Details about the RMT-based methods may be found in Section 3.A.2.

3.2.3 Evaluation Methods

The first evaluation method we used on our models was a standard validation AUC (area under the Receiver Operator Characteristic curve) on a randomly held-out validation set that comprised 20% of the data, with an even split of actives and inactives. We compared this to the far-AUC, as previously described. We repeated all standard AUC and far-AUC calculations 50 times to obtain error bars.

Since some of our models were not differentiable, we developed an attribution method designed to work on any chemical-fingerprint-based method. Our method relies on the use of SIS (sufficient input subsets), which identifies a sufficient subset of the input features that would reach a particular score on a given model [7]. Our feature mask was simply the vector of all 0s, i.e. no fragment present. If our model predicted a probability of x for a given molecule, our threshold was $\frac{\lfloor 100x \rfloor}{100}$ if x was not a multiple of 0.01 and $x - 0.01$ otherwise. Thus, given a particular active molecule and an already trained model, SIS identifies the features of the molecule that are necessary for the model to infer that it is active, i.e. attain a score close to that predicted by the model.

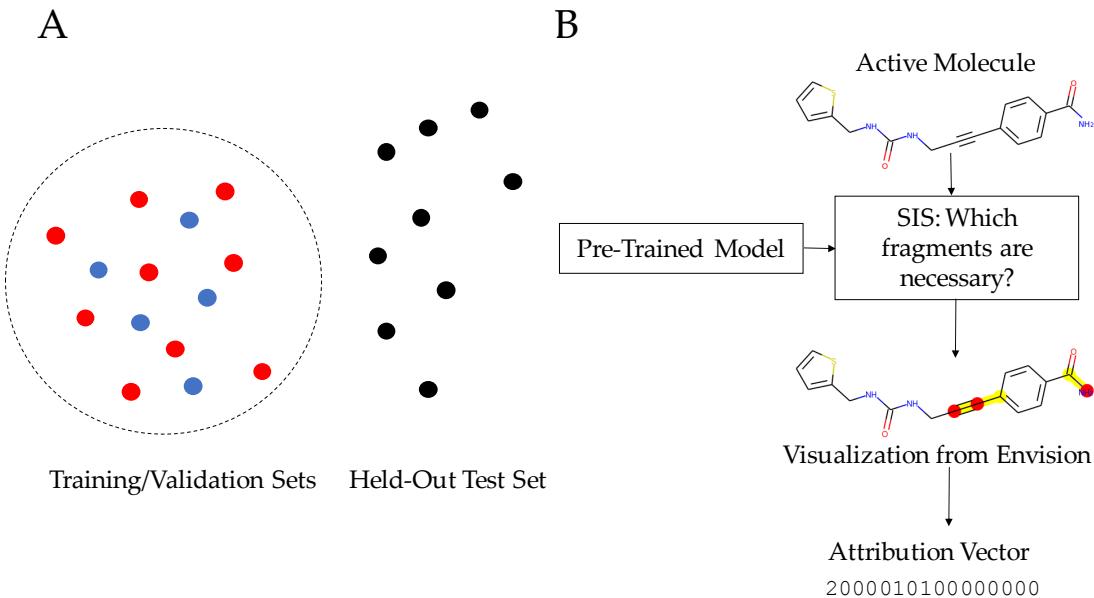


Figure 3.2.1: (a) The Far-AUC is the AUC on a distant held-out test set, used to measure the generalisation ability of our models. (b) In order to evaluate what our models were learning, we developed an attribution method relying on SIS, which determines which bits/fragments are necessary for a particular molecule to attain a given score on a given pre-trained model. This was then translated into a visualization via Envision and also into an attribution vector, where each number corresponds to a single atom and represents how many essential fragments that atom is in. We then developed a number of attribution scores based on this vector, most notably the attribution score on a toy dataset which was simply the dot product with the correct attribution vector.

The next step in our attribution procedure is to translate the features (fingerprint bits) back into fragments. This is generally impossible because each fingerprint bit corresponds to multiple fragments. For a particular molecule, rdkit can identify the relevant fragment or fragments for each bit [39]. A previous graduate student developed a graphical tool named Envision to visualize the relevant fragments, which was used for all the visualizations in this chapter.

We next developed a quantitative measure of attribution. For any given pre-trained model m and a given molecule, we generated an attribution vector \mathbf{v}_m with one element per atom corresponding to the number of fragments containing that particular atom that were considered significant according to SIS. If a single bit corresponded to multiple fragments, all were considered significant. For the toy datasets, we also produced a real attribution vector \mathbf{v}_R with a 1 on any atom that contained a relevant

fragment and a 0 otherwise. See Fig. 3.2.1b for a flowchart of the process of obtaining these attribution vectors.

This allowed us to generate 4 measures of attribution accuracy on our toy datasets. Our attribution score for the model m was the $\frac{\mathbf{v}_m \cdot \mathbf{v}_R}{\|\mathbf{v}_m\| \|\mathbf{v}_R\|}$. This score was then normalized with respect to the attribution score obtained on a model trained with randomly labeled data to ensure that our attribution scores were not measuring anything irrelevant about the structure of the data [1]. The attribution false positive score is the proportion of nonzero entries in \mathbf{v}_m that are zero in \mathbf{v}_R , i.e. atoms that the model thinks are relevant but actually are not. The attribution false negative score is the proportion of nonzero entries in \mathbf{v}_R that are zero in \mathbf{v}_m , i.e. atoms that the model thinks are not relevant but actually are. Finally, the comparative attribution score between two models m_1 and m_2 is $\frac{\mathbf{v}_{m_1} \cdot \mathbf{v}_{m_2}}{\|\mathbf{v}_{m_1}\| \|\mathbf{v}_{m_2}\|}$; this score could be computed for both toy and real datasets.

Due to the computational expense of the attribution procedure, we ran the attribution analysis on 50 molecules randomly selected that were predicted to bind with probability at least 0.99 for the real data and 0.95 for the toy data. In this work, we report both averages and distributions of the attribution scores across the data tested.

3.3 Results

3.3.1 Comparison of Model Performance

We began with a comparison of the standard AUCs of all models on all datasets tested; see Fig. 3.3.1 for our results. Random forest and logistic regression consistently performed the best, with random forest getting an $AUC = 0.0004 \pm 0.008$ better across all targets (mean \pm 1 standard deviation, not statistically significant). Logistic and random forest both score roughly 0.006 ± 0.012 better than RMT + logistic, which performs 0.0043 ± 0.012 better than Naive Bayes. Naive Bayes's standard AUC is 0.014 ± 0.037 better than that of RMD, which scores 0.043 ± 0.042 better than RMT. Finally, RMT performs 0.017 ± 0.049 better than 1-NN under the standard AUC (all other results statistically significant at a $p < 0.01$ level).

Random-matrix-theory-based methods in general do not perform as well as their standard counterparts, suggesting that the random matrix theory framework as cur-

rently applied is flawed. In particular, the addition of random matrix theory as a feature selection method does not help logistic regression, and RMD is somewhat consistently outperformed by Naive Bayes, in contrast to previous findings [41]. We also see high variance in model performance both within the same dataset and across different datasets for low-performing models, so it is possible that a poorer model may outperform a better one on any particular dataset. In the future, researchers should be careful to test their models on a large number of datasets to eliminate this variance.

3.3.2 Generalisation

We next evaluated the ability of our models to generalise by comparing their far-AUC to their standard AUC. As has been found in previous work [64, 59, 72] and as seen in Fig. 3.3.2, the far-AUC of all models is significantly lower than the standard AUC, suggesting that our models fail to generalise to distant held-out test sets (all results statistically significant at a $p < 0.01$ level). Explicitly comparing the RMT-based models to the others, we see that the use of RMT worsens the ability of any given model to generalise. This is most clearly seen with logistic regression: without RMT, the far-AUC is 0.033 ± 0.035 lower than the standard AUC, but with RMT it is 0.047 ± 0.050 lower. Thus RMT is not able to fulfill its purpose of generalising successfully to distant held-out test sets, contrary to previous claims [40, 41].

Just as in Fig. 3.3.1, we compare the far-AUCs of various models in Fig. 3.3.3. Now, logistic regression is clearly the best model, with an average far-AUC 0.004 ± 0.022 better than random forest, which is 0.015 ± 0.034 better than RMT + logistic. RMT + Logistic performs 0.018 ± 0.041 better than Naive Bayes, which scores 0.048 ± 0.083 better than RMD. RMD performs very similarly to our baseline 1-NN, with a far-AUC difference of only 0.002 ± 0.068 (not statistically significant), and 1-NN clearly outperforms RMT by 0.098 ± 0.10 (all other results statistically significant at a $p < 0.01$ level).

All models struggle to generalise; in particular, the RMT-based methods are clearly significantly worse than standard methods at generalising, with RMT and RMD unable to clearly outperform the baseline 1-NN model. Our results suggest that many high-performing models with good standard AUCs may not perform as well when tested in real life, since they may not be able to accurately predict far away from their

training set.

3.3.3 Attribution

To better understand why our models fail to generalise well, we now turn to the question of attribution, i.e. evaluating what our models are learning on a given dataset. We work with the toy datasets described previously so we can score our models' ability to attribute. For this analysis, we focused on the four highest-performing models as evaluated previously: logistic, random forest, Naive Bayes, and RMT + logistic. Unsurprisingly, our models, especially logistic and random forest, perform outstandingly well on the toy datasets since the logic is entirely fragment-based and there is no noise. Fig. 3.3.4a demonstrates that many of our models achieve very high far-AUCs. We note that our toy datasets were significantly less clustered than the real data with most of the test molecules being far away from the training set, so the far-AUC and standard AUC are essentially equivalent here. See Section 3.B.3 and Fig. 3.B.13 for a distance-based breakdown of model performance on the toy datasets.

However, the normalized attribution scores of our models in Fig. 3.3.4 are not good. Even high-performing models generate attribution scores that span the entire range $[0, 1]$, suggesting that these models are learning something very different from the intended rule. We also do not see any consistency in which models perform better than others on different datasets, which suggests that the accuracy of the rule any given model learns is highly dataset-dependent.

In order to better understand this data, we split the attribution score into false positives and false negatives in Fig. 3.3.5. Here we see some overall model-related trends: in particular, logistic regression tends to identify fewer false positives and more false negatives, especially on the three-element datasets, while the other models tend to identify more false positives and fewer false negatives. As before, there is still large variance between datasets and between molecules for the same model, suggesting that the validity of the rule learned by any particular model is highly questionable.

We directly examine some of the attribution images produced by our analysis in Fig. 3.3.6. This section only displays a few attribution images for the toy dataset Three-Elem1; the others are provided in Figs. 3.B.1 - 3.B.3. We see that logistic regression tends to highlight relatively few atoms, often omitting fragments that are important,

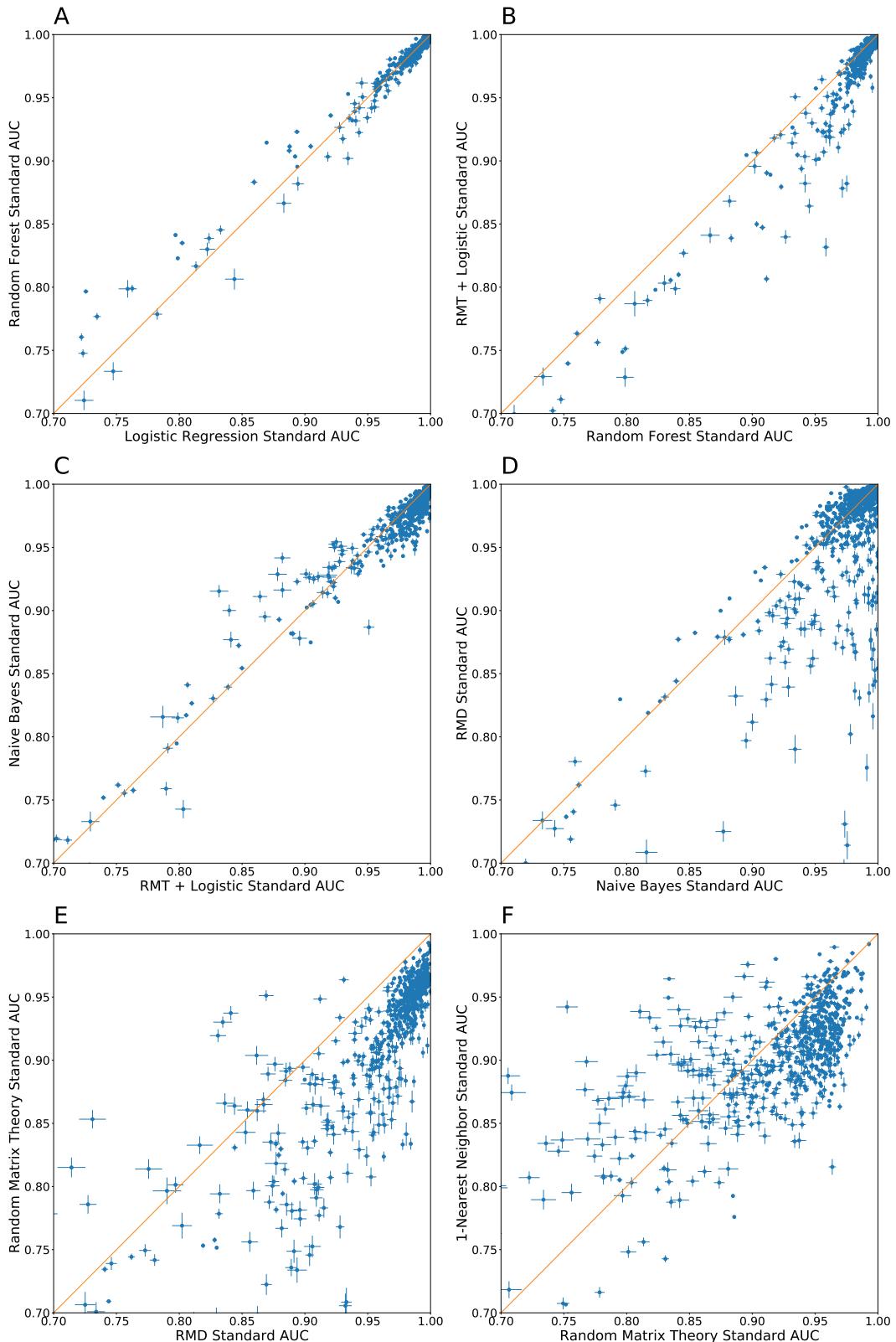


Figure 3.3.1: Comparison of Validation AUCs on ChEMBL Datasets for Various ML Models. Our most consistently high-performing models are logistic regression and random forest. Next, in order of performance, are RMT + logistic, Naive Bayes, Random Matrix Discriminant (RMD), Random Matrix Theory (RMT), and finally 1-nearest neighbor. Our results suggest that the random matrix theory framework does not aid in prediction since RMT-based models consistently underperform compared to standard models.

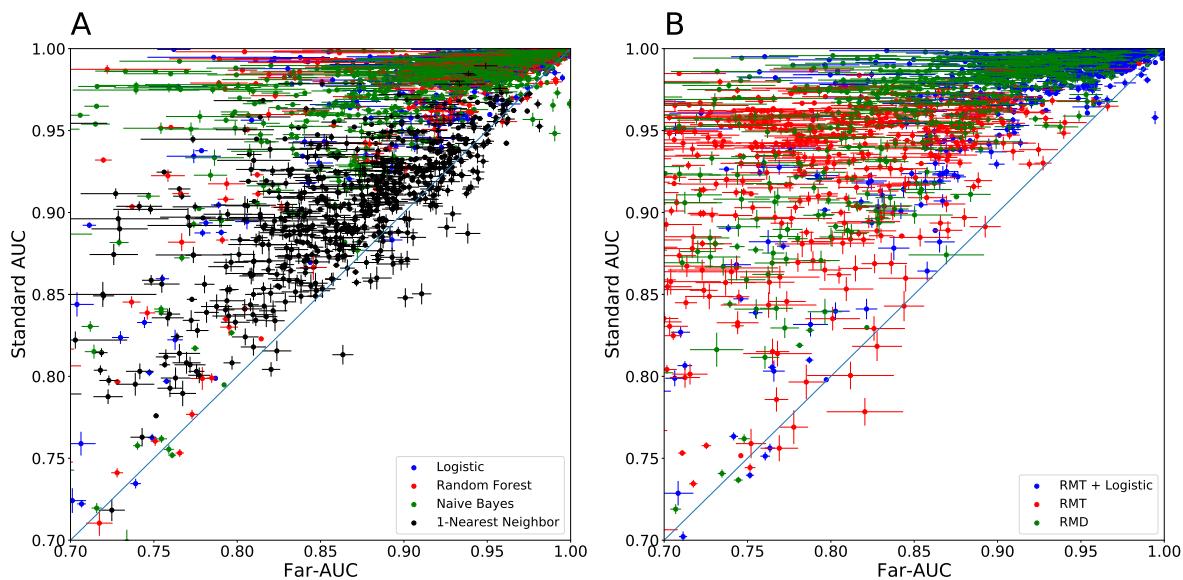


Figure 3.3.2: Comparison of Far-AUC and Standard AUC for (a) standard models and (b) RMT-based models. Our results indicate that most models have much worse far-AUCs, i.e. AUCs on a distant held-out test set, as compared to standard AUCs on a validation set; thus, most machine learning models fail to generalise well on protein/ligand binding data. RMT-based models are particularly bad, suggesting that the RMT framework does not aid generalisation overall.

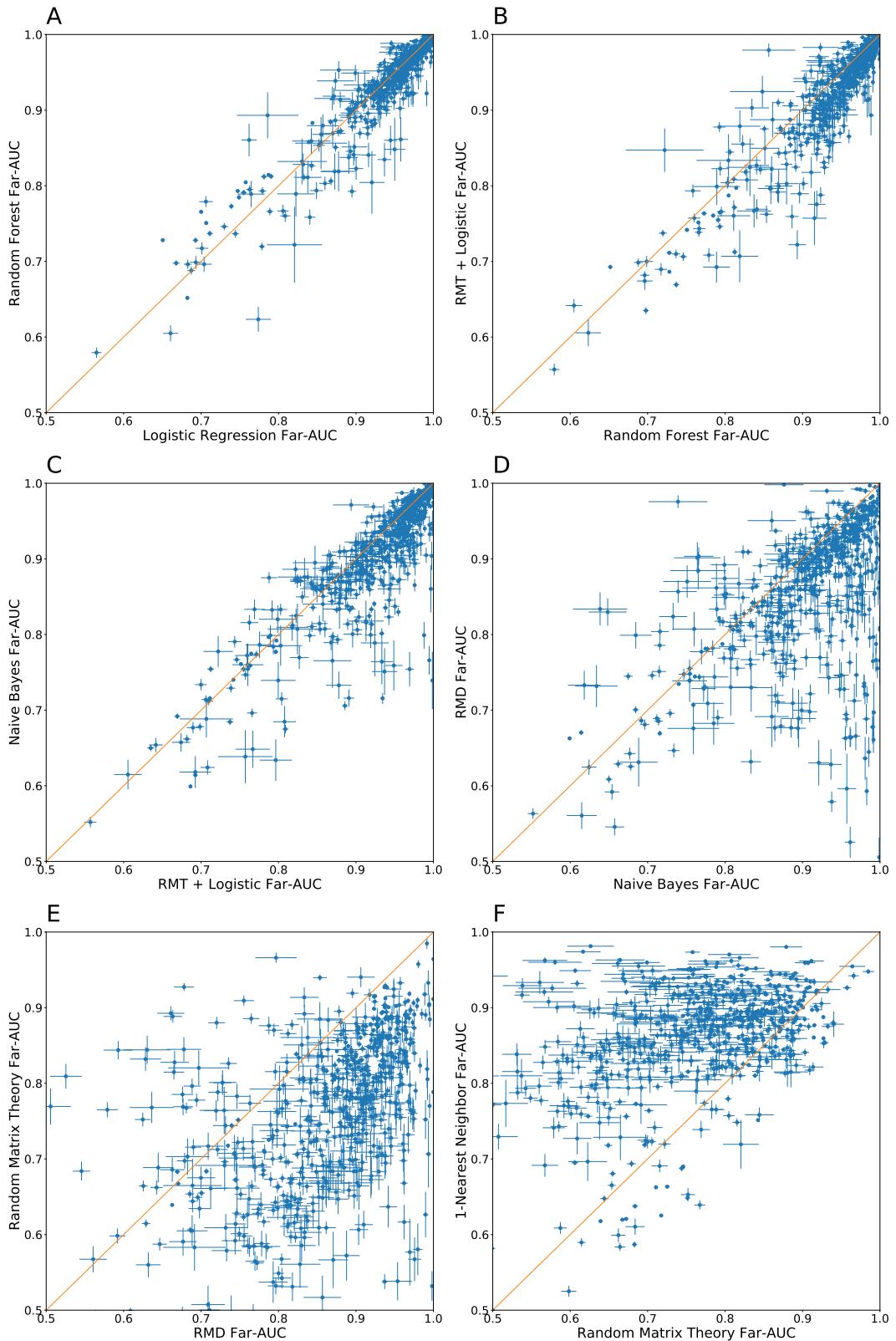


Figure 3.3.3: Comparison of Far-AUCs on ChEMBL Datasets for Various ML Models. Our results indicate that logistic regression is the best model at generalising on protein/ligand binding data, followed by random forest, RMT + logistic, Naive Bayes, RMD, 1-NN, and RMT, in that order. The RMT-based methods are particularly bad, suggesting that an RMT-based framework does not actually improve generalisation ability.

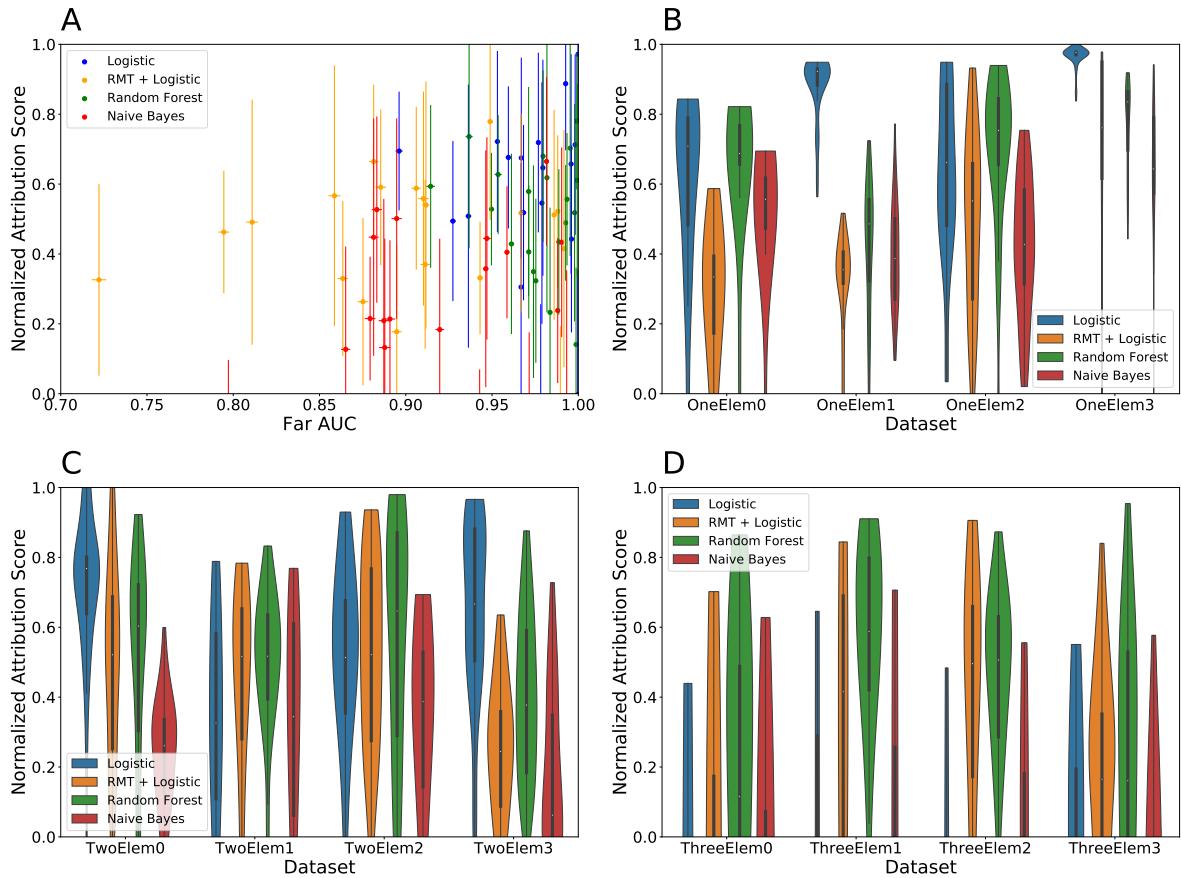


Figure 3.3.4: (a) Comparison of Attribution Scores to Far-AUC on Toy Datasets. The attribution score is a measure of how close the model's attribution is to the real rule for a particular molecule; the distributions shown here are distributions over molecules within a given dataset. Even models that generalise well have poor attribution scores, suggesting that they are learning the wrong rule. (b-d) Normalized Attribution Scores for Model on (b) One-Element Datasets, (c) Two-Element Datasets, and (d) Three-Element Datasets. There is little consistency in which models have high attribution scores, suggesting that no model can consistently learn the correct rule for a given dataset.

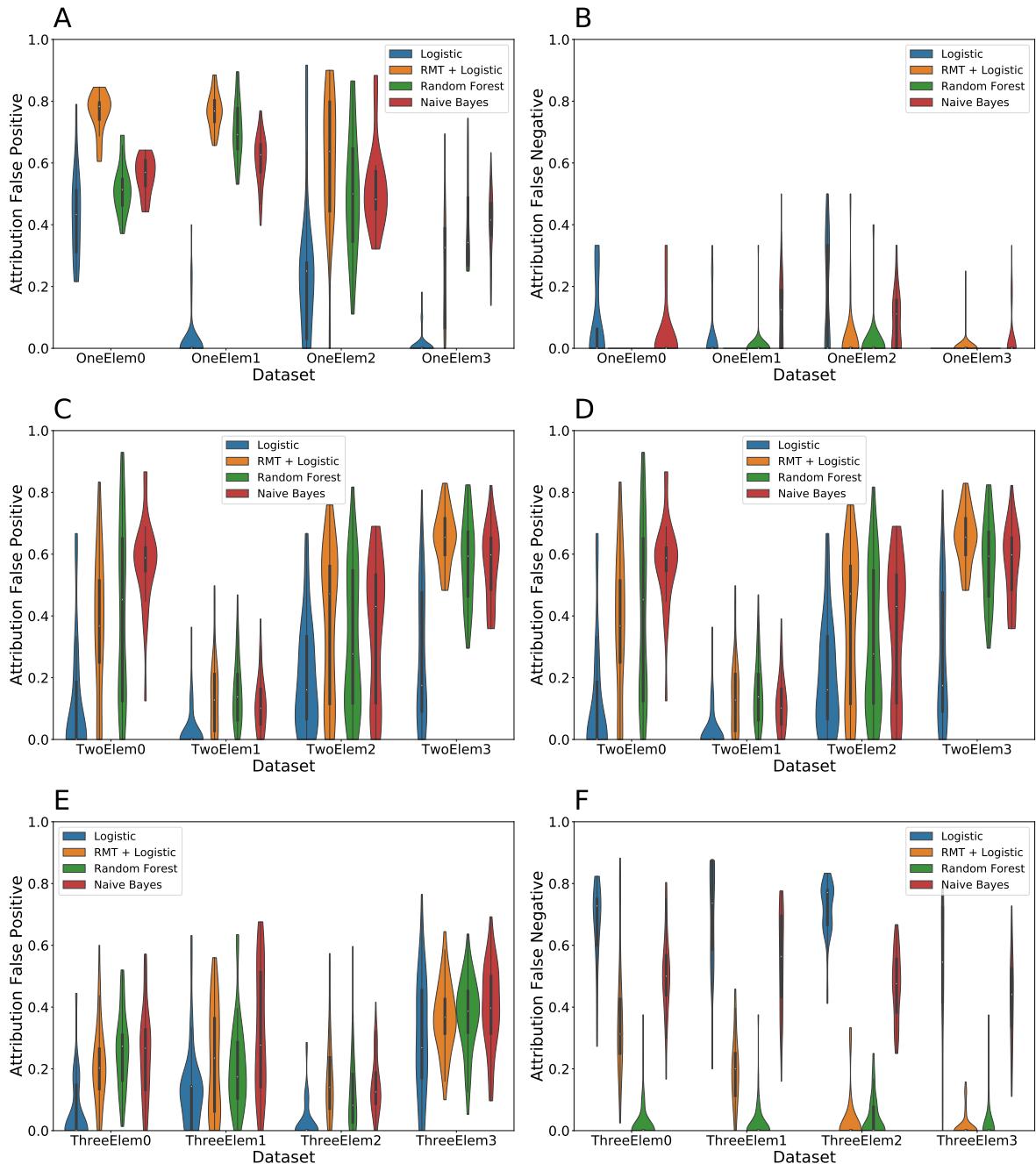


Figure 3.3.5: Attribution False Positive (a, c, e) and False Negatives (b, d, f) for One-Element (a-b), Two-Element (c-d), and Three-Element (e-f) Datasets. We observe higher rates of false negatives, atoms that the model misattributes as relevant, in logistic regression, especially on three-element datasets. The other models in contrast have higher rates of false positives.

while the other models tend to highlight irrelevant atoms; this agrees with the attribution false positive and false negative results shown previously.

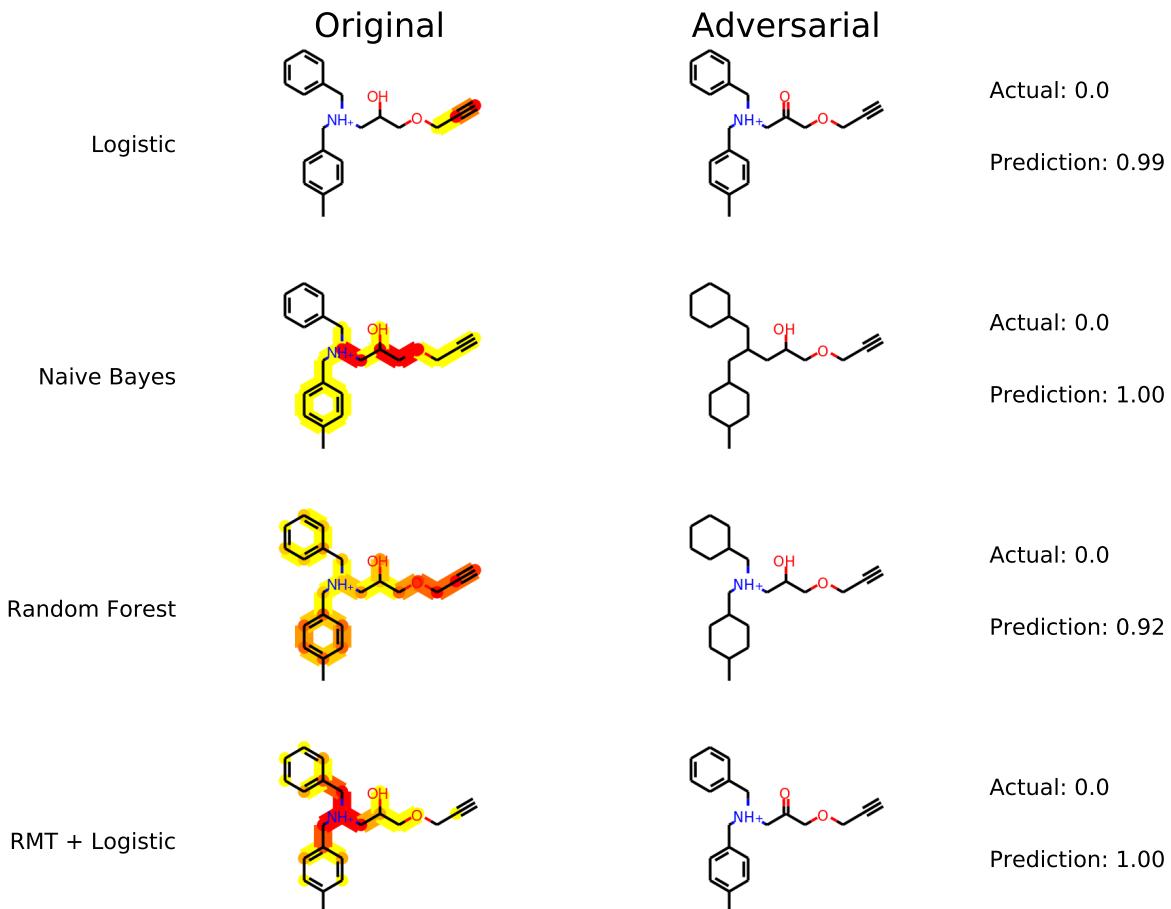


Figure 3.3.6: Adversarial Examples for Toy Dataset ThreeElem1. The logic for this dataset is benzene, alkyne, and hydroxyl. The left molecule is the attribution image for the specified model and molecule. The right image is an adversarial example constructed based on misattributions in the attribution image, usually by modifying an attribution false negative to generate a false negative result. The adversarial examples indicate that our attribution method is correctly identifying flaws in model performance.

Our attribution results call into question the validity of using these models to physically interpret predictions or to garner medicinal chemistry insight for further refinement of predictions. In order to further establish that our attribution results did correspond to model performance in some way, we generated adversarial examples for each molecule; these are shown in Fig. 3.3.6. These examples took advantage of mis-

attributed atoms. For example, if a model predicted a false negative attribution for a particular atom, we perturbed it to generate a related inactive molecule that the model would falsely predict as active. Similarly, if the model predicted false positive attribution for a particular atom, we perturbed it to generate a related active molecule that the model would falsely predict as inactive. The structure of SIS makes it significantly easier to generate adversarial molecules using attribution false negatives, so most examples fall in that category. We observe that it is relatively easy to generate an adversarial example for every misattribution, suggesting that our attribution scores do correspond to decreases in performance beyond simply hindering interpretability.

We will now examine some of the reasons behind misattributions, looking at both false negative and false positive results.

Attribution False Negatives: Fragment-Matched Decoys

To understand misattributed false negatives, we look first at the features most correlated with activity. Fig. 3.3.7 displays the top 10 such features for the dataset Three-Elem1, where the binding logic requires a benzene, alkyne, and hydroxyl. The first 4 features correspond to the alkyne, but we then see other functional groups show up instead of the benzene and the hydroxyl. The hydroxyl does show up at the 9th feature, but we do not see the benzene. The lack of benzene explains why many models fail to place high weight on it, as seen by the attribution examples provided in Fig. 3.3.6. We can see similar logic for the other toy datasets; see Figs. 3.B.7, 3.B.9, and 3.B.11.

It is difficult for our models to conclude that benzene is relevant for the binding logic simply because many inactive molecules in our dataset already have benzene. In order to aid our models, we can add fragment-matched decoys: decoys that have some, but not all, of the fragments required for binding, as detailed previously. Adding fragment-matched decoys immediately boosts benzene to the second most correlated feature in Fig. 3.3.8, suggesting that our algorithms may learn a better rule. Similar logic holds for the other toy datasets; see Figs. 3.B.8, 3.B.10, and 3.B.12, and Section 3.B.2.

In Fig. 3.3.9, we see the impact of adding fragment-matched decoys to our dataset. The graphs on the left demonstrate that the number of attribution false negatives dramatically decreases upon addition of fragment-matched decoys. This does result in

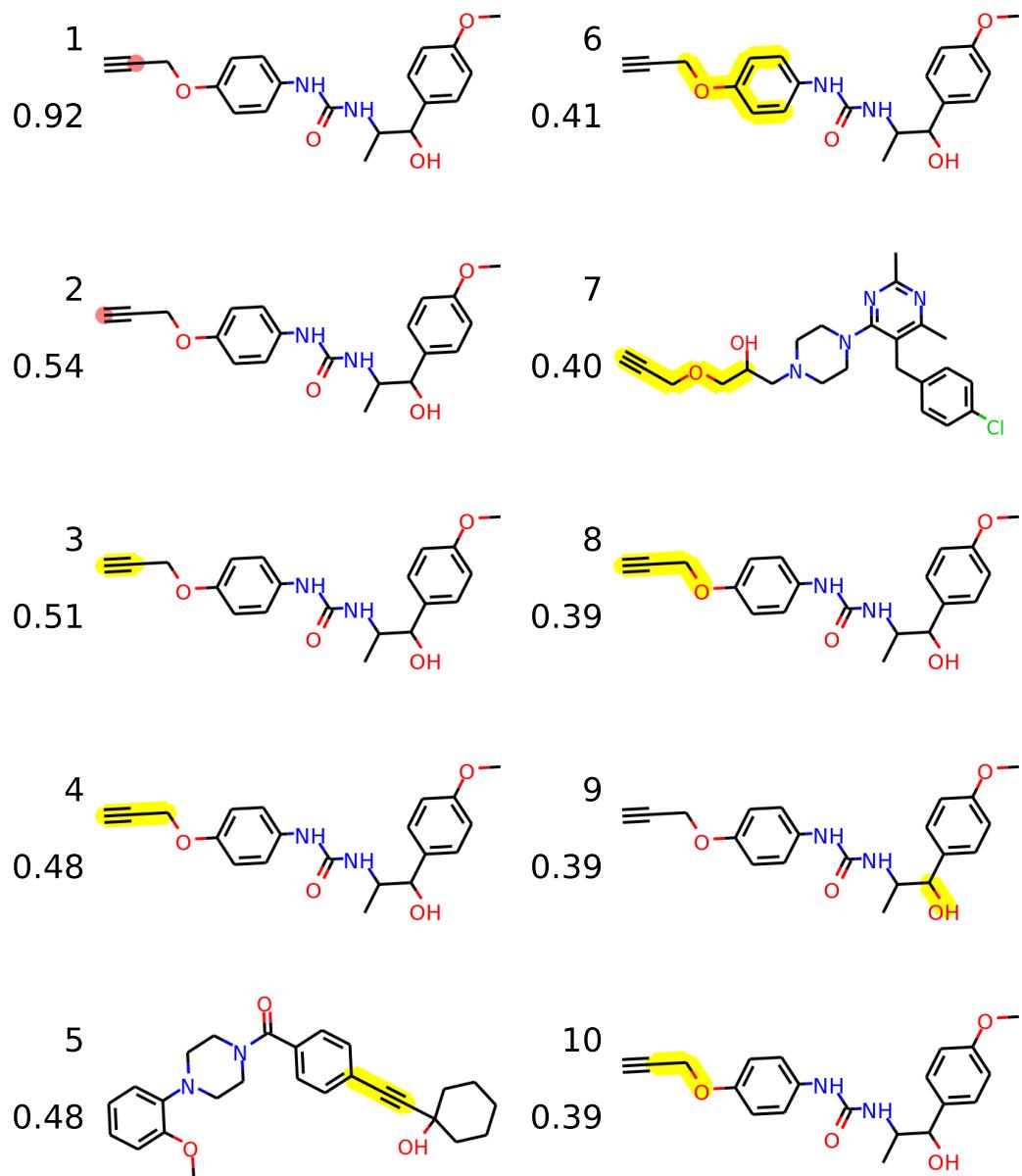


Figure 3.3.7: Top Features Correlated with Activity from Toy Dataset ThreeElem1. The features are ranked by correlation with activity with the correlations with activity listed below the rank. The logic for this dataset is benzene, alkyne, and hydroxyl. In the dataset without fragment-matched decoys, we see that alkyne ranks very highly, but the hydroxyl is much lower ranked and the benzene does not appear at all. This explains why models have trouble learning that the benzene is important for binding.

a significant increase in normalized attribution score, as shown in the graphs on the right. However, our models are still not perfect, as they still have plenty of attribution false positives.

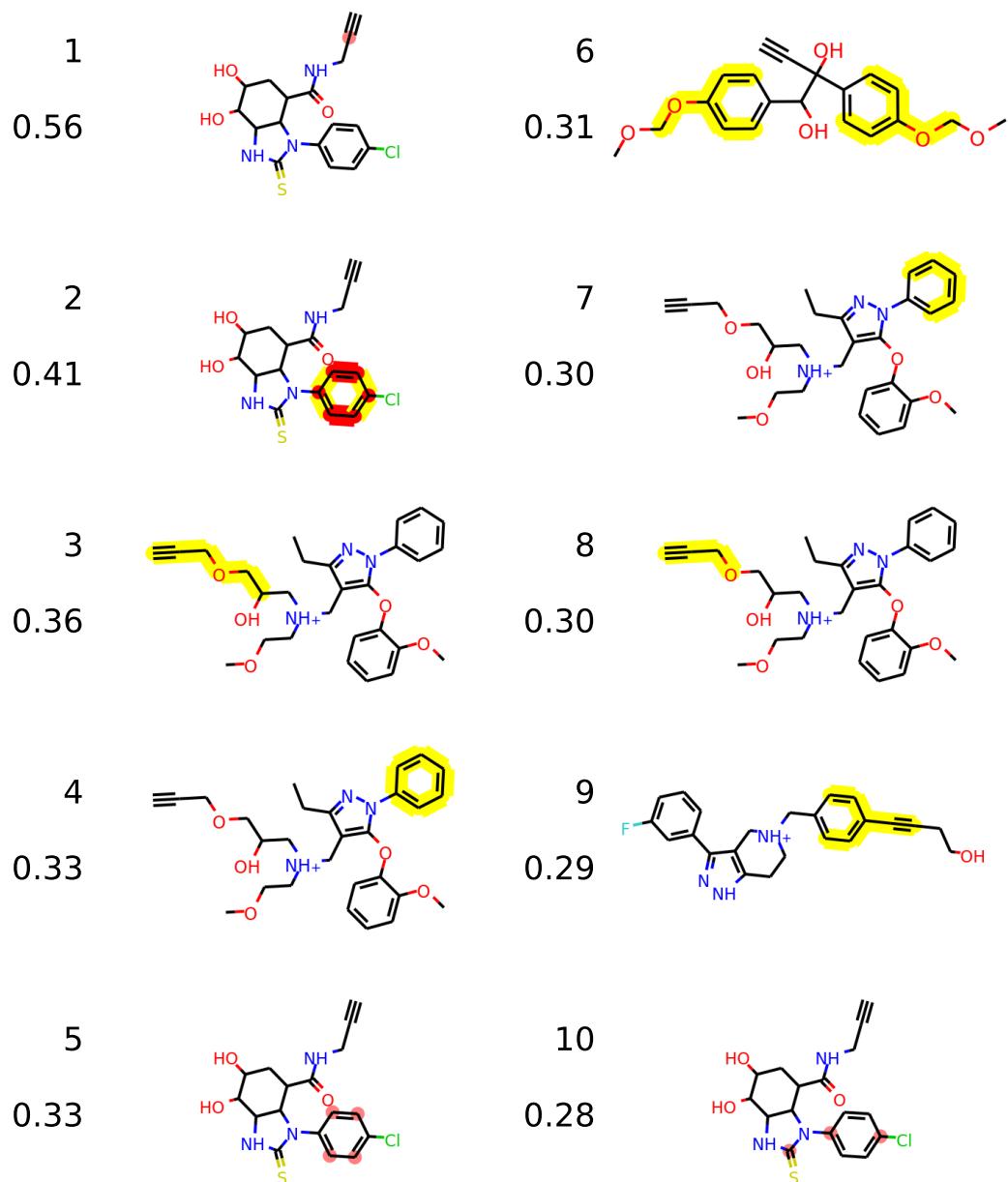


Figure 3.3.8: Top Features Correlated with Activity from Toy Dataset ThreeElem1 with Fragment-Matched Decoys. The features are ranked by correlation with activity with the correlations with activity listed below the rank. The logic for this dataset is benzene, alkyne, and hydroxyl. Adding fragment-matched decoys helps, but there are still a number of false positives, like the ether group that is the third-ranked feature.

Identifying adversarial examples for attribution false positives is trickier, since our attribution method does not tell us how to change a particular atom to make the model consider it no longer significant. Regardless, Fig. 3.3.10 demonstrates that, while more

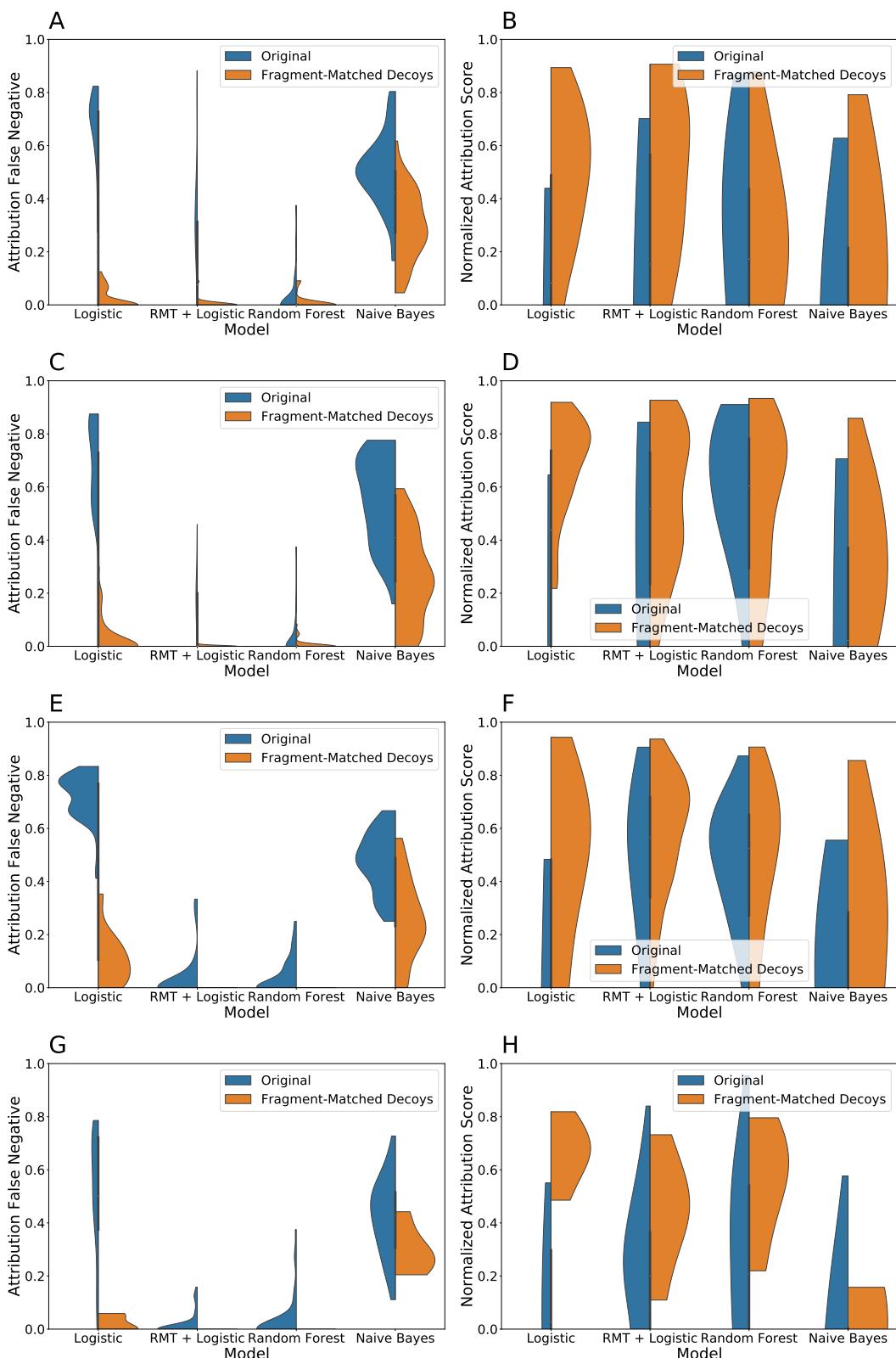


Figure 3.3.9: Effect of Adding Fragment-Matched Decoys on (a, c, e, g) attribution false negatives and (b, d, f, h) normalized attribution score for Three-Element Toy Datasets (Dataset 0 is (a-b), 1 is (c-d), 2 is (e-f), 3 is (g-h)). Adding fragment-matched decoys significantly decreases the rate of attribution false negatives consistently and does improve overall attribution scores. However, the attribution scores are still not perfect due to high rates of false positives.

difficult, it is still possible to generate large numbers of adversarial examples by hand; see Figs. 3.B.4 - 3.B.6 for the other toy datasets. Thus adding fragment-matched decoys is necessary but not sufficient to improve the overall attribution accuracy of our models.

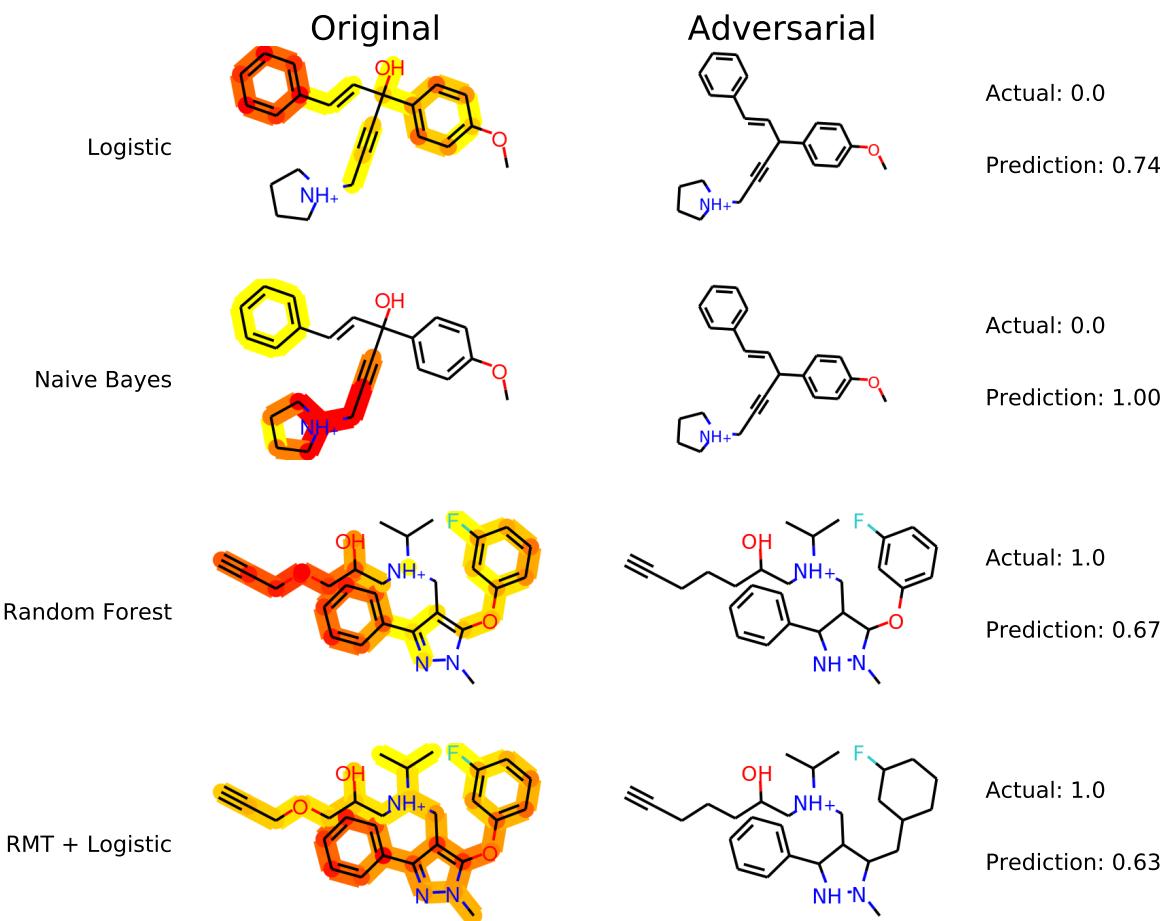


Figure 3.3.10: Adversarial Examples for Dataset ThreeElem1 with Fragment-Matched Decoys. The logic for this dataset is benzene, alkyne, and hydroxyl. The existence of these adversarial examples indicates that our models are still failing to learn the correct rule. False positive adversarial examples are more difficult to generate but still possible.

Further, the general procedure used in this step to add fragment-matched decoys is impossible in practice, because real datasets do not provide us with information about the binding logic of the protein in question. However, our results suggest that screening a variety of molecules with slightly different fragments is valuable for improving the attribution accuracy of protein/ligand binding models.

Attribution False Positives: Background Correlations

To understand misattributed false positives, we begin by examining the features most correlated with activity in Fig. 3.3.7. We see a number of ethers that connect alkynes to benzenes or to alcohols; the alkynes, benzenes, and alcohols are part of the binding logic, but the ether is not. Similarly, in Figs. 3.3.6 and 3.3.10, we see ethers incorrectly included in the attribution logic. This suggests that some of the attribution false positives are due to features highly correlated with activity that are not part of the binding logic.

Many of these features include multiple fragments of the binding logic: for example, the group (2-propyn-1-yloxy)benzene, which contains both a benzene and an alkyne joined by an ether, helps reinforce the presence of both the benzene and alkyne. However, this encourages our models to learn larger fragments which are correlated with but not part of the binding logic. We do not want our models learning that the benzene and alkyne must be found within the group (2-propyn-1-yloxy)benzene for the molecule to be considered active. Worse, often these larger groups have higher correlations than actual elements of the binding logic, increasing the likelihood that our models learn the wrong rule along the way.

Since the attribution false positives can be understood purely in terms of correlation with activity, they must fundamentally be a property of the dataset and not of the model in question. In particular, these larger features are likely highly correlated with features that are relevant for activity, like benzene or alkyne. Given our construction of the toy datasets, these inherent spurious correlations could only have arisen because they were already present in the background ZINC12 dataset. Thus we now look at the background correlation.

In order to compute the background correlation, due to high rates of bit collision across the entire ZINC12 dataset [28], we used unhashed ECFP6 fingerprints. We then took a random sample of 4000000 molecules out of the dataset to compute the correlation matrix. Fig. 3.3.11 compares the dataset correlation to the background ZINC12 dataset correlation for the 20 features most correlated with activity in each dataset. Due to the computational expense of computing the entire background correlation matrix, we only focused on these top 20 features for each dataset.

Our results indicate that the dataset correlations and the background correlations

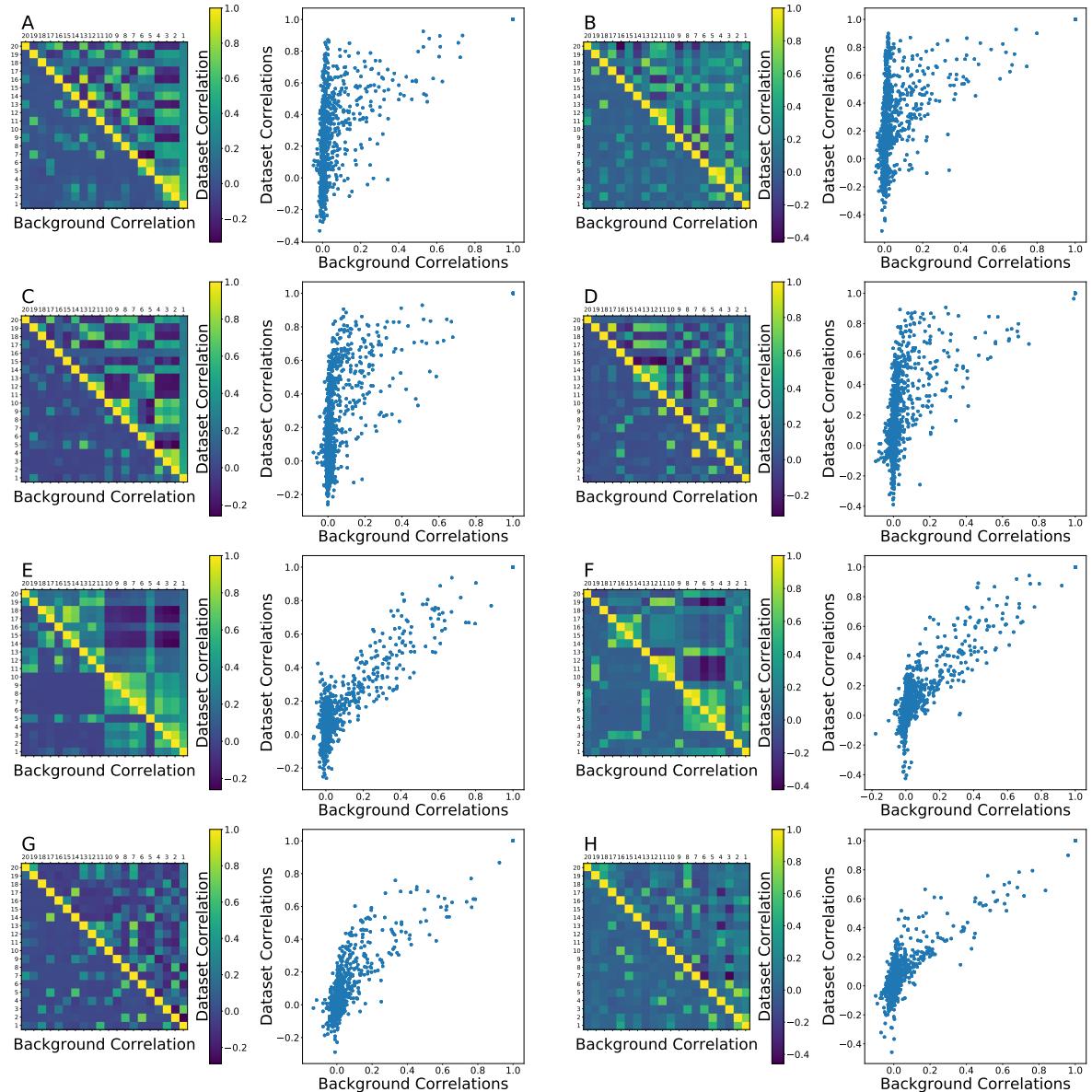


Figure 3.3.11: Correlation Matrices for the Background Dataset Compared with Toy Datasets. For each figure, left is a heatmap of the toy dataset correlation matrix in the top right and the background dataset correlation matrix in the bottom left. The right is a scatter plot of the correlations. (A, C, E, G) are original datasets, (B, D, F, H) include fragment-matched decoys. (A-B) are ThreeElem0, (C-D) ThreeElem1, (E-F) ThreeElem2, (G-H) ThreeElem3. We observe that the toy dataset correlation and background correlation are often closely related, with many spurious correlations in the toy datasets arising from the background. Correlations that are in the dataset but not in the background are useful for determining the correct binding logic for each dataset.

are closely related. A number of spurious correlations in the dataset are due to similar correlations in the background; this generates the cloud of points surrounding

the $y = x$ line in the scatterplots of Fig. 3.3.11. Other correlations appear only in the dataset and are not present in the background; this generates the cloud of points surrounding the y -axis in the scatterplots of Fig. 3.3.11. The goal of our models should be to distinguish between the spurious correlations arising from the background and the legitimate correlations that hint at the actual binding logic.

Successfully distinguishing between spurious and legitimate correlations does help uncover the correlations that correspond directly to the binding logic. We normalized the correlations by dividing the dataset correlation by the background correlation and looked at the pairs of features that had the highest mutual correlation for the dataset ThreeElem1 (without fragment-matched decoys). The top correlation was between features 2 and 17, which includes both an alkyne and a hydroxyl. The second correlation was between features 9 and 19, which includes both a hydroxyl and a benzene. The third correlation was between 18 and 4, which also corresponds to both an alkyne and a hydroxyl. The fourth correlation was between features 2 and 13, which corresponds to an alkyne and a benzene. Thus for our toy datasets correlations that occur much more strongly in the toy dataset than in the background do directly correspond to the binding logic used to build the dataset.

Comparison to Real Datasets

In order to verify that attribution results on our toy datasets are somewhat similar to those on real datasets, we examined comparative attribution scores. As mentioned previously, it is impossible to compute a real attribution score on a real dataset since we do not know the inherent binding logic. However, we can compare comparative attribution scores. Fig. 3.3.12 shows histograms of the comparative attribution scores on the toy data and on real datasets for high-performing and high-generalising models only (specifically a standard AUC and far-AUC of at least 0.99).

We see similar degrees of disagreement between the models on the toy datasets and on the real datasets. If two high-performing models like logistic regression and random forest disagree on the attribution logic, it suggests that many of the issues described above on the toy datasets likely plague attribution analysis on real datasets. In particular, we know that at least one of the two models must be learning the wrong logic, even though both performed and generalised very well. Our results on the toy

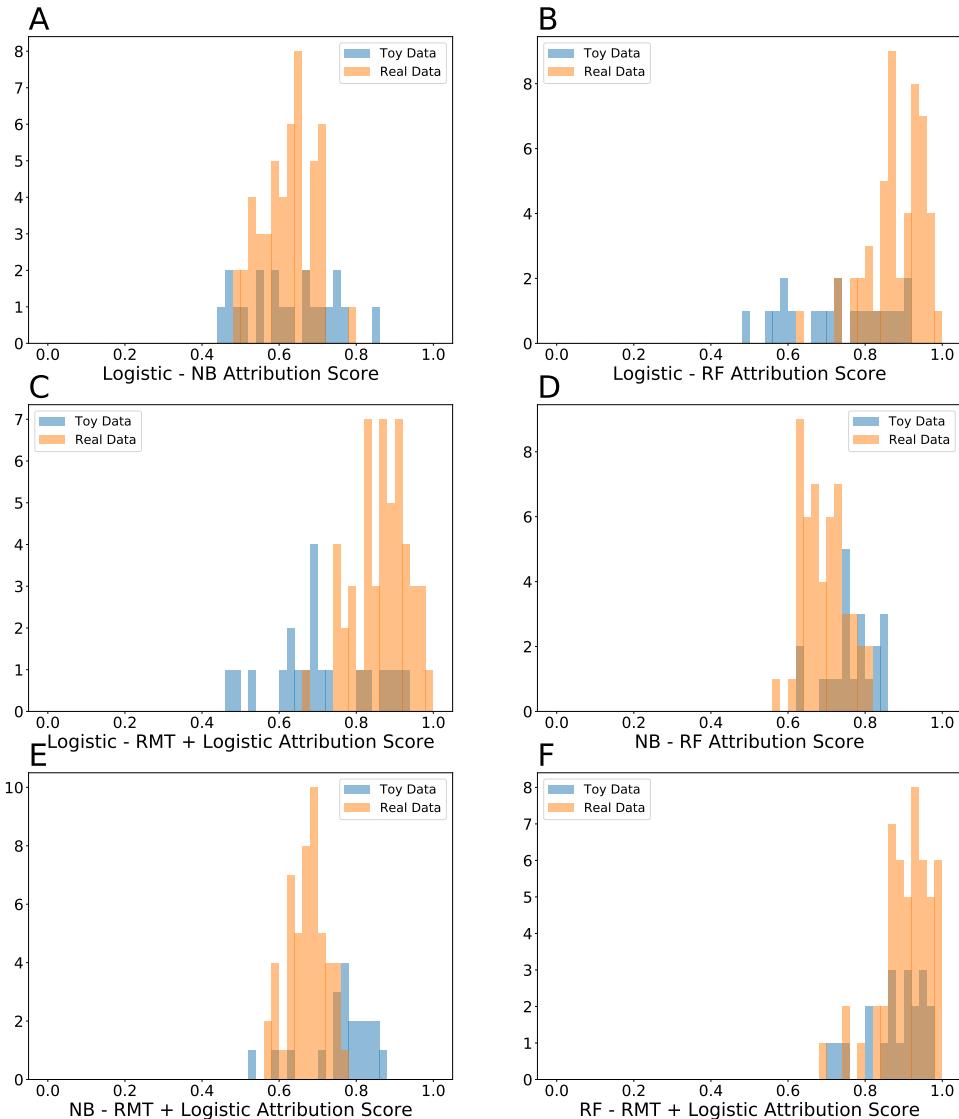


Figure 3.3.12: Histograms of Comparative Attribution Scores on Toy and Real Data between (a) logistic and Naive Bayes, (b) logistic and random forest, (c) logistic and RMT + logistic, (d) Naive Bayes and random forest, (e) Naive Bayes and RMT + logistic, and (f) random forest and RMT + logistic. Comparative attribution scores measure how closely two models agree on the attribution for a particular molecule. Our results show similar disagreement between models on both the real and the toy data, suggesting that our conclusions on the toy datasets would also hold on real datasets.

datasets suggest that it is likely both models have fallen victim to different spurious correlations in the data and are learning the wrong rule on the real data as well. It is also likely possible to generate adversarial examples similar to the ones generated above on the toy data.

3.4 Conclusions

Our results suggest a number of important cautionary notes about the success of fingerprint-based protein/ligand binding models. We have shown several examples of models that perform well which do not generalise well, suggesting that the validation AUC is not a particularly useful metric of performance and distance-based splits like the far-AUC should be used more often.

Further, all models, even those that perform and generalise well, may not learn the correct rule due to dataset bias. Explicit testing with toy datasets like those constructed in this chapter can help distinguish between models that can and cannot perform attribution correctly. Correct attribution is important both because scientists would like to use interpretable models to make medicinal chemistry decisions later in the drug development process and because incorrect attribution can allow relatively easy generation of adversarial examples that drastically reduce model performance. Prior to using any model for either of these purposes, it should be tested on toy datasets to ensure that it does consistently learn the correct rule.

In particular, contrary to previous claims, the RMT-based framework that was claimed to aid in generalisation and attribution at present is unable to outperform standard models. Distinguishing between signal and noise in the dataset is a good idea, but the implementation needs further research before it can be used in practice.

Our analysis of the reasons behind attribution false positives and false negatives suggests that the primary reason behind many of them is bias inherent in the dataset. We demonstrated that attribution false negatives can be mitigated by adding fragment-matched decoys. A corresponding approach on real datasets would be to identify inactive molecules with similar fragments as actives and add those to the training set. This may worsen performance on validation sets but improve attribution and generalisation, just as property-matched decoys can aid models based on physico-chemical

Chapter 3 Generalisation and Attribution

features. Collecting this data would require a more diverse screening library that contained many molecules with different fragments and in particular better selection of inactives.

Understanding attribution false positives is more difficult. We have shown that they largely originate from spurious correlations already present in the background data that cause various fragments to be incorrectly correlated with activity in the toy dataset. Thus the key to understanding false positives is to examine the correlation structure of the background dataset. There are two potential approaches here: change the background dataset, i.e. the screening library, to eliminate these correlations, or adjust our models so they account for those correlations in training.

The second approach is likely the simpler one. We have demonstrated that normalizing by dividing by the background correlation correctly identifies the binding logic in a toy dataset. It remains to develop models that accurately account for this background correlation.

The first approach is slightly more complicated. There are practical difficulties associated with running a high-throughput screen on a different screening library possibly including expensive or difficult-to-synthesize molecules, so all molecules in any new proposed screening library would have to remain relatively easy to synthesize. Further, even a perfectly diverse chemical dataset would possess a base-rate correlation due to the correlation in fingerprint bits from fragments that are subfragments of others. We must compute this base-rate correlation and distinguish it from the correlation due to dataset bias. Then, we must design a fragment library that only possesses the base-rate correlation. This will allow us to eliminate spurious correlations in our dataset and minimize attribution-related errors.

Appendix: Supporting Information

3.A Supplementary Methods

3.A.1 Toy Dataset Construction

The exact SMARTS string used to match each of the fragments for the toy datasets may be found in Table 3.A.1.

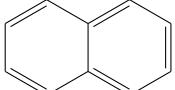
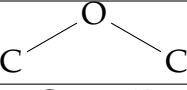
| Fragment | SMARTS Code |
|---|--|
| $\text{C}=\text{C}$ | $\text{C}=\text{C}$ |
|  | $\text{c}1\text{cccc}\text{c}1$ |
|  | $\text{c}12\text{ccccc}\text{c}1\text{ccc}\text{c}2$ |
| $\text{C}-\text{NH}_2$ | $\text{C}[\text{NH}2]$ |
|  | COC |
| $\text{C}-\text{F}$ | CF |
| $\text{C}-\text{OH}$ | $\text{C}[\text{OH}]$ |
| $\text{C}=\text{O}$ | $\text{C}=\text{O}$ |
| $\text{C}\equiv\text{C}$ | $\text{C}\#\text{C}$ |

Table 3.A.1: SMARTS Code Corresponding to Each Fragment Used in Toy Datasets.

3.A.2 Random Matrix Theory Based Methods

This section provides a description of the random-matrix-theory-based methods tested in this work.

1. Random Matrix Theory (RMT). We followed the exact algorithm from the original RMT paper [40]. Specifically, given a dataset of n_+ actives each with $p = 2048$

fingerprint bits, we construct the $n_+ \times p$ active data matrix A_+ . We eliminate repeated columns from A_+ as well as columns that are constant (either 1 or 0) and normalize to z -scores. We then compute the correlation matrix $C_+ = \frac{A_+^T A_+}{N}$.

We then compute the eigenvalues and eigenvectors of this correlation matrix and select eigenvectors corresponding to eigenvalues $\lambda_+ > \left(1 + \sqrt{\frac{p}{n_+}}\right)^2$, the Marchenko-Pastur bound. These eigenvectors span a hyperplane V_+ . For a given unknown ligand \mathbf{u} (in the scaled coordinates), we may now compute its projection onto V_+ , \mathbf{u}_{V_+} . A ligand is predicted to be active if $\|\mathbf{u} - \mathbf{u}_{V_+}\| < \epsilon$ for a given threshold ϵ , which may be varied to obtain an ROC curve.

2. Random Matrix Discriminant (RMD). We followed the exact algorithm from the original RMD paper [41]. This also accounts for inactives in the algorithm above. Specifically, we also create an inactive data matrix A_- of the n_- known inactives and compute the normalized correlation matrix C_- in the same fashion. After filtering using the Marchenko-Pastur distribution, we obtain a second hyperplane V_- and a second projection \mathbf{u}_{V_-} . A ligand is predicted to be active if $\|\mathbf{u} - \mathbf{u}_{V_+}\| < \|\mathbf{u} - \mathbf{u}_{V_-}\| + \epsilon$ for a given threshold ϵ .
3. RMT + Logistic. We may also use the random matrix theory method as a feature selection method. In this case, we use the combined data matrix A , including both actives and inactives. Applying the same steps as above, we obtain eigenvectors corresponding to eigenvalues above the Marchenko-Pastur threshold. We then use the projection of our unknown ligand \mathbf{u} onto each individual eigenvector as a new feature set to train a model, in this case logistic regression, on.

3.B Supplementary Results

3.B.1 Additional Adversarial Examples

We provide additional adversarial examples for the other three-element toy datasets, as well as the datasets with fragment-matched decoys. The adversarial examples for the original datasets can be found in Figs. 3.B.1, 3.B.2 and 3.B.3. The adversarial exam-

3.B Supplementary Results

ples for the datasets with fragment-matched decoys can be found in Figs. 3.B.4, 3.B.5, and 3.B.6.

It is relatively easy to use our attribution method to manually generate adversarial examples for each individual molecule. These results suggest that our attribution method does correspond to misclassifications by our models. Generating adversarial examples becomes more difficult for datasets with fragment-matched decoys due to the necessity of using attribution false positives. However, we are still able to consistently generate adversarial examples for each model.

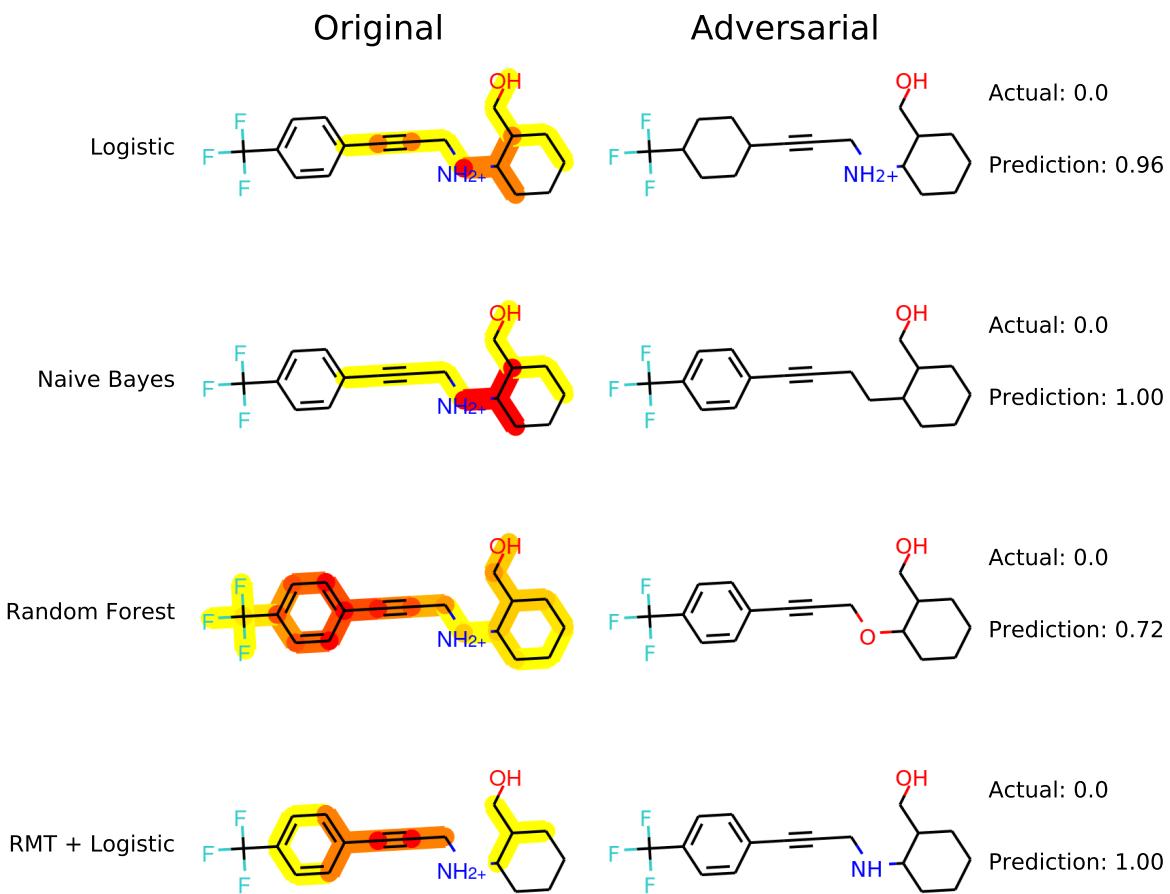


Figure 3.B.1: Adversarial Examples for Toy Dataset ThreeElem0. The logic for this dataset is benzene and alkyne and amino. The existence of these adversarial examples suggests that our attribution method is correctly identifying flaws in model performance.

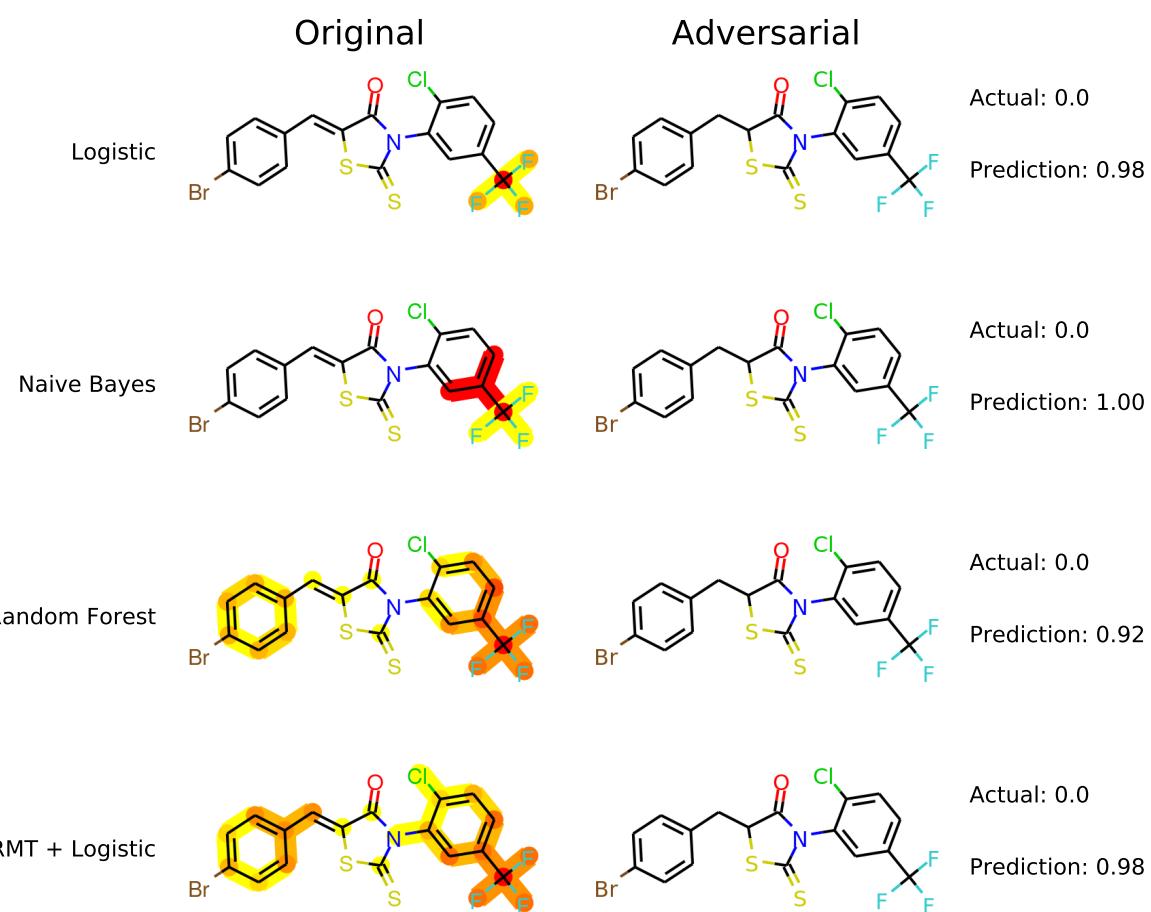


Figure 3.B.2: Adversarial Examples for Toy Dataset ThreeElem2. The logic for this dataset is fluorine and alkene and benzene. The existence of these adversarial examples suggests that our attribution method is correctly identifying flaws in model performance.

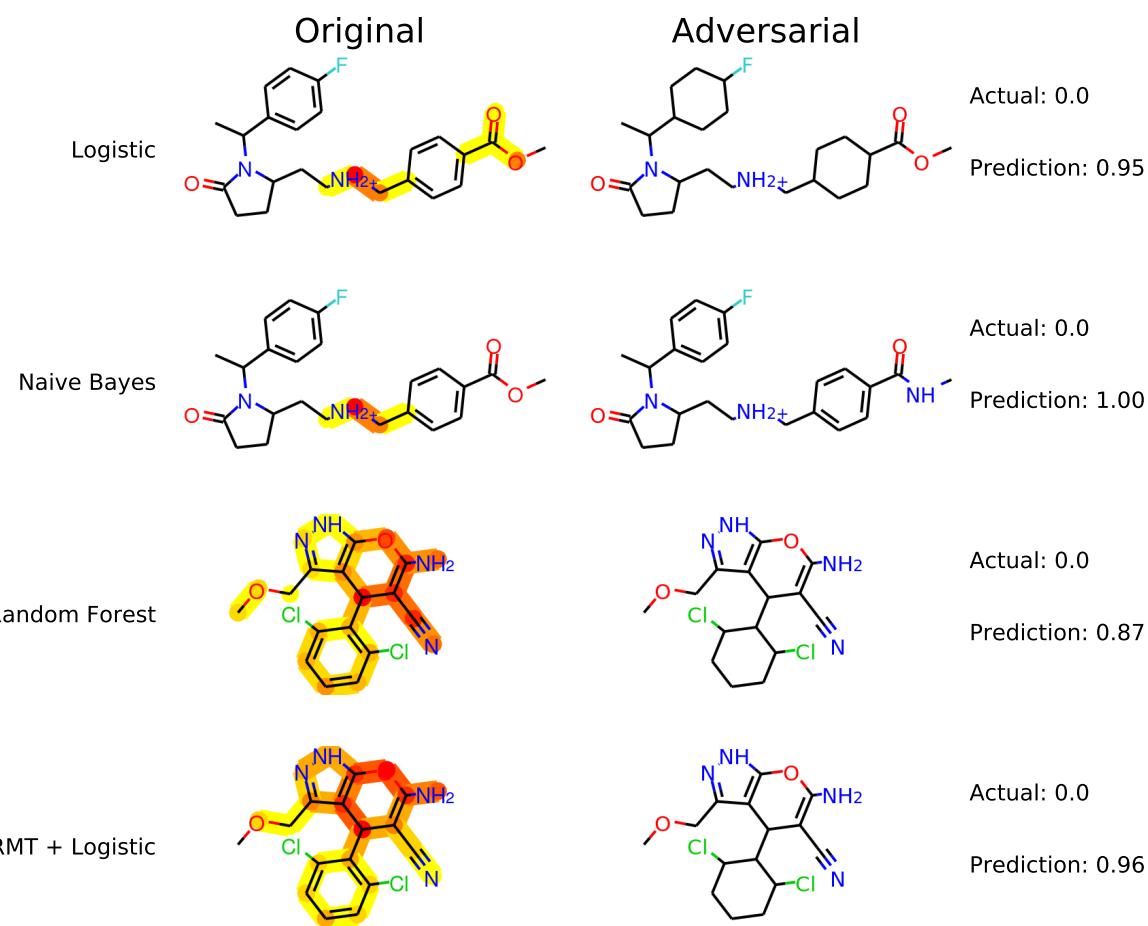


Figure 3.B.3: Adversarial Examples for Toy Dataset ThreeElem3. The logic for this dataset is benzene and ether and amino. The existence of these adversarial examples suggests that our attribution method is correctly identifying flaws in model performance.

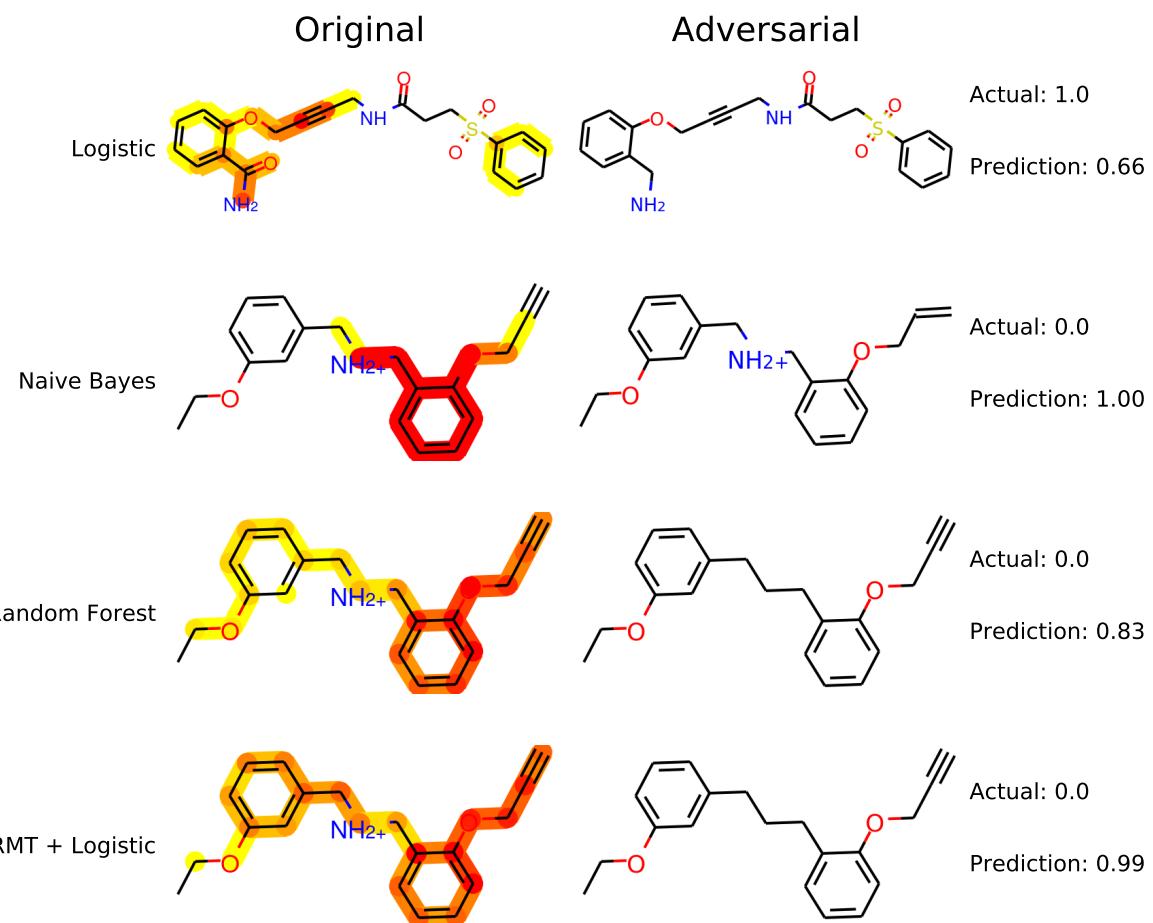


Figure 3.B.4: Adversarial Examples for Toy Dataset ThreeElem0 with Fragment-Matched Decoys. The logic for this dataset is benzene and alkyne and amino. These adversarial examples are more difficult to generate due to the inclusion of fragment-match decoys, but our models are still not consistently learning the correct rule.

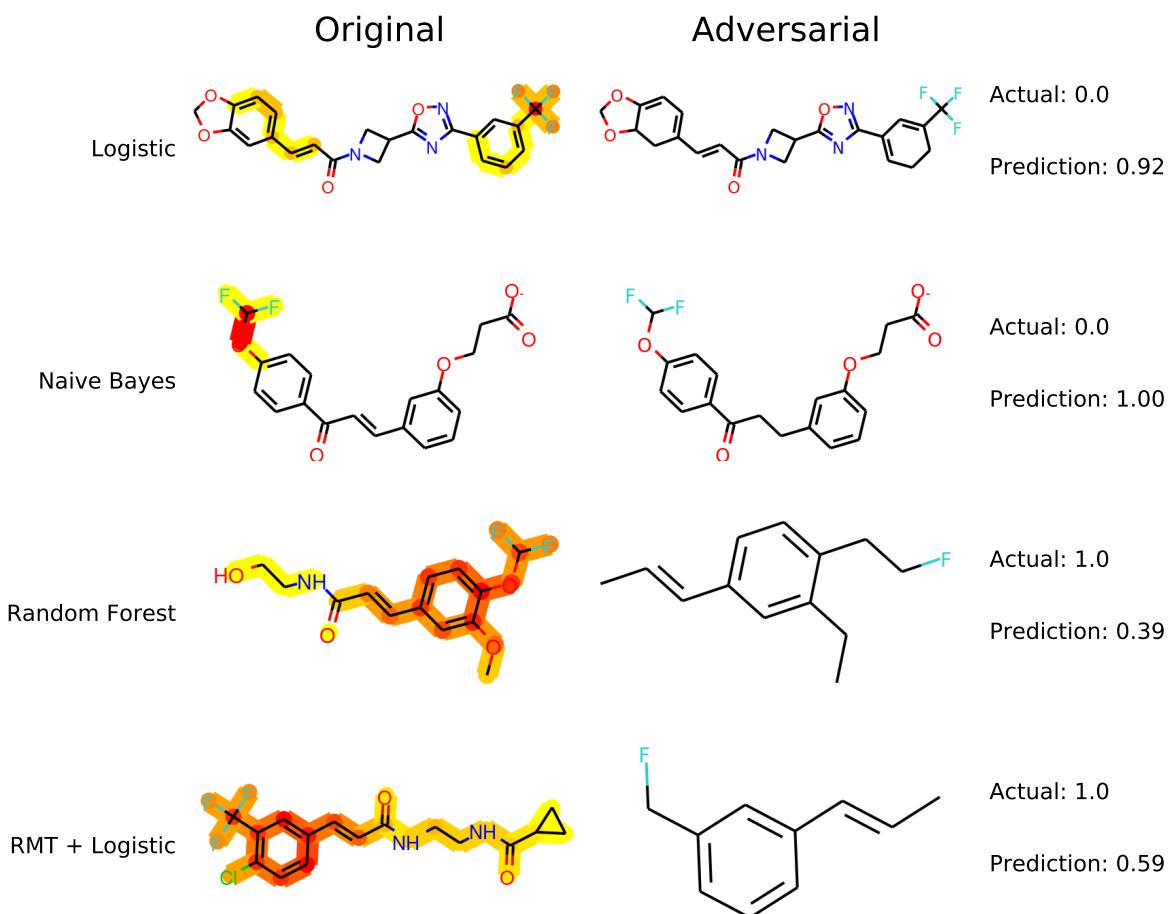


Figure 3.B.5: Adversarial Examples for Toy Dataset ThreeElem2 with Fragment-Matched Decoys. The logic for this dataset is fluorine and alkene and benzene. These adversarial examples are more difficult to generate due to the inclusion of fragment-match decoys, but our models are still not consistently learning the correct rule.

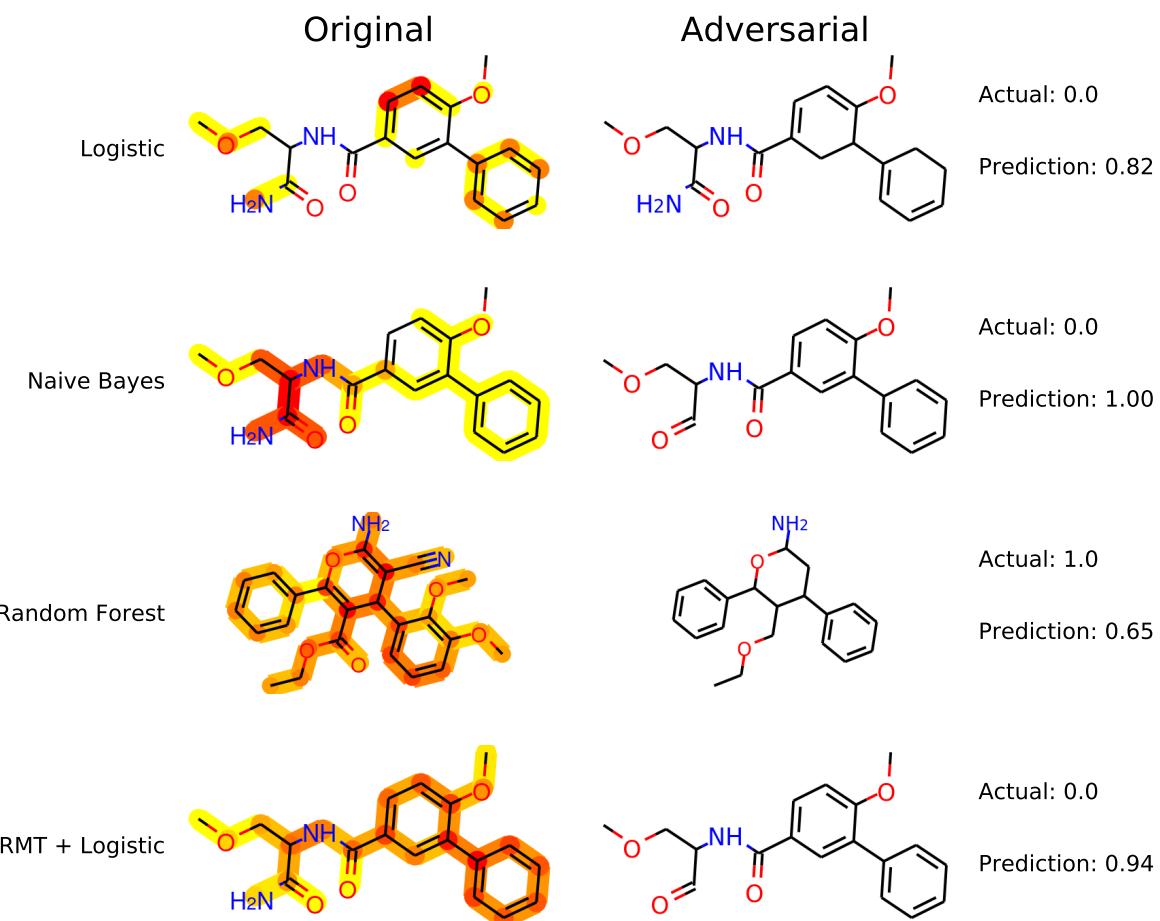


Figure 3.B.6: Adversarial Examples for Toy Dataset ThreeElem3 with Fragment-Matched Decoys. The logic for this dataset is benzene and ether and amino. These adversarial examples are more difficult to generate due to the inclusion of fragment-match decoys, but our models are still not consistently learning the correct rule.

3.B.2 Top Features Correlated with Activity for Other Datasets

We can also examine the claim that attribution false negatives partially arise from features incorrectly correlated with activity on other datasets. The top features for the datasets without fragment-matched decoys are in Figs. 3.B.7, 3.B.9, and 3.B.11 and for the datasets with fragment-matched decoys in Figs. 3.B.8, 3.B.10, and 3.B.12.

We see fairly consistently that some of the fragments in the logic do not appear in the top features list for the datasets without fragment-matched decoys. Benzene is particularly troublesome across all of the datasets. Adding fragment-matched decoys consistently helps models by increasing the ranking of features that are actually in the logic.

However, we also see spurious features: features that are not in the logic, but are still highly correlated due to correlations in the background. Many of these are a result of synthetic moieties used as building blocks that happen to contain one or more features found in the logic and are therefore highly correlated with activity. This problem is not resolved simply by adding fragment-matched decoys; as explained in the main text, looking at background correlation rates is key.

3.B.3 Toy Model AUC as a Function of Distance

In order to examine generalisation ability of our models on the toy datasets, we computed the AUC as a function of distance in Fig. 3.B.13. For this step, we bucketed the test set into 5 equal-size bins based on nearest-neighbor distance to the training set. We computed the AUC of our models on each of these smaller test sets, eliminating test sets with fewer than 5 actives or inactives. We generally see that our models perform worse as the distance from the training set increases, implying that our models are worse at generalising. Just as for the real data, we generally see that logistic regression and random forest are the best at generalising.

However, there is one important difference between the toy data and the real data: the toy data is significantly less clustered than the real data is. Our graphs in Fig. 3.B.13 indicate that the average distance in the smallest bucket is already generally greater than 0.4, whereas for the real data this was the cutoff we used for the far-AUC and most actives were below that cutoff. Thus the toy data test sets are farther from the

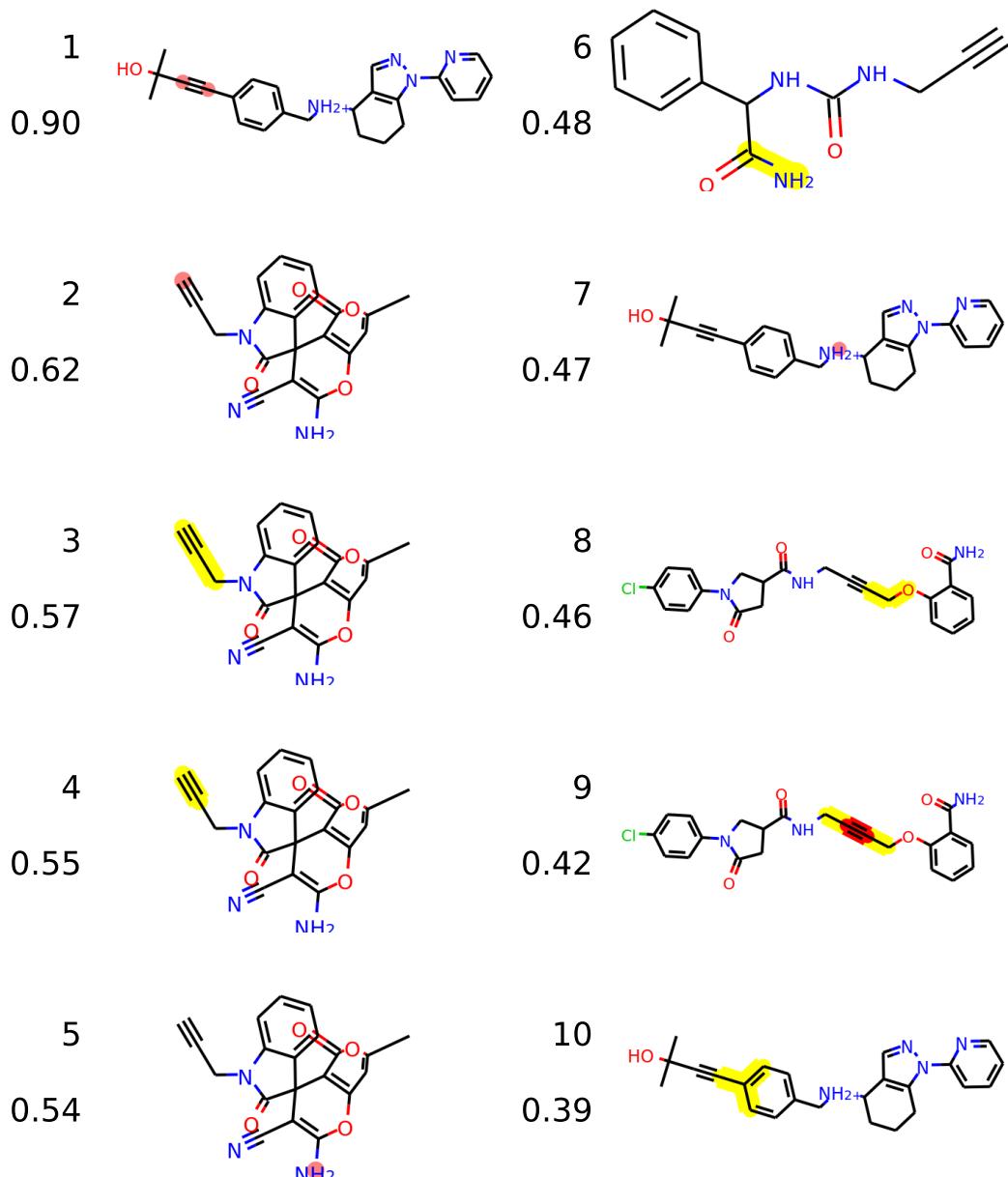


Figure 3.B.7: Top Features Correlated with Activity for Toy Dataset ThreeElem0. The logic for this dataset is benzene and alkyne and amino. We see alkyne and amino appear repeatedly, but benzene only shows up as a neighbor to the alkyne group, and an extraneous ether appears higher than the benzene.

training set than the real data is. This is why the far-AUC of the toy datasets is essentially identical to the standard-AUC. Given that our models perform well on the toy datasets' test sets, our attribution results suggest that even models that generalise reasonably well can fail to learn the correct binding logic.

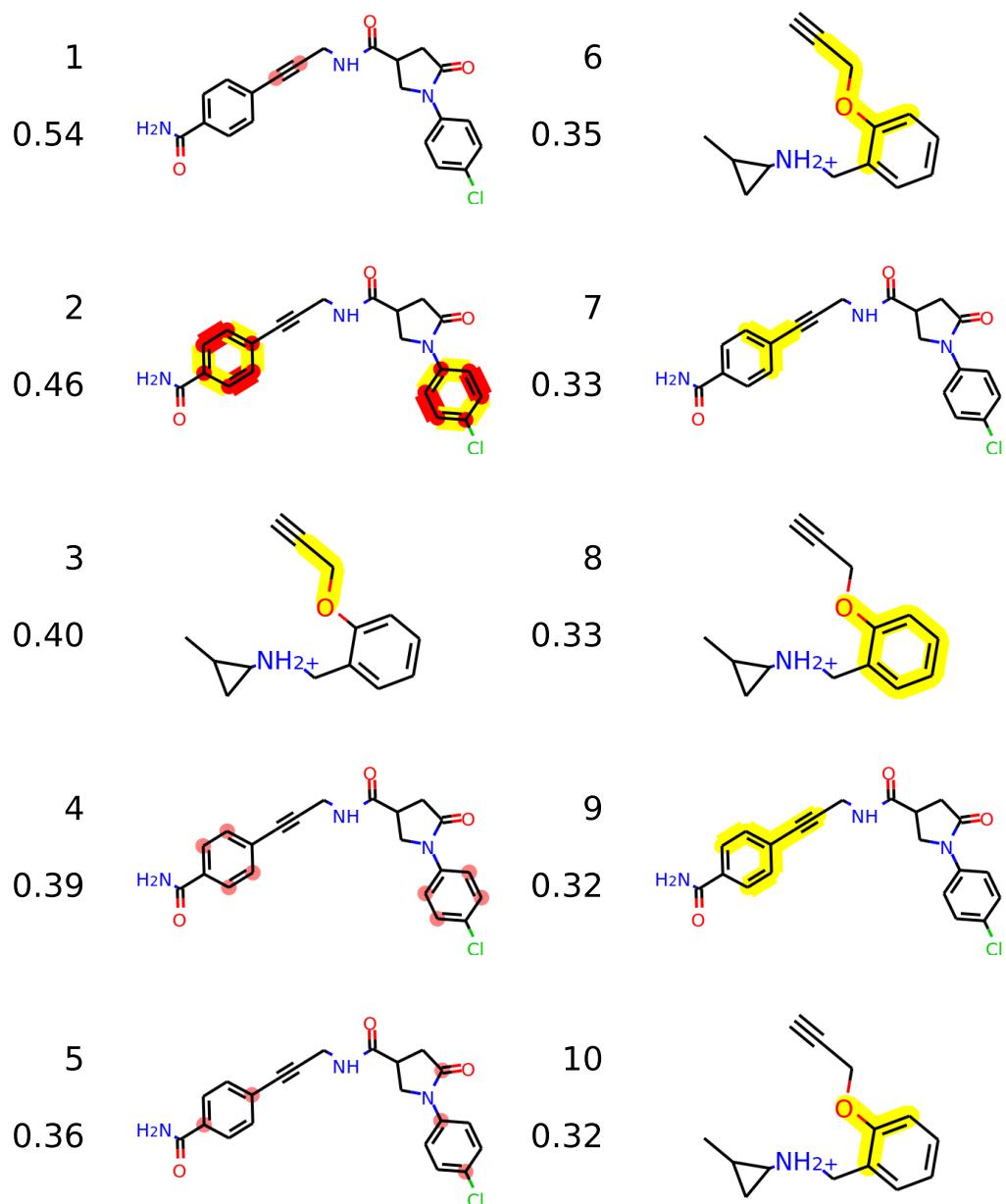


Figure 3.B.8: Top Features Correlated with Activity for Toy Dataset ThreeElem0 with Fragment-Matched Decoys. The logic for this dataset is benzene and alkyne and amino. We now see both benzene and alkyne rank fairly highly, but the same extraneous ether group is still present.

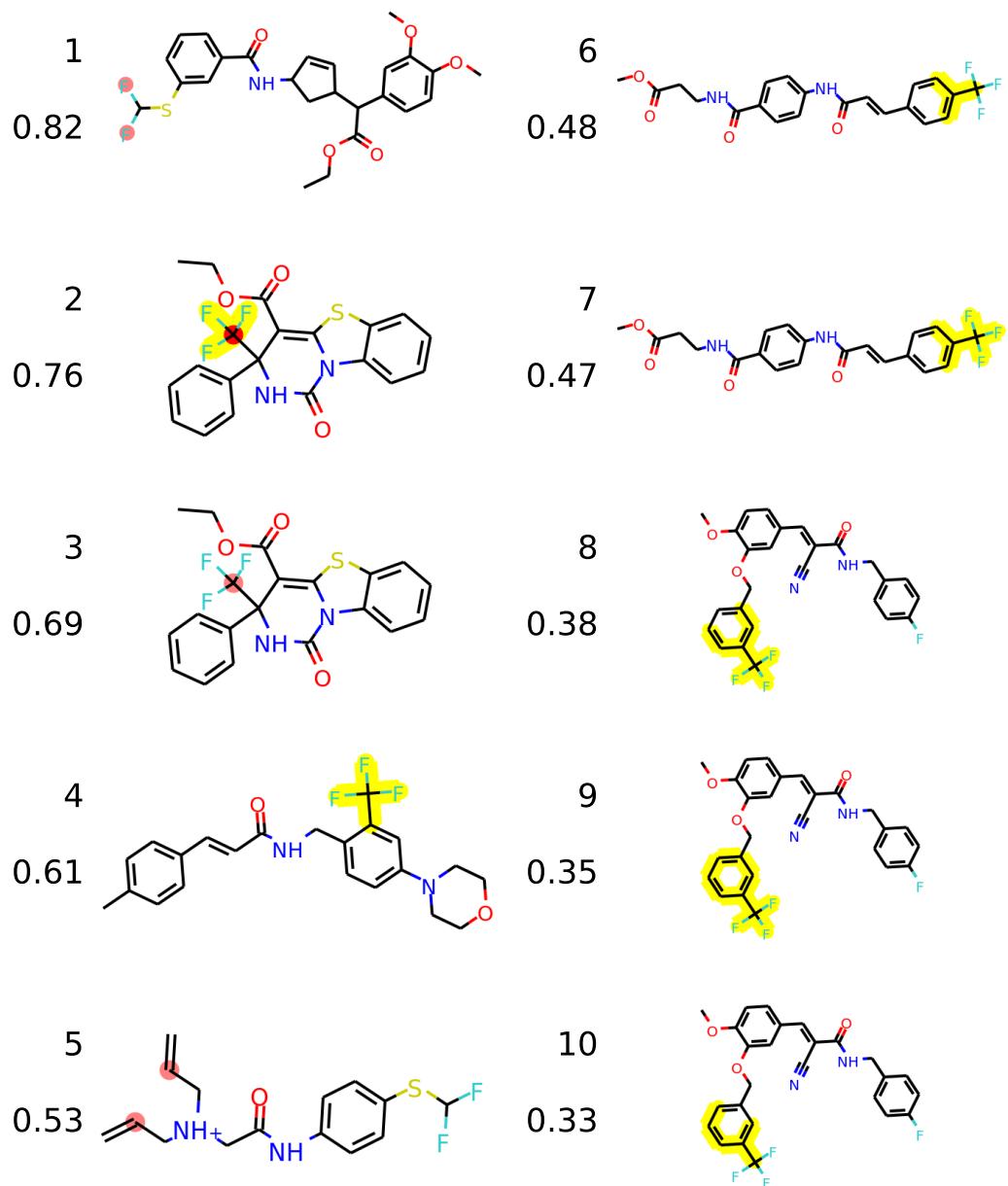


Figure 3.B.9: Top Features Correlated with Activity for Toy Dataset ThreeElem2. The logic for this dataset is fluorine and alkene and benzene. We see fluorine appear repeatedly and alkene appear once, but spurious CF_3 groups also show up, and benzene only appears when attached to a CF_3 .

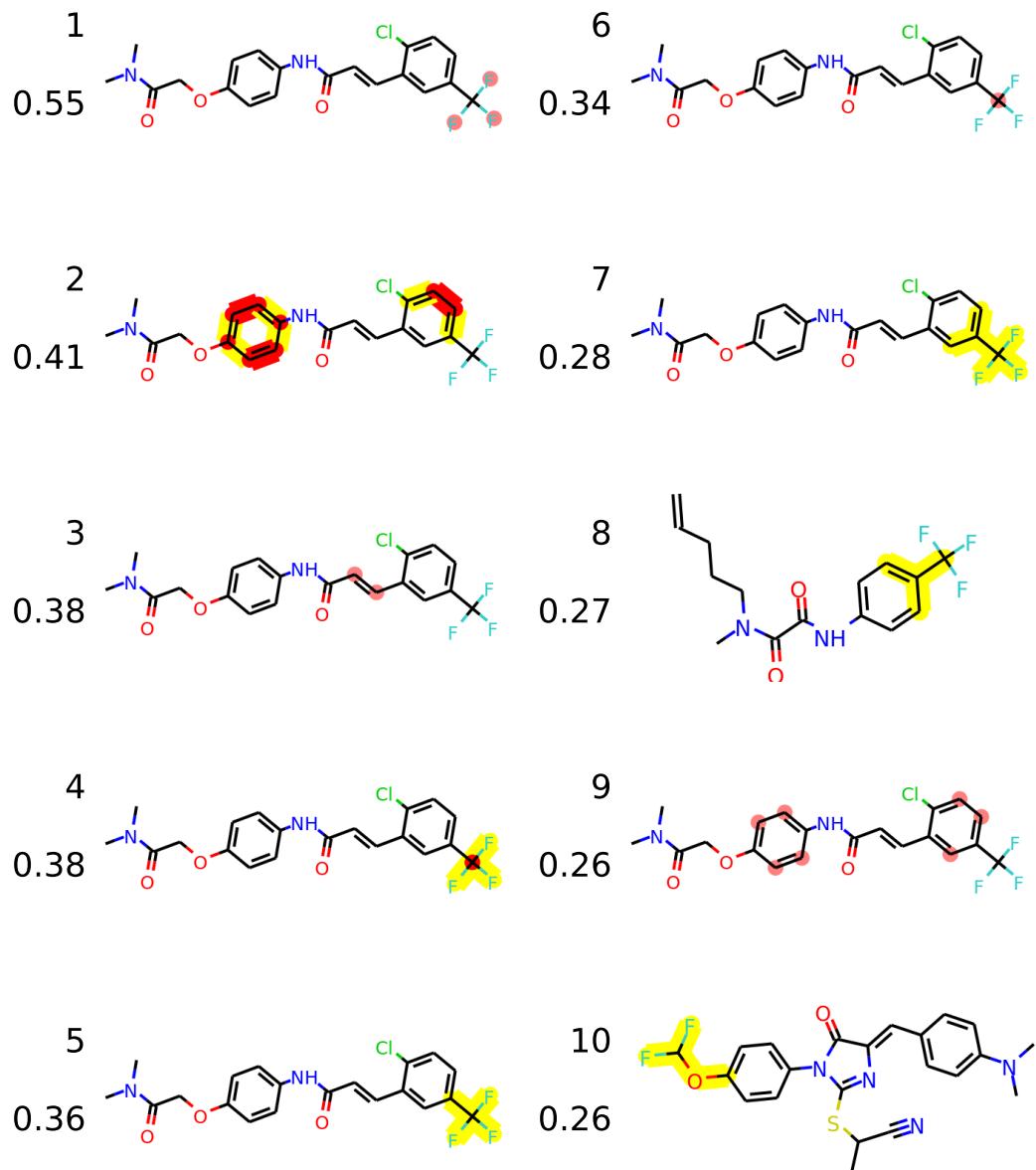


Figure 3.B.10: Top Features Correlated with Activity for Toy Dataset ThreeElem2 with Fragment-Matched Decoys. The logic for this dataset is fluorine and alkene and benzene. All three elements of the logic do appear, but we still see spurious highly correlated CF₃ groups.

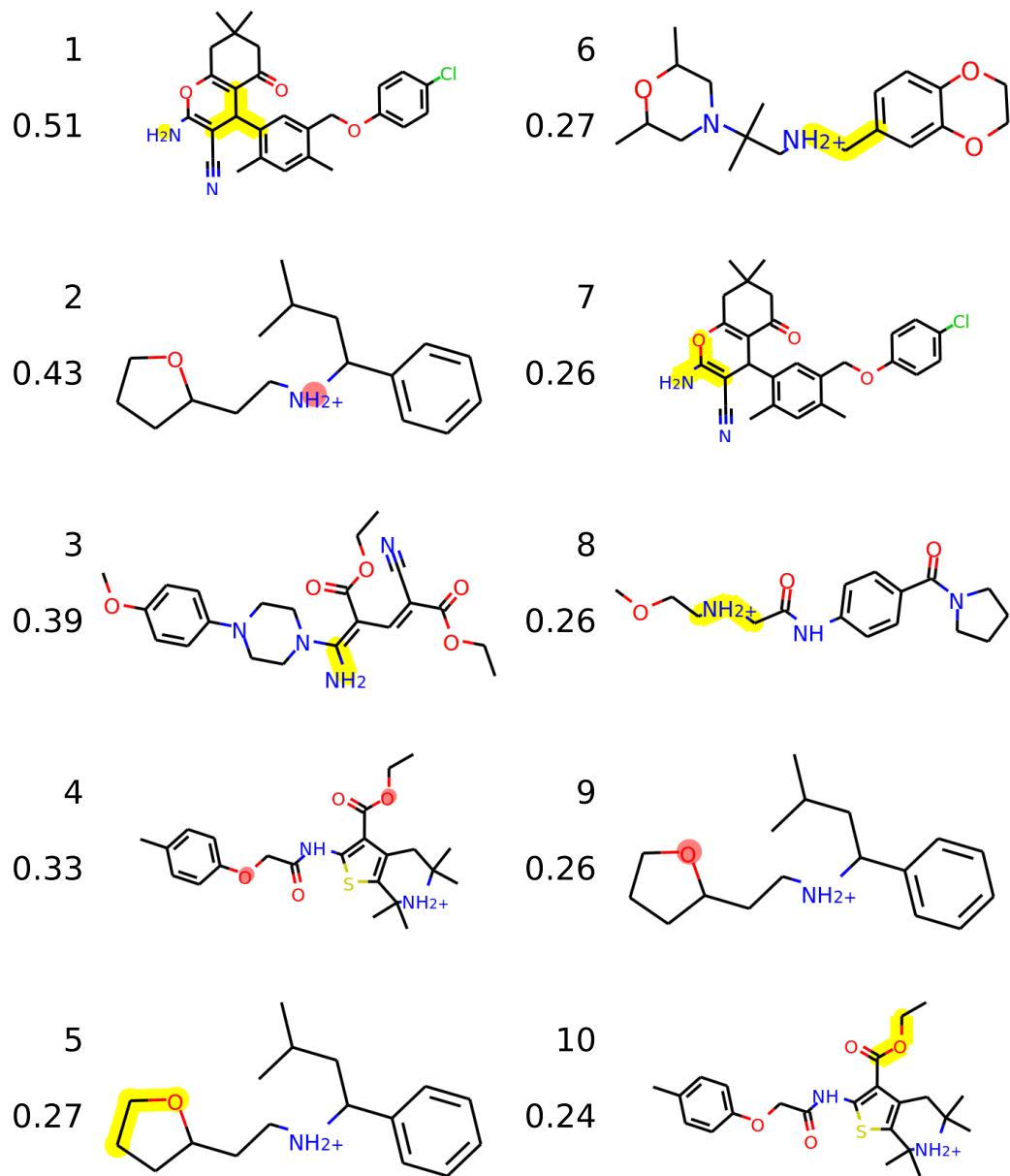


Figure 3.B.11: Top Features Correlated with Activity for Toy Dataset ThreeElem3. The logic for this dataset is benzene and ether and amino. Amino and ether appear several times, but benzene does not appear in the top features. The first feature is actually an amino group; the other branched alkane is an example of bit collision which could confound model results.

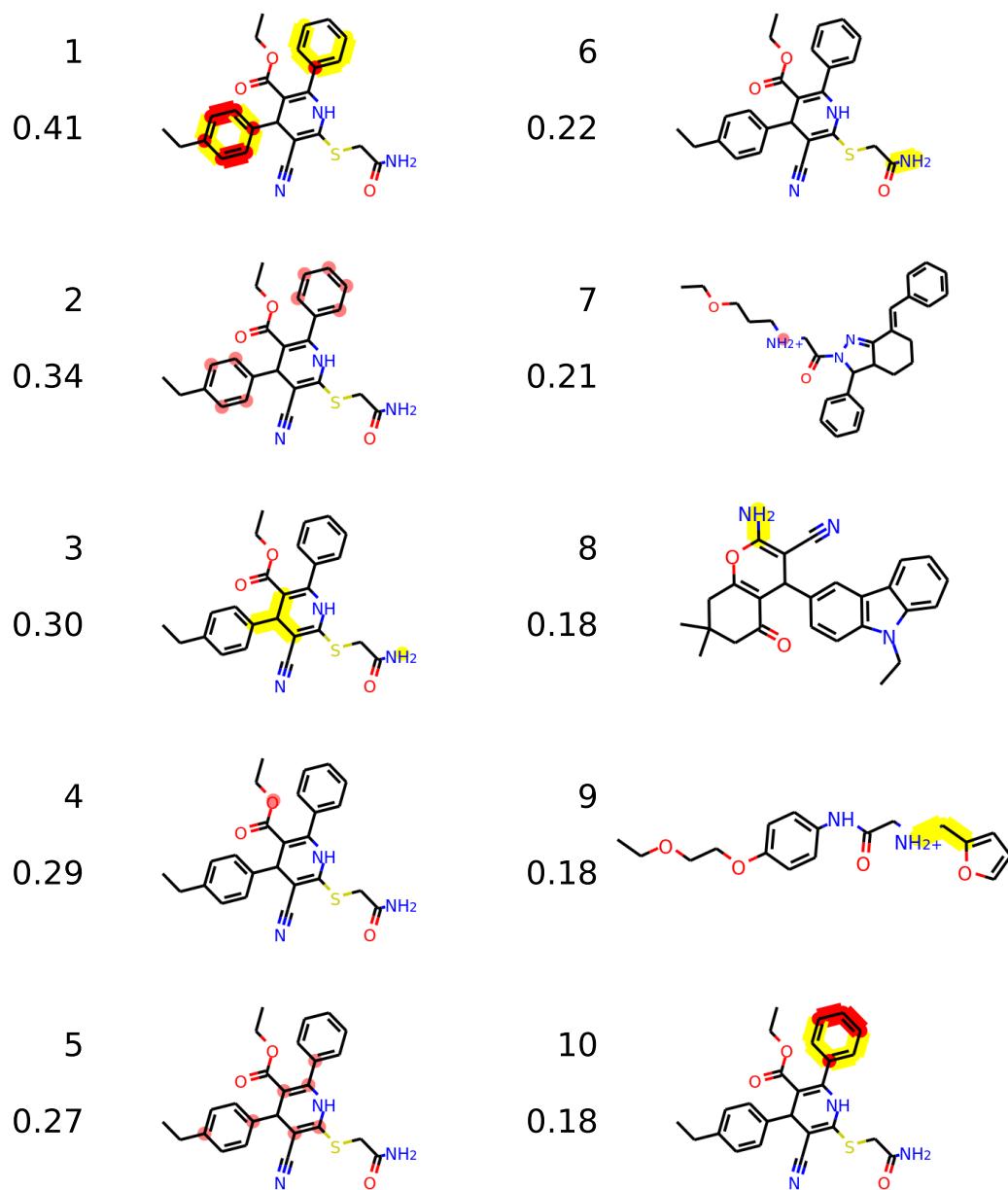


Figure 3.B.12: Top Features Correlated with Activity for Toy Dataset ThreeElem3 with Fragment-Matched Decoys. The logic for this dataset is benzene and ether and amino. All three elements of the logic do appear and we see few extraneous elements aside from the example of bit collision noted earlier.

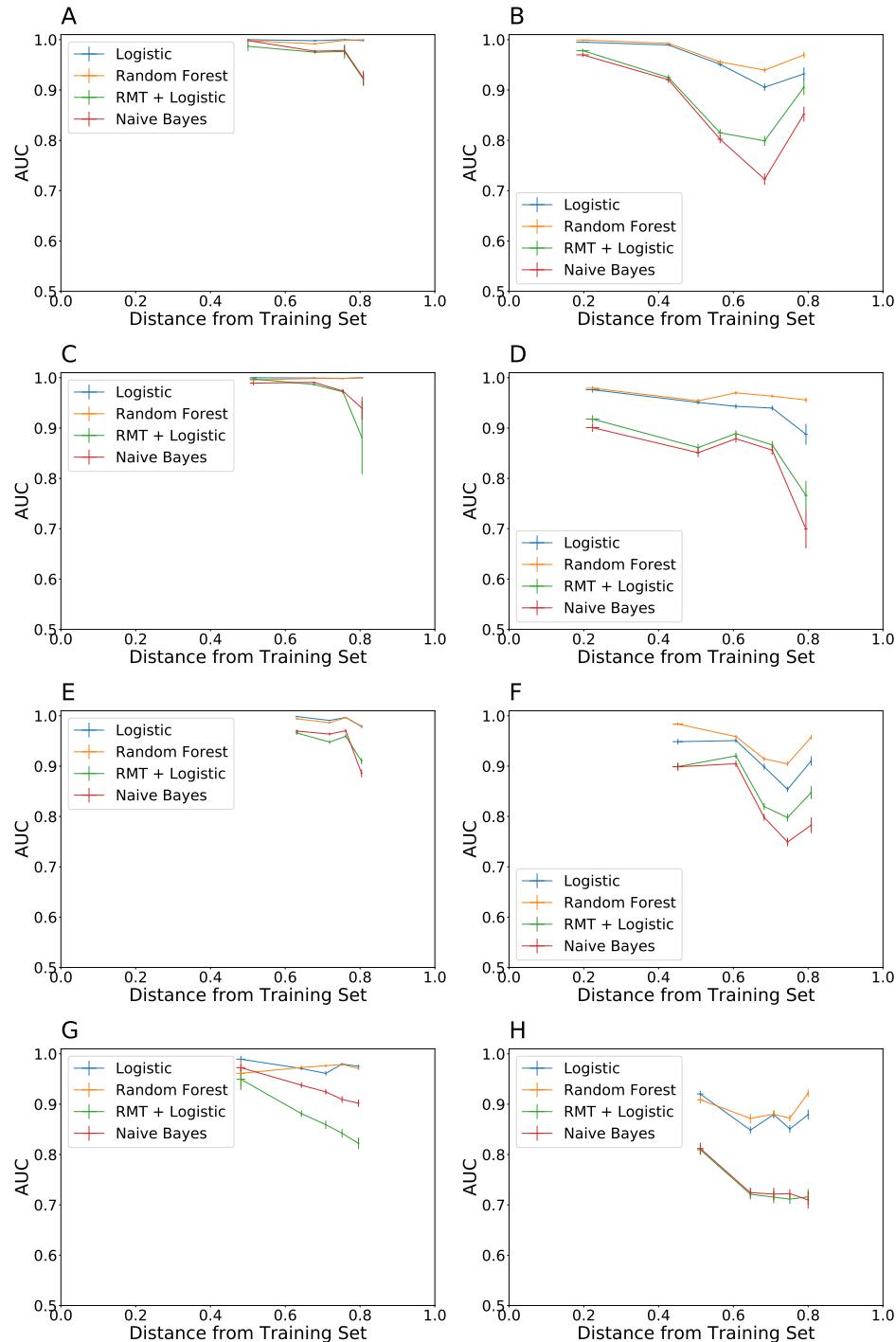


Figure 3.B.13: AUC as a Function of Nearest-Neighbor Distance from Training Set for Toy Datasets (a-b) ThreeElem0, (c-d) ThreeElem1, (e-f) ThreeElem2, and (g-h) ThreeElem3, (a, c, e, g) without and (b, d, f, h) with Fragment-Matched Decoys. As expected, the AUC of our models decreases as nearest-neighbor distance from the training set increases, indicating that our models do not generalise well. The points on these graphs are at the average of the [0, 20]th, [20, 40]th, [40, 60]th, [60, 80]th, and [80, 100]th percentile of test set distances to the training set, respectively. The lowest distance is already higher than 0.4, explaining why the far-AUC for the toy datasets is not a particularly good measure of generalisation ability.

Chapter 4

Using Single Protein/Ligand Binding Models to Predict Active Ligands for Previously Unseen Proteins

4.1 Introduction

ML-based virtual screening campaigns for a single protein target rely on the existence of preliminary experimental screening data that identifies active and inactive ligands for that target, but that can be costly and time-consuming to obtain. Models that predict global drug-target interactions (DTI) aim to remove this bottleneck by predicting the interactions between multiple protein targets and multiple candidate ligands or drugs [9, 57, 52, 19]. We consider a matrix of all interactions between a given set of small molecule ligands, and a given set of protein targets as illustrated in Fig. 4.2.1. These models use preliminary experimental data for some subset of the ligands and targets, together with computational descriptors of both the protein targets and the small molecule ligands to predict this matrix. The goal is to build models that generalise and can accurately predict interactions that involve either previously unseen ligands or protein targets, or in the most challenging case interactions in which no experimental data is available for either the protein target or the small molecule ligand. Such models would enable for example the prediction of active ligands for protein targets for which no prior screening data is available [19, 9]. If successful, these models would be particularly useful in the prediction of off-target effects for a given drug, and

would further enable a better understanding of their polypharmacology.

A variety of approaches have been developed to use machine learning to build models for DTI prediction. They can be broadly classified into similarity-based and feature-based methods, depending on what information they use about the given proteins and ligands and how this information is encoded. Similarity-based methods make predictions based entirely on the matrix that describes the similarity between different proteins and ligands. Typically a sequence similarity metric is used for proteins and the Tanimoto coefficient, which describes 2D structural similarity, is used for the ligands [19, 9]. One simple successful approach is regularized least squares (RLS) which uses kernel ridge regression to make predictions for proteins and ligands that are not contained in the training data [67]. Matrix factorization approaches have also been introduced such as collaborative matrix factorization (CMF) and weighted graph-regularized matrix factorization (WGRMF) which aim to factor the protein/ligand binding matrix as a product of lower-rank matrices that conform to the given similarity matrices [76, 20]. Feature-based methods use standard machine learning algorithms such as decision trees or neural networks on a given feature set. Popular options include structural fingerprints for ligands and amino acid or dipeptide one-hot frequency counts for proteins [19]; more physicochemical descriptors have been used for proteins whose structure is known [57]. All of these methods attempt to predict the entire DTI matrix in one step, using all of the available information at the same time.

However, performance analyses of these methods using standard benchmark datasets demonstrates that in many cases they struggle to generalise in either protein space or ligand space alone [19]. Specifically, when interactions involving a specific protein or ligand are not present in the training data, existing methods often struggle to make accurate predictions. The difficulty of generalising to previously unseen ligands is particularly surprising, since many virtual screening models successfully solve this problem without data about related proteins [13]. The most difficult and most interesting test case of generalising in both protein and ligand space simultaneously has rarely been tried in the literature due to poor model performance [19].

In this chapter, we introduce a preprocessing step that uses robust single protein/ligand binding models to improve the accuracy of DTI models when generalising within the

space of small molecule ligands. We show that this preprocessing step results in improvements for a wide variety of DTI models on a number of datasets. Most importantly, our method makes it possible to tackle the most challenging case involving simultaneous generalisation to both unseen drugs and unseen targets with reasonable accuracy. Our results suggest that, surprisingly, additional binding data about related proteins does not help make better predictions of interactions between unseen ligands and known proteins. However, the additional data generated by a single protein/ligand binding model is crucial for the accuracy of DTI models. This shift in perspective should significantly improve the performance of DTI models.

4.2 Methods

4.2.1 Dataset Construction and Problem Set-up

The dataset analyzed here is based on the gold-standard dataset developed by Yamanishi et al. [75], which we have updated to reflect the vast quantity of protein/ligand binding data that is now available. We took the list of protein targets from Yamanishi’s original work and found active ligands from ChEMBL 24.1 [15, 21] by filtering for compounds with an IC_{50} , K_i , K_d , or EC_{50} of less than 1 μM . Targets with fewer than 20 active ligands in ChEMBL were eliminated. To reduce computational run times for the models, the enzyme dataset was split into a kinase dataset and an other enzyme dataset. Inactive ligands were acquired from two sources: inactive ligands labeled on PubChem indexed by UniProt Protein ID [48, 36] and for targets with a DUD-E decoy set, some of the inactives from the DUD-E set were included [49]. To ensure a reasonable balance of actives to inactives, we also added a randomly selected set of 500 decoys per run; these decoy ligands were selected to not be in any previous set of ligands, either active or inactive. We assumed that these decoys did not interact with any of the given targets. Unlike previous work [19], we did not assume in either training or testing our models that any other unknown interactions between proteins and non-decoy ligands were inactives, since actives for a given protein target may be active for related proteins. Some basic statistics about the size of the datasets may be found in Table 4.A.1.

We next split our dataset into training and test sets. In order to explicitly test our

models' ability to generalise in both protein and ligand space, we chose to provide our model with a training submatrix of protein/ligand interactions as illustrated in Fig. 4.2.1a. Specifically, we first split the protein targets randomly into an 80% training and 20% test set. For each known target, we randomly selected 20% of the active and known inactive ligands to be in the test set. We further selected 20% of the decoys to be in the test set. All remaining ligands were placed in the training set. The models were not provided with any information about interactions involving any of the test ligands or proteins. Our methodology ensures that all protein targets in the training set have some active and inactive ligands in the training set. Further, the training submatrix is potentially incomplete, since there may be interactions between known proteins and known ligands about which we have no experimental data. A number of statistics about our train/test splits are reported in Table 4.A.1.

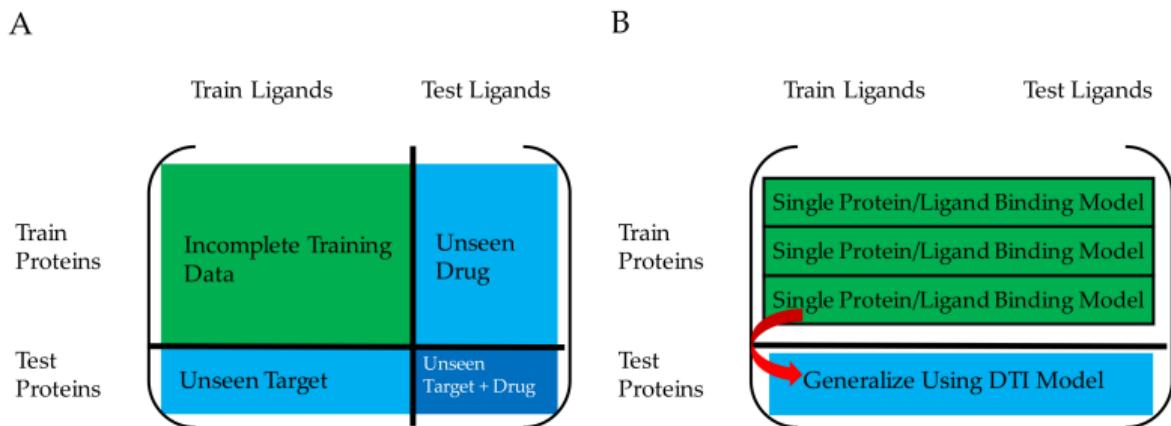


Figure 4.2.1: (a) Our split into training and test sets for the DTI problem, with three subproblems: unseen target, unseen target and drug, unseen drug. This chapter focuses on the unseen target and drug problem, which is most difficult as it requires generalisation in both protein and ligand space. (b) Our proposed method, wherein we build protein/ligand binding models for each training protein with the given data (ignoring any unknown interactions) and use these models to predict the interactions between the given protein and all the ligands provided. We then use a standard DTI model to generalise to the test proteins using all the provided information.

Our set-up allows us to define 3 distinct subproblems testing different methods of generalisation. The unseen drug subproblem on train proteins and test ligands tests our models' ability to generalise in ligand space. The unseen target subproblem on train ligands and test proteins tests our models' ability to generalise in protein space.

The unseen target and drug subproblem on test ligands and test proteins tests our models' ability to generalise simultaneously in both protein and ligand space; we expect this to be the hardest subproblem. This is significantly harder than tasks tested in previous DTI work [19] and will be the subproblem we focus on. Our data from Table 4.A.1 indicate that despite large variance, on average all 3 subproblems are roughly balanced in terms of the number of actives and inactives due to our setup. This allows us to use the Area Under the Receiver Operating Characteristic Curve (AUC) as our metric for measuring model accuracy.

Some models required hyperparameter tuning, so we created a validation submatrix. This was created in the same way as described previously with the train/test split, but the split into train and validation was done within the training submatrix generated previously. Hyperparameters were tuned separately for every repetition using the overall AUC on the validation set; model performance was then evaluated on the test set. We performed 20 repetitions to test all models; error bars reported below are to 1 SEM.

4.2.2 Models

The principle behind our proposal (see Fig. 4.2.1b) is to add a preprocessing step for all the training proteins. We build single protein/ligand binding models for each training protein with the given data (ignoring any unknown interactions) and use these models to predict the interactions between the given protein and all the ligands provided. We then use a standard DTI model to generalise to the test proteins using all the provided information.

The single protein/ligand binding models we tested were random forest and logistic regression with ECFP6 fingerprints with 2048 bits. [58] Both models were implemented with scikit-learn. [51] We used $C = 1$ for logistic regression and 100 trees with a maximum depth of 25 for random forest. These models are known to perform well on the single protein/ligand binding problem.

There are two important modifications that must be made to all DTI models used. First, the DTI model must accept probabilities in its training set. Second, the DTI model must not incorporate any additional information about ligand similarity, since we do not need to generalise to any further ligands. Section 4.B.4 and Fig. 4.B.12 show

that failing to make this modification results in poorer performance. We made these modifications on a case-by-case basis to all of the models.

We tested one baseline model and four high-performing standard models from the literature [19]. In all cases, the similarity matrices used were normalized pairwise E-value scores from HMMER for the proteins [18] and Tanimoto similarities on the ECFP6 fingerprints for the ligands. A list of the DTI models tested, their definitions and properties, and the hyperparameters used for training may be found in Sections 4.A.2 - 4.A.3.

4.3 Results

We omit results about the Kinase, Other Enzyme, and Ion Channel datasets in the main text. Those results may be found in Section 4.B.1 and Figs. 4.B.1 - 4.B.4, and also support our conclusions.

4.3.1 On the Unseen Target and Drug Subproblem

We present the AUCs of our models with and without logistic regression as preprocessing on the unseen target and drug subproblem in Fig. 4.3.1. We see improvement upon adding either logistic regression or random forest to almost all models on almost all datasets. These results appear to hold for a variety of model types, ranging from our baseline nearest-neighbor models to matrix factorization-based methods to one-hot featurized methods, and do not depend on dataset size or sparsity. All of these results are statistically significant at a $p < 0.01$ level. Figs. 4.B.5 and 4.B.8 demonstrate that these results largely hold on a trial-by-trial basis as well.

Fig. 4.3.2 shows the results of our models with both logistic regression and random forest as preprocessing on the unseen target and drug subproblem. In general, we see greater improvement when logistic regression is added as a preprocessing step rather than random forest. When logistic regression is used, even baseline models like weighted-NN, regardless of their previous performance, often achieve high accuracy with AUCs exceeding 0.85, suggesting that our method is crucial in allowing essentially any DTI model to generalise simultaneously in both protein and ligand space. Many of the similarity-based models perform very similarly once our preprocessing is

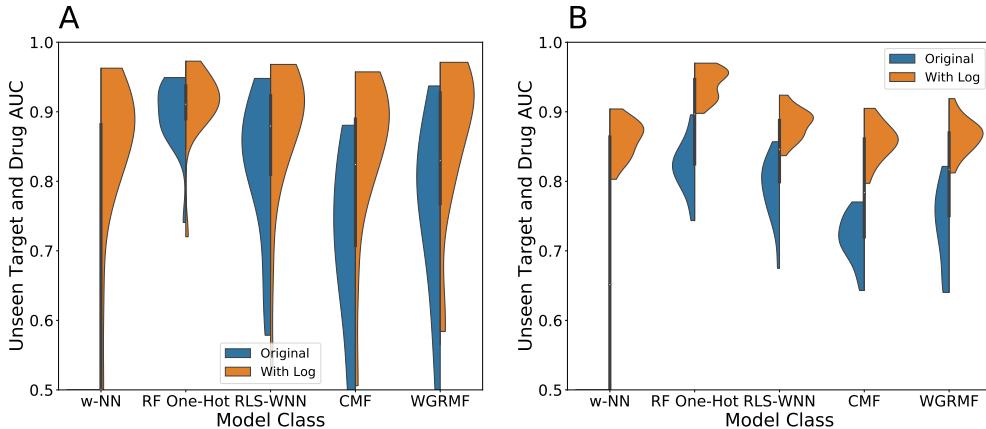


Figure 4.3.1: Effect of Adding Logistic Regression Preprocessing on Unseen Target and Drug AUCs for DTI Models for (a) the Nuclear Receptor Dataset and (b) the GPCR Dataset. We see consistent improvement in all models upon adding logistic regression as preprocessing with single protein/ligand binding models, regardless of dataset size or sparsity and type of DTI model used. In particular, RF one-hot with logistic regression as preprocessing consistently attains AUCs above 0.9, suggesting that it is now capable of simultaneously generalising in protein and ligand space.

added, suggesting that our method is responsible for the vast majority of their performance on this problem.

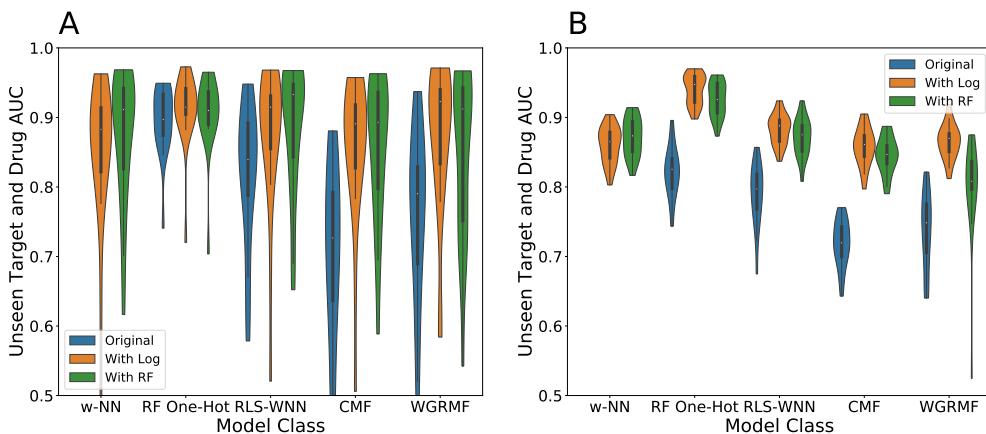


Figure 4.3.2: Unseen Target and Drug AUCs for All DTI Models for (a) the Nuclear Receptor Dataset and (b) the GPCR Dataset. We typically see greater improvement with logistic regression as preprocessing than with random forest, but both methods are still effective.

RF one-hot with logistic regression as preprocessing consistently performs very well, attaining AUCs above 0.9. Our results suggest that this model is now capable of simultaneously generalising in both protein and ligand space. Further, in most datasets

the addition of preprocessing with logistic regression is key to the performance of this model.

We did not compute the exact runtime of each individual model since a number of steps common to multiple models were performed once for optimization purposes and different runs of the program occurred on different computers. However, our observations indicate that adding single protein/ligand binding models significantly reduces both runtime and memory costs since the drug similarity matrix does not have to be computed and stored; doing so for a reasonably large dataset can take hundreds of gigabytes and several hours of computer time on a single core.

4.3.2 On the Other Subproblems

We now examine the performance of our models on the other 2 subproblems outlined above. We begin with the unseen drug subproblem; our results are shown in Fig. 4.3.3. As expected, we see significant improvement in the performance of all models upon adding the preprocessing with single protein/ligand binding models; all of these results are statistically significant at a $p < 0.01$ level. Figs. 4.B.7 and 4.B.10 demonstrate that these results hold on a trial-by-trial basis as well. For all but the kinase dataset, it appears that the single protein/ligand binding models are nearly perfect, suggesting that this subproblem is relatively easy to model likely due to dataset bias [59, 72]. Regardless, the DTI models on their own are unable to replicate this success.

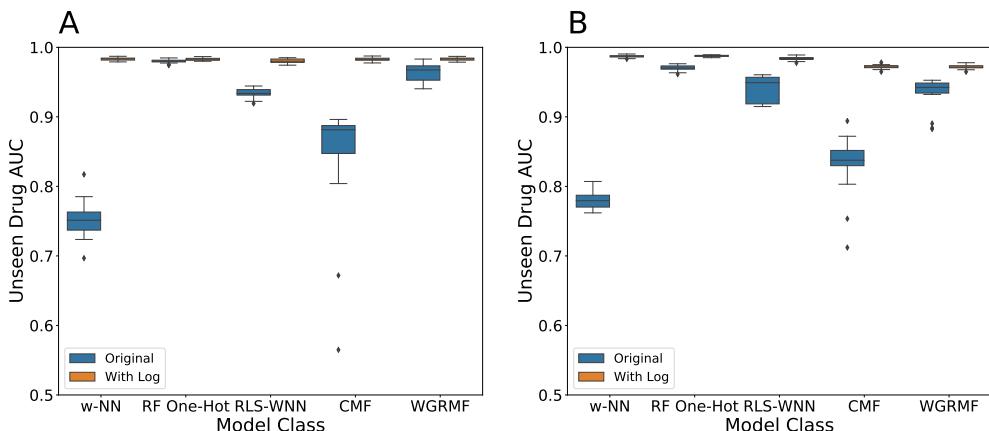


Figure 4.3.3: Unseen Drug AUCs for All DTI Models for (a) the Nuclear Receptor Dataset and (b) the GPCR Dataset. We see significant improvement in all models upon adding the preprocessing with single protein/ligand binding models.

Our results on the unseen target problem are shown in Fig. 4.3.4. Unexpectedly, we see consistent improvement in all the non-one-hot-based methods, despite the fact that this problem focuses on known drugs; all of these results are statistically significant at a $p < 0.01$ level. Figs. 4.B.6 and 4.B.9 demonstrate that these results hold on a trial-by-trial basis as well. This is true even for models such as WGRMF, where preprocessing has minimal effect on the unseen drug subproblem but a much larger effect in the unseen target subproblem. Thus even a model that performs well in the unseen drug subproblem can benefit from our method of preprocessing with single protein/ligand binding models.

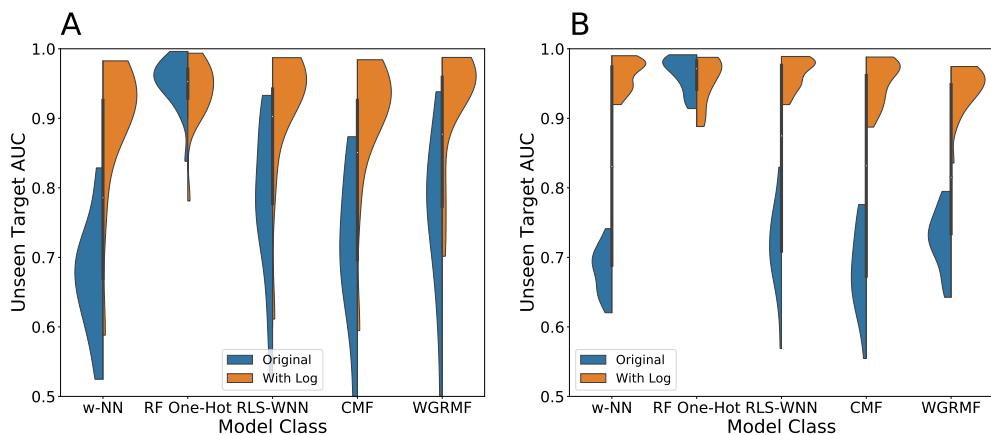


Figure 4.3.4: Unseen Target AUCs for All DTI Models for (a) the Nuclear Receptor Dataset and (b) the GPCR Dataset. We see consistent improvement in all models aside from RF one-hot upon adding preprocessing with single protein/ligand binding models.

Our method does not show improvement for RF one-hot on the unseen target subproblem. It is unclear why this is the case, but we observed that the improvements on the unseen target and drug subproblem and unseen drug subproblem consistently outweigh the small worsening of performance on the unseen target subproblem, suggesting that our method is still useful for RF one-hot.

4.3.3 Proposed Explanations

We propose two explanations of the improvement in performance described previously. For the unseen drug problem, many standard DTI models implicitly use Tanimoto similarity as the only ligand feature. Analysis of single protein/ligand binding models has indicated that Tanimoto similarity is not a particularly robust model and

options such as logistic regression and random forest perform better. Thus it is not surprising that we see improvements in the unseen drug problem for these models.

This explanation does not apply to the one-hot feature models, which receive the same ligand features as the single protein/ligand binding models. Our results suggest that unexpectedly the added information about related proteins tends to confuse the model instead of improving its performance. It is possible that relatively simple models like logistic regression and random forest are not sufficiently expressive to capture the full complexity of the multiple protein/ligand binding problem, since massively multitask neural networks have previously proven successful at transferring data between related proteins [54].

For the unseen target and unseen target and drug problems, the key advantage our preprocessing provides DTI models with is additional data that they can fit to. This data is clearly very accurate, as demonstrated by our performance on the unseen drug problem. It is also nontrivial. Fig. 4.3.5 is a histogram of the additional information our method provides to DTI models on the GPCR dataset. Specifically, it shows the probabilities of interactions predicted by the single protein/ligand binding logistic regression and random forest. Supplementary Fig. S12 shows similar data for the other datasets. While most interactions are predicted to be inactive, a small but significant proportion of the interactions are predicted to be active; in contrast, standard approaches either ignore this data or assume that any unknown interactions are inactive [19]. We believe that this additional data explains the improvement in performance in both the unseen target and unseen target and drug problems.

Logistic regression consistently performs better as a single protein/ligand binding model than random forest. Fig. 4.3.5 suggests that this is at least partly a calibration issue; it is known that logistic regression outputs probabilities whereas random forest is not generally correctly calibrated [6]. It is also known that logistic regression is marginally better at generalising than random forest; this may help explain some of the difference.

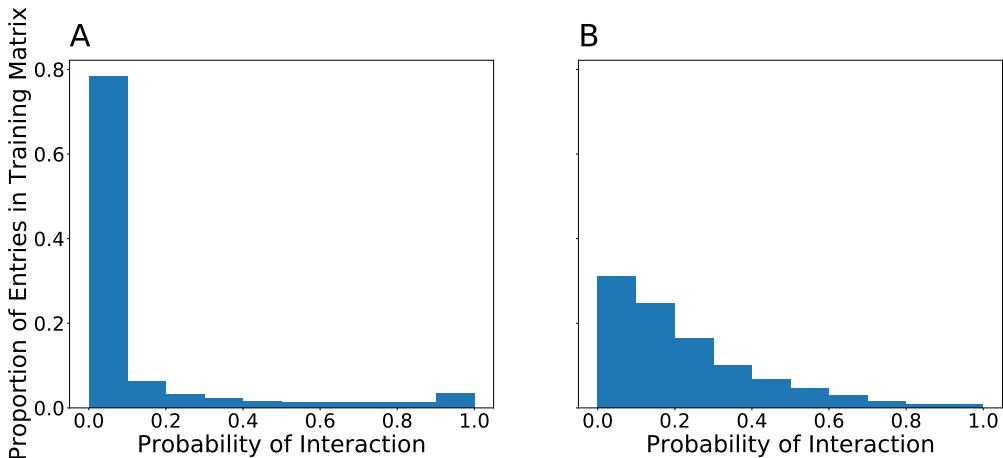


Figure 4.3.5: Histogram of Added Information to Training Matrix when Preprocessed by (a) Logistic Regression and (b) Random Forest. This shows the predicted probabilities of interactions that are added to the training matrix by the appropriate single protein/ligand binding model. Most of the interactions are predicted to be inactive, but some are predicted to be active. We hypothesize that this additional data helps explain some of the improvement in the unseen target and drug and unseen target problems.

4.4 Conclusions

In this work, we have demonstrated that preprocessing with robust single protein/ligand binding models allows generalisation simultaneously in both protein and ligand space to predict interactions between unseen drugs and unseen targets. We observe this effect consistently regardless of dataset size or DTI model and see that it also applies for the unseen drug and unseen target problems. We believe this method works primarily because the additional high-quality predictions provided by the single protein/ligand binding models significantly improves the accuracy of the predictions made by the overall DTI models.

In particular, we note that logistic regression as a single protein/ligand binding model tends to improve performance the most. Our best-performing DTI method was random forest with the one-hot feature set of amino acid counts and ECFP6 ligand fingerprints. Since this is a very simple model, we suspect that more complicated models like deep neural networks will be able to learn more information about the dataset and perform better. Our work also suggests feature-based methods perform significantly better than similarity-based methods and should be the preferred option for future research into DTI models.

There are a number of open questions that remain about this work. We have shown

the unexpected result that adding more data about related proteins consistently degrades performance on the unseen drug subproblem when compared to the same single protein/ligand binding model that does not have access to that data, in contrast to the results of multitask neural networks [54]. We believe that this is due to the lack of expressivity of our models, logistic regression and random forest, to fully model the multiple protein/ligand binding problem. Further research should be done to see if this still holds for more complicated models like deep neural networks.

Next, all of the DTI models we have used in this chapter accept probabilities in their training data, but none of them interpret the probabilities in the most natural way. The canonical probability loss function is cross-entropy loss, but all of the regression approaches use least-squares loss to fit. It is possible that changing the loss function will improve overall performance. Similarly, it is possible that correctly calibrating the single protein/ligand random forest models will improve performance, as we previously noted that they are currently incorrectly calibrated in Fig. 4.3.5b.

Another important concern is about the generalisation ability of our model. It is known that single protein/ligand binding models that perform well on benchmark datasets prove unable to generalise due to clustering in chemical space and dataset bias [70, 11, 59, 61, 72, 43]. Similarly, clustering in protein space due to phylogenetic relationships can result in dataset bias and overly optimistic predictions of a model’s ability to generalise. We suspect that these biases may occasionally play a role in the high performance of our models given the high variance in performance observed between different train/test splits on the same dataset. More careful data splits should be designed that account for these biases accurately to gain a better evaluation of our models.

Finally, we see that many of the target similarity-based methods perform very similarly and very close to the baseline methods on the unseen target and drug problem once the single protein/ligand binding models are added. This suggests that the key to improving performance is to develop a better similarity matrix or use explicit features instead, rather than developing more complicated similarity-based models.

Appendix: Supporting Information

4.A Supplementary Methods

4.A.1 Dataset Statistics

Some basic statistics about the size of the datasets may be found in Table 4.A.1.

| Dataset | Nuclear Receptor | Ion Channel | GPCR | Kinase | Other Enzyme |
|-------------------------------|------------------|-----------------|------------------|------------------|------------------|
| Proteins | 21 | 40 | 91 | 62 | 113 |
| Ligands | 19197 | 17536 | 75380 | 66120 | 69923 |
| Actives | 12978 | 19717 | 96289 | 65654 | 69968 |
| Non-Decoy Inactives | 11438 | 1480 | 13289 | 28899 | 16972 |
| Mean Protein Similarity | 0.145 | 0.033 | 0.074 | 0.081 | 0.009 |
| Train-Test Protein Similarity | 0.51 ± 0.20 | 0.43 ± 0.28 | 0.37 ± 0.14 | 0.47 ± 0.23 | 0.33 ± 0.28 |
| Train-Test Ligand Similarity | 0.61 ± 0.20 | 0.70 ± 0.16 | 0.72 ± 0.15 | 0.67 ± 0.18 | 0.67 ± 0.17 |
| Train Ligands | 14969 ± 14 | 13860 ± 14 | 55885 ± 32 | 49776 ± 45 | 53865 ± 28 |
| Test Ligands | 4723 ± 14 | 4172 ± 14 | 19974 ± 32 | 16830 ± 45 | 16537 ± 28 |
| Unseen Drug Actives | 3018 ± 350 | 4168 ± 561 | 25979 ± 1170 | 20492 ± 1125 | 16083 ± 1374 |
| Unseen Drug Inactives | 3861 ± 290 | 3508 ± 54 | 10250 ± 375 | 10860 ± 688 | 12306 ± 315 |
| Unseen Target Actives | 2020 ± 932 | 2927 ± 1426 | 13658 ± 2252 | 8417 ± 2455 | 10098 ± 2731 |
| Unseen Target Inactives | 4185 ± 841 | 3407 ± 165 | 8957 ± 1548 | 9561 ± 2261 | 11503 ± 1099 |
| Unseen Target/Drug Actives | 889 ± 351 | 1116 ± 567 | 7127 ± 1115 | 5435 ± 1071 | 4384 ± 1379 |
| Unseen Target/Drug Inactives | 1275 ± 285 | 889 ± 49 | 2655 ± 378 | 2866 ± 696 | 3207 ± 301 |

Table 4.A.1: Statistics of Datasets Used. This table reports the number of proteins, ligands, active interactions, non-decoy inactive interactions, and the mean similarity between all proteins in the dataset. We also report the mean similarity between any protein or ligand in the test set and its closest protein or ligand in the training set, the number of training and test ligands, and the number of actives and inactives in the unseen drug, unseen target, and unseen drug + target problems.

4.A.2 Model Descriptions

A list of the DTI models tested follows, along with the modifications we made. For convenience, let the training drug-target similarity matrix be Y , the predicted output from the model Y_{pred} , the drug similarity matrix be S_d , and the target similarity matrix be S_t .

1. Weighted Nearest Neighbor (Weighted NN), a baseline method which weights the nearest interactions in protein and ligand space by their similarity [75]. Unknown interactions are assumed to be non-interactions.
2. Random Forest One-Hot (RF One-Hot). The feature set used was ECFP6 fingerprints with 2048 bits for the ligands and a count of amino acids for the proteins. This is distinct from the single protein/ligand binding models since it attempts to fit the data for all the proteins at once. In order to incorporate the probabilistic information, we used a regressor model instead of a classifier with a least-squared loss.
3. Regularized Least Squares-Weighted Nearest Neighbor (RLS-WNN), a regularized least-squares linear regression model based on protein and ligand similarity. Unknown interactions are assumed to be non-interactions. We first infer interactions for completely unseen drugs and targets by letting

$$Y(d_i) = \sum_{j=1}^n w_j Y(d_j)$$

where the drugs are sorted in descending order of similarity and $w_j = \eta^{j-1}$ for some hyperparameter η [67].

Next, we compute Gaussian interaction profile (GIP) kernels; the drug kernel is

$$\text{GIP}_d(d_i, d_j) = \exp(-\gamma ||Y(d_i) - Y(d_j)||^2)$$

for some hyperparameter γ and the target kernel is defined similarly. We then merge these with the similarity matrices to get

$$K_d = \alpha S_d + (1 - \alpha) \text{GIP}_d$$

4.A Supplementary Methods

for some hyperparameter $\alpha, 0 \leq \alpha \leq 1$. Let $K = K_d \otimes K_t$ be the kernel over drug-target pairs; we can then use kernel ridge regression to compute

$$\hat{Y}_{\text{pred}} = K(K + \sigma I)^{-1} \hat{Y}$$

where \hat{Y} is the flattened version of the training matrix. The inverse is computed via an eigendecomposition due to computational constraints [67].

Due to the larger dataset size, we approximated the inverse of the drug similarity matrix using only the top 100 eigenvalues instead of computing it exactly. When we incorporated our preprocessing step, we only used the target similarity matrix in the RLS computation and omitted the weighted nearest neighbor preprocessing.

4. Collaborative Matrix Factorization (CMF) based on protein and ligand similarity. CMF finds matrices A and B to minimize the function

$$\left\| W \odot (Y - AB^T) \right\|^2 + \lambda_l (||A||^2 + ||B||^2) + \lambda_t \left\| S_t - AA^T \right\|^2 + \lambda_d \left\| S_d - BB^T \right\|^2$$

where W is a weight matrix with $W_{ij} = 0$ for unknown pairs, \odot is an elementwise product, all norms are the Frobenius norm, and λ_l , λ_d , and λ_t are hyperparameters. Optimization occurs via an alternating least squares algorithm [76]. When incorporating the single protein/ligand binding models, we set the hyperparameter λ_d corresponding to drug similarity to 0.

5. Weighted Graph-Regularized Matrix Factorization (WGRMF) based on protein and ligand similarity. The loss function is

$$\left\| W \odot (Y - AB^T) \right\|^2 + \lambda_l (||A||^2 + ||B||^2) + \lambda_t \text{Tr}(A^T \ell_t A) + \lambda_d \text{Tr}(B^T \ell_d B)$$

where ℓ_d, ℓ_t are normalized graph Laplacians for a sparsified drug/target graph. Sparsification occurred by only keeping the p nearest neighbors for any given point in the graph. Optimization is essentially the same as in CMF [20]. When incorporating the single protein/ligand binding models, we set the hyperparameter corresponding to drug similarity to 0.

4.A.3 Hyperparameter Selection

All hyperparameters were individually selected for each run of every given model. In order to select the hyperparameters, we used a 3-way train/validation/test split, where the train+validation submatrix was generated as described in the main text and the training submatrix was generated as a subset of the train+validation submatrix. Specifically, we used the procedure previously described to identify train+validation and test ligands and proteins. We then repeated this procedure within the set of train+validation ligands and proteins, splitting the protein targets randomly into an 75% training and 25% test set and proceeding similarly with the ligands. The optimal hyperparameter on the validation set was selected and used to measure performance on the test set.

Hyperparameters were adapted from successful values in the literature [19]. The list of hyperparameters used follows:

1. Weighted NN had no relevant hyperparameters.
2. RF One-Hot. The number of trees was allowed to vary within {50, 100, 150} and the maximum depth {15, 20, 25, 30}.
3. RLS-WNN. The ordered triple (σ, α, η) was allowed to vary within $\{(0.25, 0.1, 0.4), (0.5, 0.9, 0.4), (2, 0.6, 0.1), (0.5, 1, 0.6)\}$.

When preprocessing was added, the same ordered triple was allowed to vary within

$$\{(0.5, 0.5, 0.6), (0.5, 1, 0.7), (0.25, 1, 0.7)\};$$

we found these values consistently performed better than the previous ones only with preprocessing.

4. CMF. The ordered triple $(\lambda_l, \lambda_d, \lambda_t)$ was allowed to vary within $\{(1, 4, 32), (2, 8, 0.125), (4, 32, 0.25), (2, 64, 0.125), (0.25, 0.25, 32), (1, 0.0625, 2)\}$.

When preprocessing was added, we fixed $\lambda_d = 0$ and allowed

$$(\lambda_l, \lambda_t) \in \{(1, 32), (0.25, 32), (0.5, 64), (0.5, 32)\}.$$

5. WGRMF. The ordered quadruple $(\lambda_l, \lambda_d, \lambda_t, p)$ was allowed to vary within

$$\{(0.0625, 0.05, 0.1, 2), (0.25, 0.2, 0.2, 4), (0.25, 0.2, 0.2, 4), (0.25, 0.2, 0.2, 5)\}.$$

When preprocessing was added, we fixed $\lambda_d = 0$ in the above values. p determines the degree of sparsification of the training matrix used as a form of regularization prior to running the WGRMF algorithm [20].

4.B Supplementary Results

4.B.1 Additional Datasets

Figs. 4.3.1, 4.3.2, 4.3.3, and 4.3.4 show the improvement in performance on all three subproblems upon adding our preprocessing method for all datasets, not just the 2 shown in the main text, and for both logistic regression and random forest preprocessing. We see consistent improvement regardless of the size or other properties of the dataset and improvement for both logistic regression and random forest. Figs. 4.3.2, 4.3.3, and 4.3.4 suggest that logistic regression is a slightly superior means of preprocessing when compared to random forest.

4.B.2 Trial-by-Trial Comparisons

Instead of examining averages over all 20 trials of the algorithm, we can also examine performance on each individual train/test split. A comparison of the unseen drug and target AUCs is shown in Fig. 4.B.5, of the unseen target AUCs in Fig. 4.B.6, and of the unseen drug AUCs in Fig. 4.B.7. Each data point in these plots corresponds to one of the 20 train/validation splits and shows the AUC of our models on the appropriate subproblem with and without preprocessing with logistic regression.

Figs. 4.B.5 and 4.B.6 demonstrate that despite large variance in performance between trials, on almost every individual trial for almost all models we see consistent

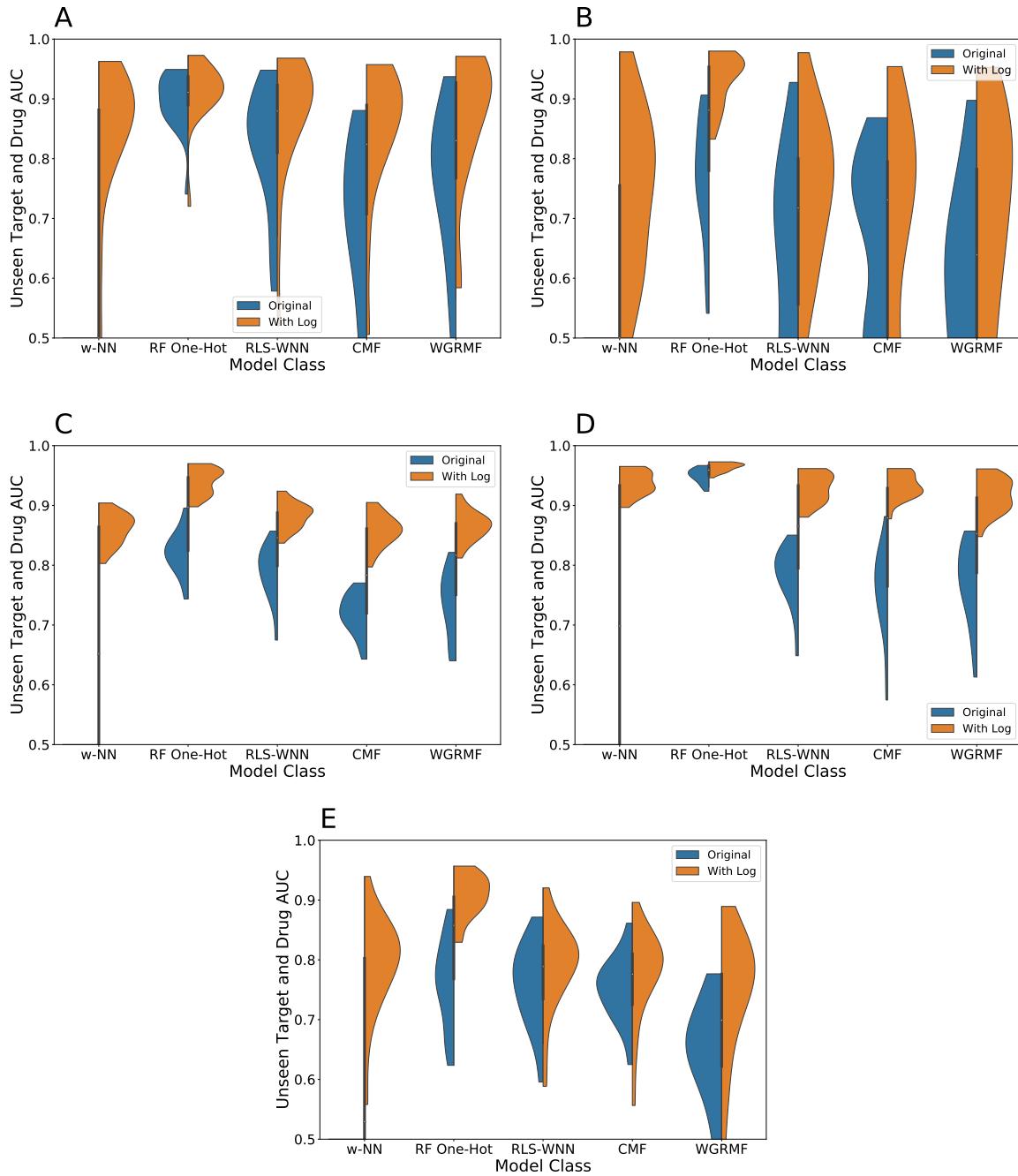


Figure 4.B.1: Effect of Adding Logistic Regression Preprocessing on Unseen Target and Drug AUCs for DTI Models for (a) the Nuclear Receptor Dataset, (b) Ion Channel Dataset, (c) the GPCR Dataset, (d) the Kinase Dataset, and (e) the Other Enzyme Dataset. We see consistent improvement in all models upon adding logistic regression as preprocessing with single protein/ligand binding models, regardless of dataset size or sparsity and type of DTI model used.

improvement after preprocessing is added. The lone exception is RF One-Hot on the unseen target AUC, as mentioned in the main text. The variance in model performance

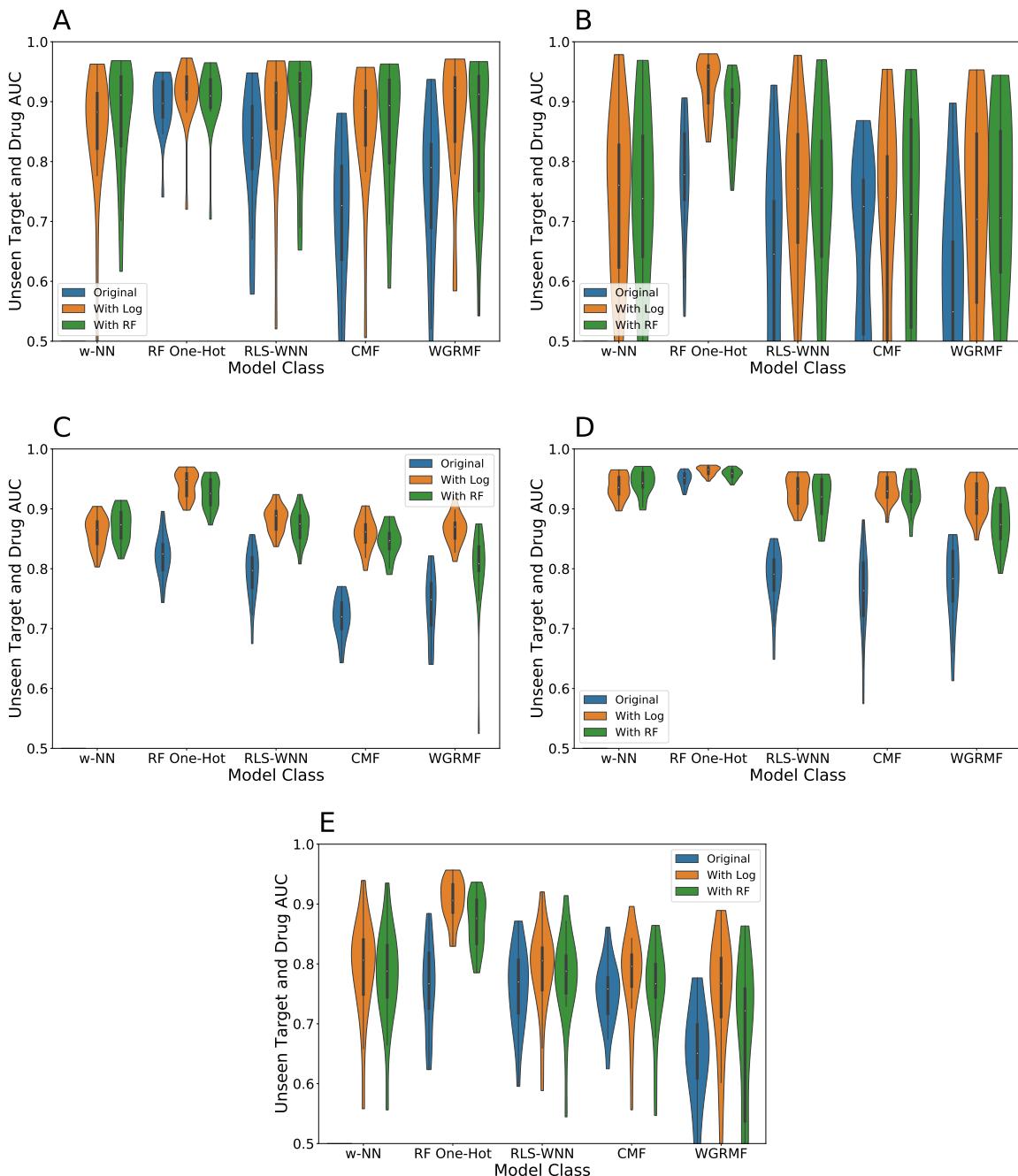


Figure 4.B.2: Unseen Target and Drug AUCs for All DTI Models for (a) the Nuclear Receptor Dataset, (b) Ion Channel Dataset, (c) the GPCR Dataset, (d) the Kinase Dataset, and (e) the Other Enzyme Dataset. We typically see greater improvement with logistic regression as preprocessing than with random forest, but both methods are still effective.

is likely due to randomness in the set of test proteins; since there are a small number of protein targets for all datasets, if a protein that is particularly distant is held-out, that could decrease performance. This variance also tends to decrease after application of our preprocessing.

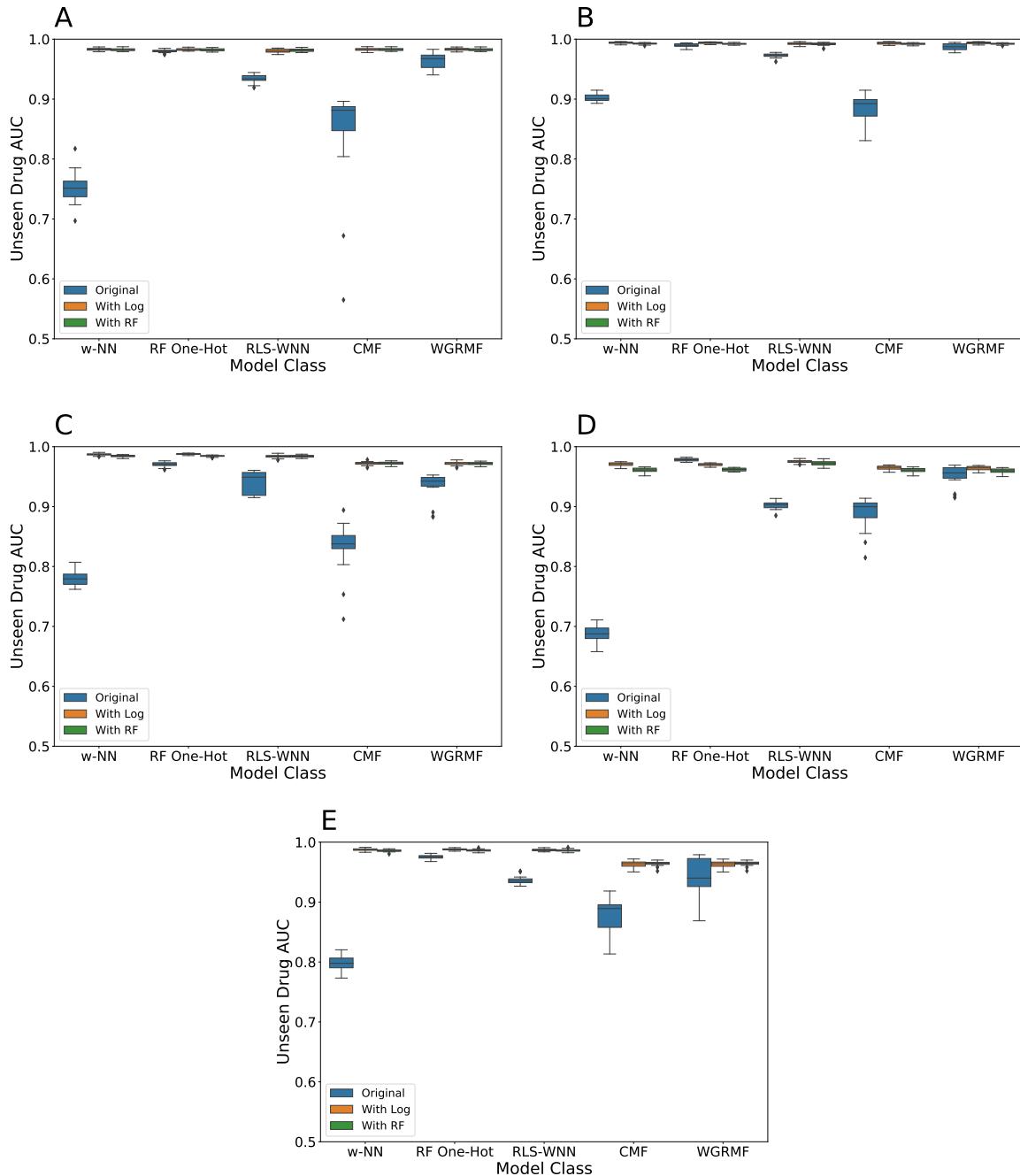


Figure 4.B.3: Unseen Drug AUCs for All DTI Models for (a) the Nuclear Receptor Dataset, (b) Ion Channel Dataset, (c) the GPCR Dataset, (d) the Kinase Dataset, and (e) the Other Enzyme Dataset. We see significant improvement in all models upon adding the preprocessing with single protein/ligand binding models.

Fig. 4.B.7 demonstrates that our models consistently perform outstandingly on the unseen drug subproblem. As noted in the main text, this is likely at least partially a result of dataset bias in chemical space.

We make the same comparisons with random forest used as the preprocessing method

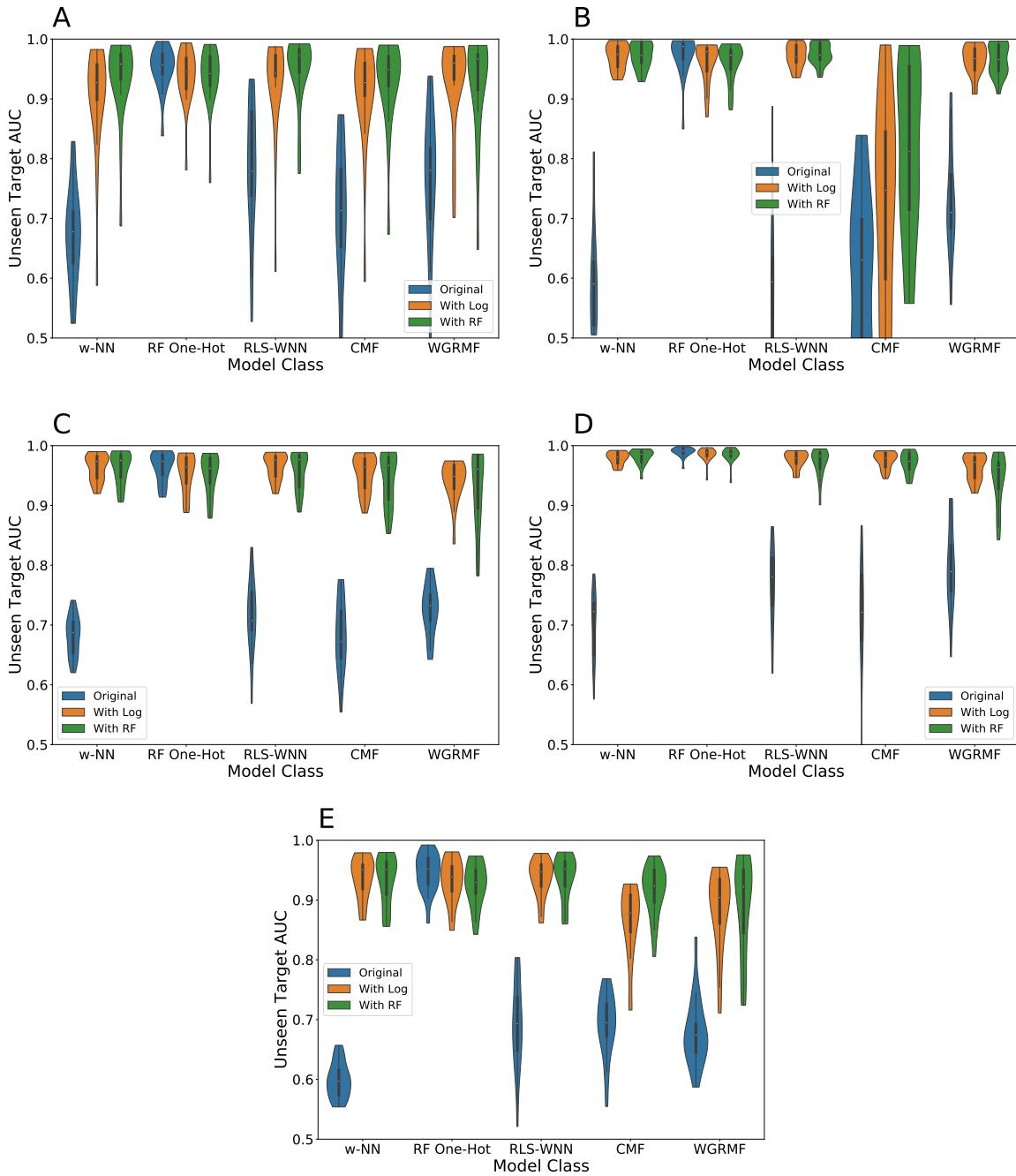


Figure 4.B.4: Unseen Target AUCs for All DTI Models for (a) the Nuclear Receptor Dataset, (b) Ion Channel Dataset, (c) the GPCR Dataset, (d) the Kinase Dataset, and (e) the Other Enzyme Dataset. We see consistent improvement in all models aside from RF one-hot upon adding preprocessing with single protein/ligand binding models.

in Figs. 4.B.8, 4.B.9, and 4.B.10. Our conclusions are similar as to the case with logistic regression.

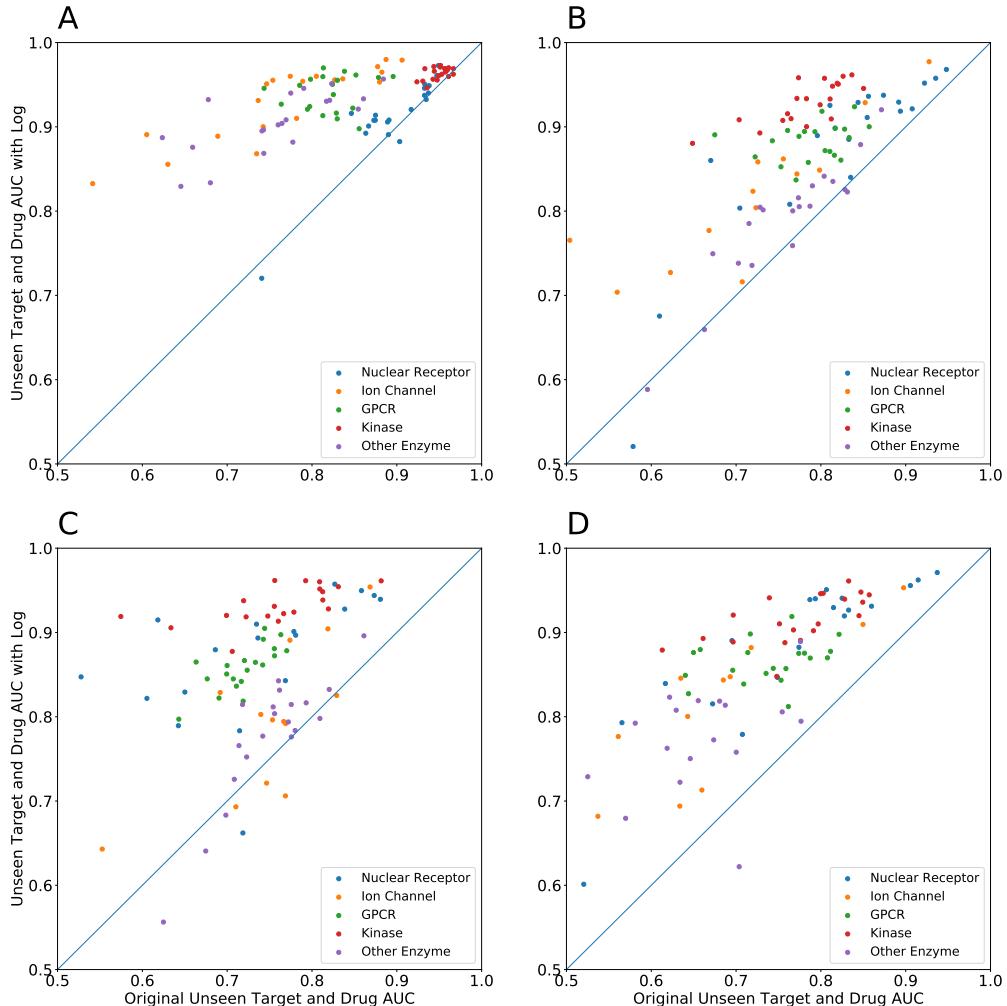


Figure 4.B.5: Comparison of Unseen Drug and Target AUCs with and without Logistic Regression Preprocessing for (a) RF One-Hot, (b) RLS-WNN, (c) CMF, and (d) WGRMF. We see improvement on every method in almost all test runs. The wide variance in performance is likely due to randomness in which proteins are held-out; our method also decreases this variance.

4.B.3 Failure to Eliminate Ligand Similarity

To better understand the importance of eliminating the ligand similarity matrix in the DTI model, we examined the effects of not eliminating it in the RLS-WNN algorithm. We compared five models: the original RLS-WNN algorithm, the RLS-WNN algorithm unmodified but with preprocessing via logistic regression or random forest, and the RLS-WNN modified to only look at the target similarity matrix and with both

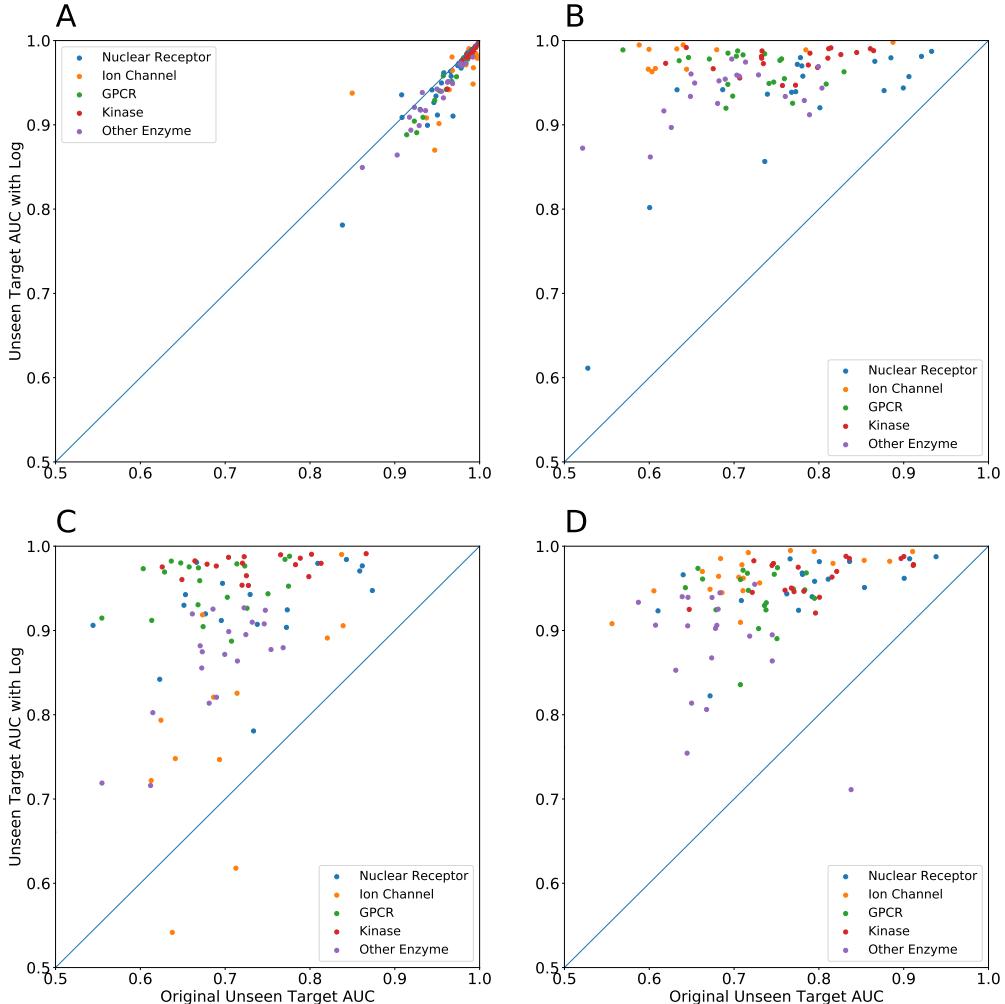


Figure 4.B.6: Comparison of Unseen Target AUCs with and without Logistic Regression Preprocessing for (a) RF One-Hot, (b) RLS-WNN, (c) CMF, and (d) WGRMF. We see improvement on every method in almost all test runs for models aside from RF One-Hot. The wide variance in performance is likely due to randomness in which proteins are held-out; our method also decreases this variance.

preprocessing methods.

Our results are shown in Fig. 4.B.11. We observe that the unmodified RLS-WNN algorithm with our preprocessing method is often better than the original RLS-WNN model, but it is still significantly worse than the RLS-WNN model that correctly ignores ligand similarity. In particular, the unseen target AUCs and some of the unseen drug AUCs are significantly worse without the modification. We also see that when

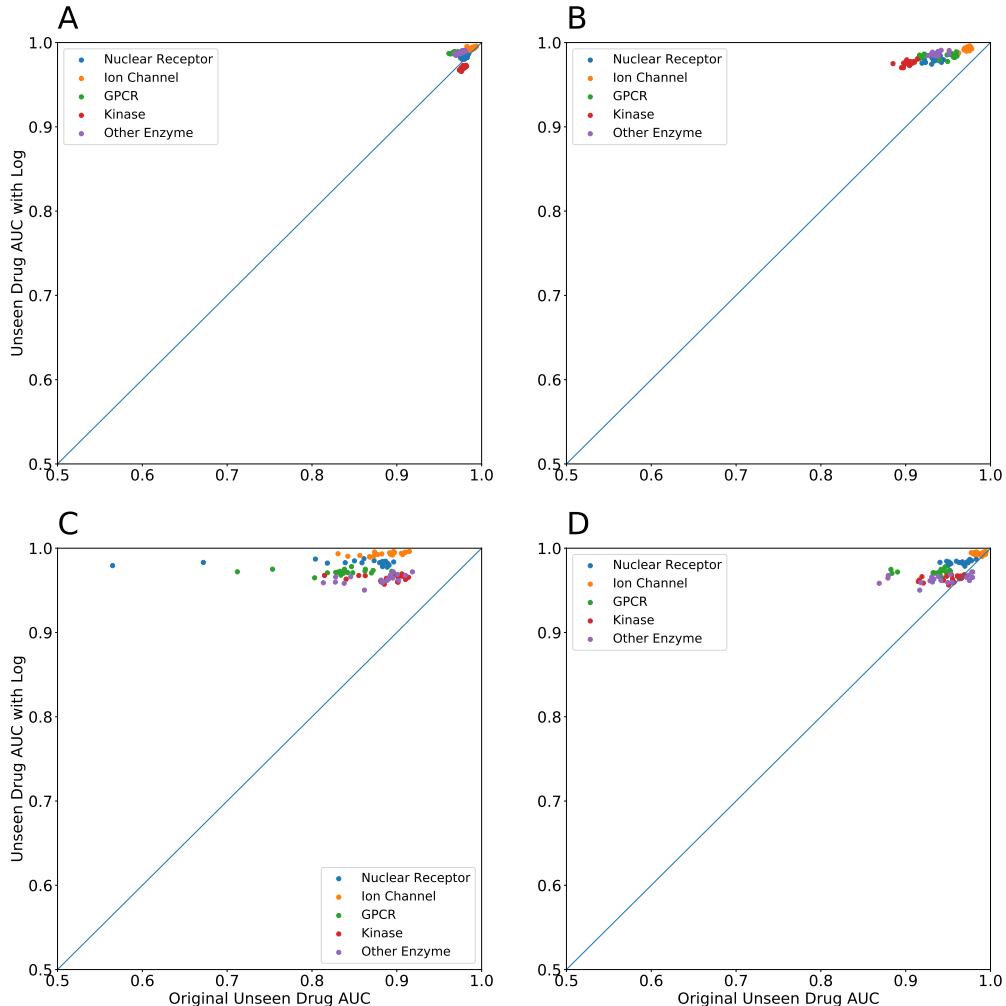


Figure 4.B.7: Comparison of Unseen Drug AUCs with and without Logistic Regression Preprocessing for (a) RF One-Hot, (b) RLS-WNN, (c) CMF, and (d) WGRMF. We see improvement on every method in almost all test runs. The high performance is likely a result of bias within the chemical dataset.

the unseen drug AUC is not significantly affected like in the nuclear receptor dataset, the drop in performance in the other AUCs is smaller; however, when the unseen drug AUC is significantly affected, the drop in performance in the other AUCs is larger, almost negating the advantages of preprocessing.

Clearly, adding information about ligand similarity to the DTI model worsens the performance of the overall model. This is likely because logistic regression and ran-

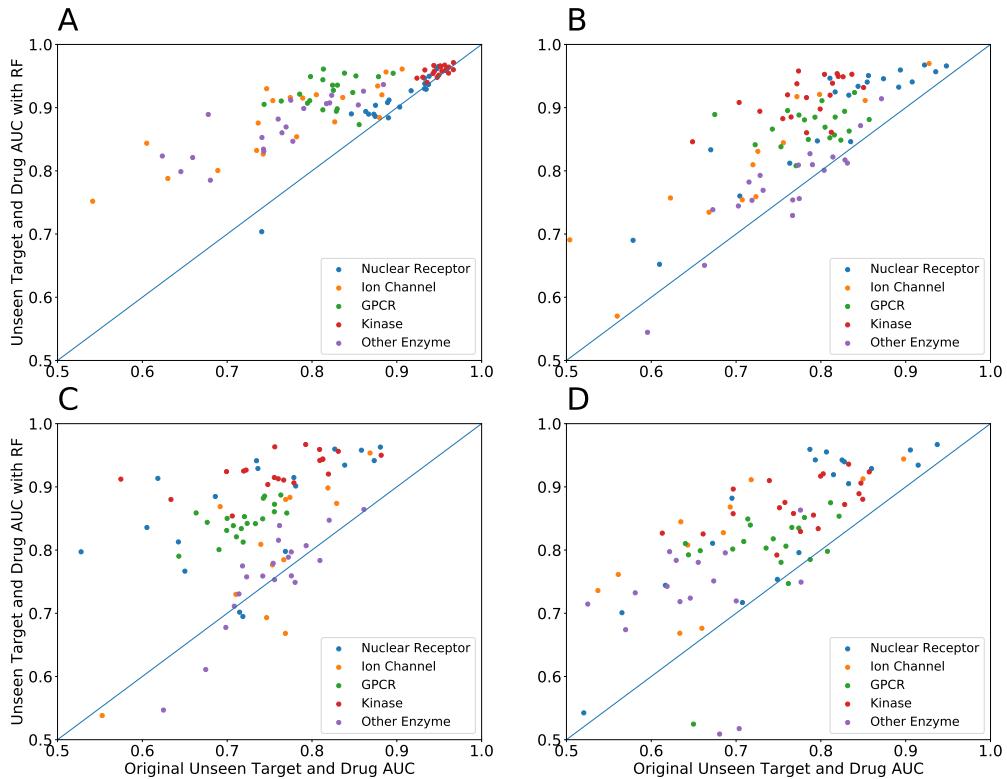


Figure 4.B.8: Comparison of Unseen Drug and Target AUCs with and without Random Forest Preprocessing for (a) RF One-Hot, (b) RLS-WNN, (c) CMF, and (d) WGRMF. We see improvement on every method in almost all test runs. The wide variance in performance is likely due to randomness in which proteins are held-out; our method also decreases this variance.

dom forest are more accurate methods of determining similarity between ligands than the Tanimoto similarity matrix. Incorporation of the Tanimoto ligand similarity matrix corrupts the information provided by our preprocessing method and lowers accuracy of the models overall. It is therefore important to ensure that DTI methods do not incorporate additional inaccurate information about the ligands; when our method is used, they should be used exclusively for generalisation in target space.

4.B.4 Probabilities Predicted in Training Submatrix

As mentioned in the main text, one of the primary explanations behind the added predictive power of our preprocessing method is the additional information provided by filling in the training matrix for all of the train proteins. We show a histogram of

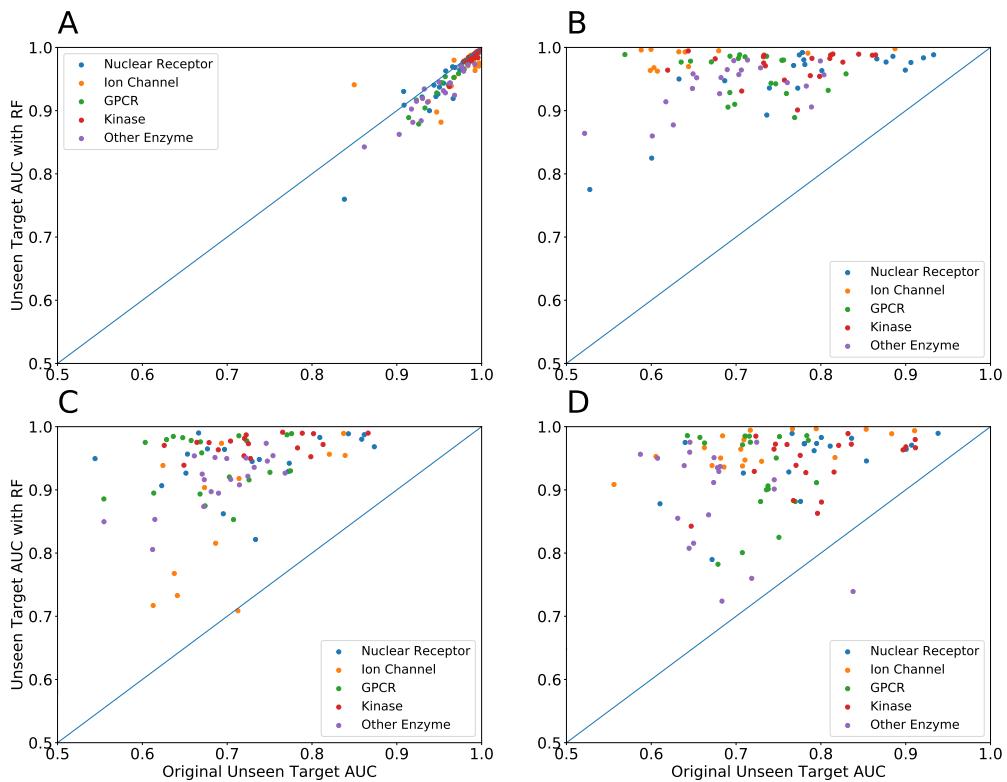


Figure 4.B.9: Comparison of Unseen Target AUCs with and without Random Forest Preprocessing for (a) RF One-Hot, (b) RLS-WNN, (c) CMF, and (d) WGRMF. We see improvement on every method in almost all test runs for models aside from RF One-Hot. The wide variance in performance is likely due to randomness in which proteins are held-out; our method also decreases this variance.

the probabilities of interaction predicted by our preprocessing step for both logistic regression and random forest in Fig. 4.B.12 for all other datasets (aside from GPCR, which was shown in the main text).

As with the GPCR dataset, we see a number of active interactions predicted that were not in the original training matrix. This added information helps the DTI models train more effectively and make more accurate predictions. We also see similar issues with random forest calibration as on the GPCR dataset, suggesting that recalibrating the random forest model could improve overall performance.

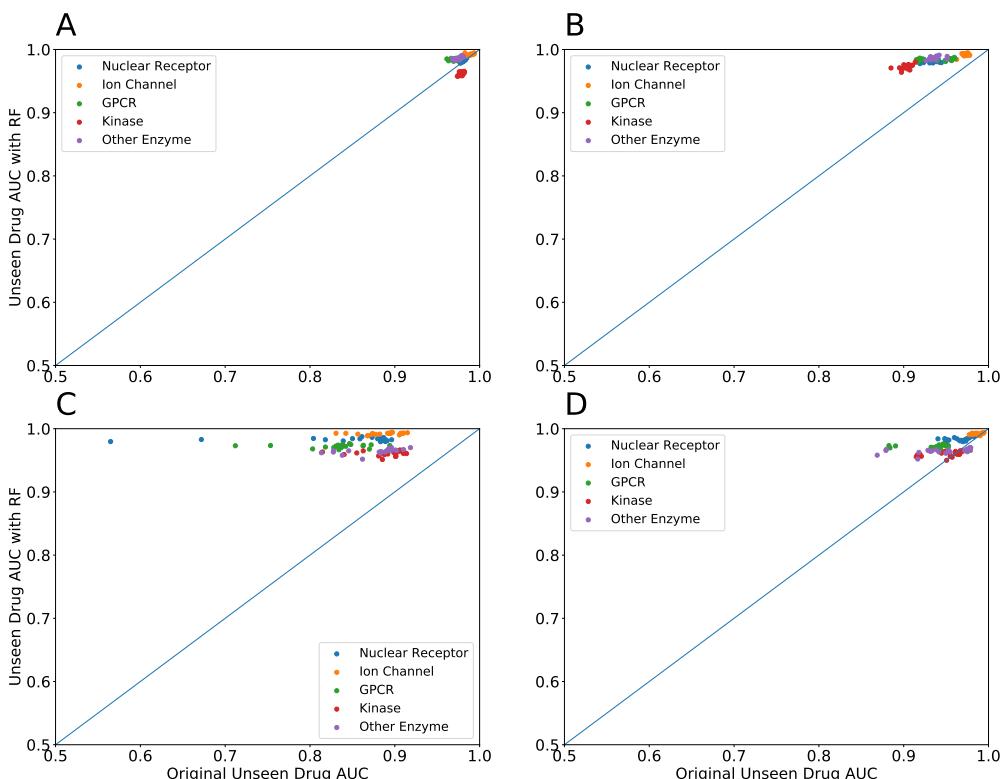


Figure 4.B.10: Comparison of Unseen Drug AUCs with and without Random Forest Preprocessing for (a) RF One-Hot, (b) RLS-WNN, (c) CMF, and (d) WGRMF. We see improvement on every method in almost all test runs. The high performance is likely a result of bias within the chemical dataset.

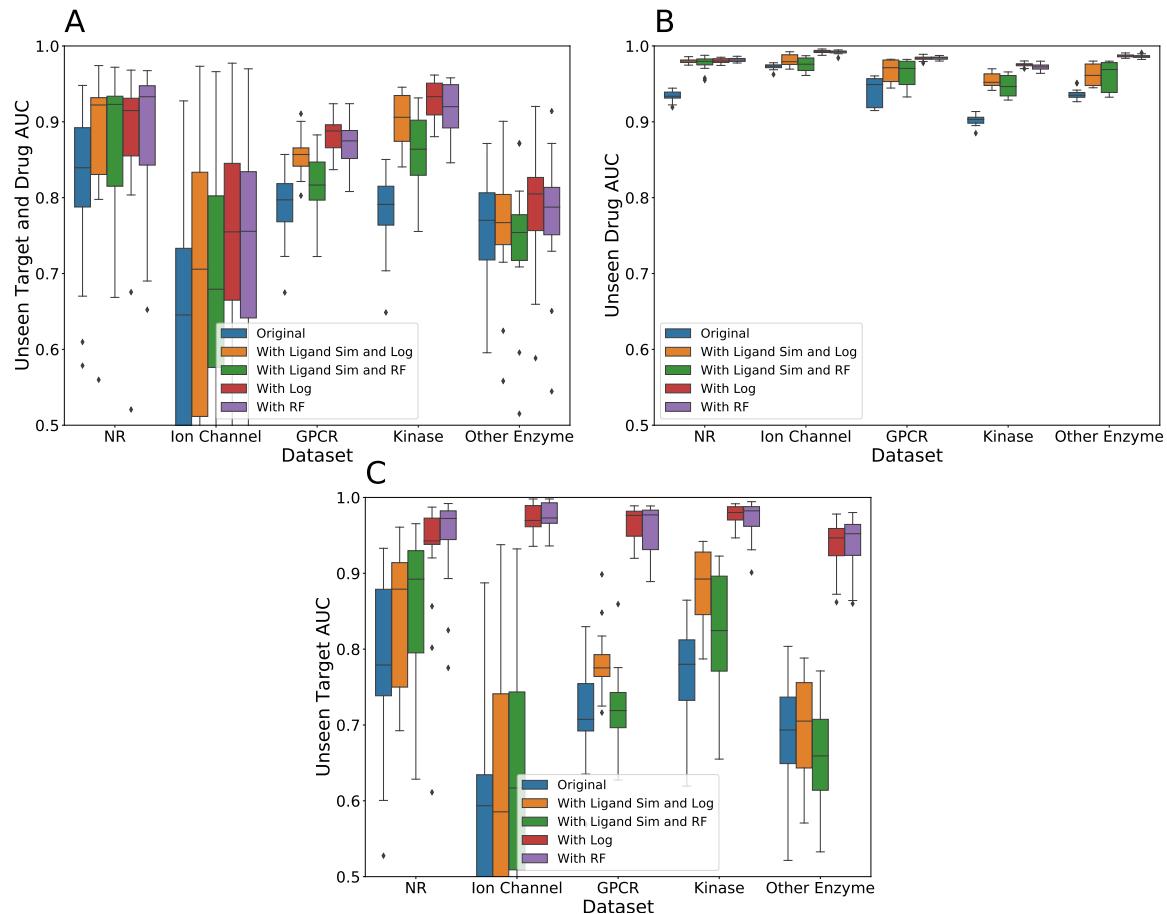


Figure 4.B.11: Effect of Not Eliminating Ligand Similarity on RLS for (a) Unseen Target and Drug AUC, (b) Unseen Drug AUC, and (c) Unseen Target AUC. Models that incorrectly incorporate ligand similarity into the RLS algorithm consistently perform worse than those that do not, emphasizing the need to eliminate ligand similarity in the DTI model after our preprocessing.

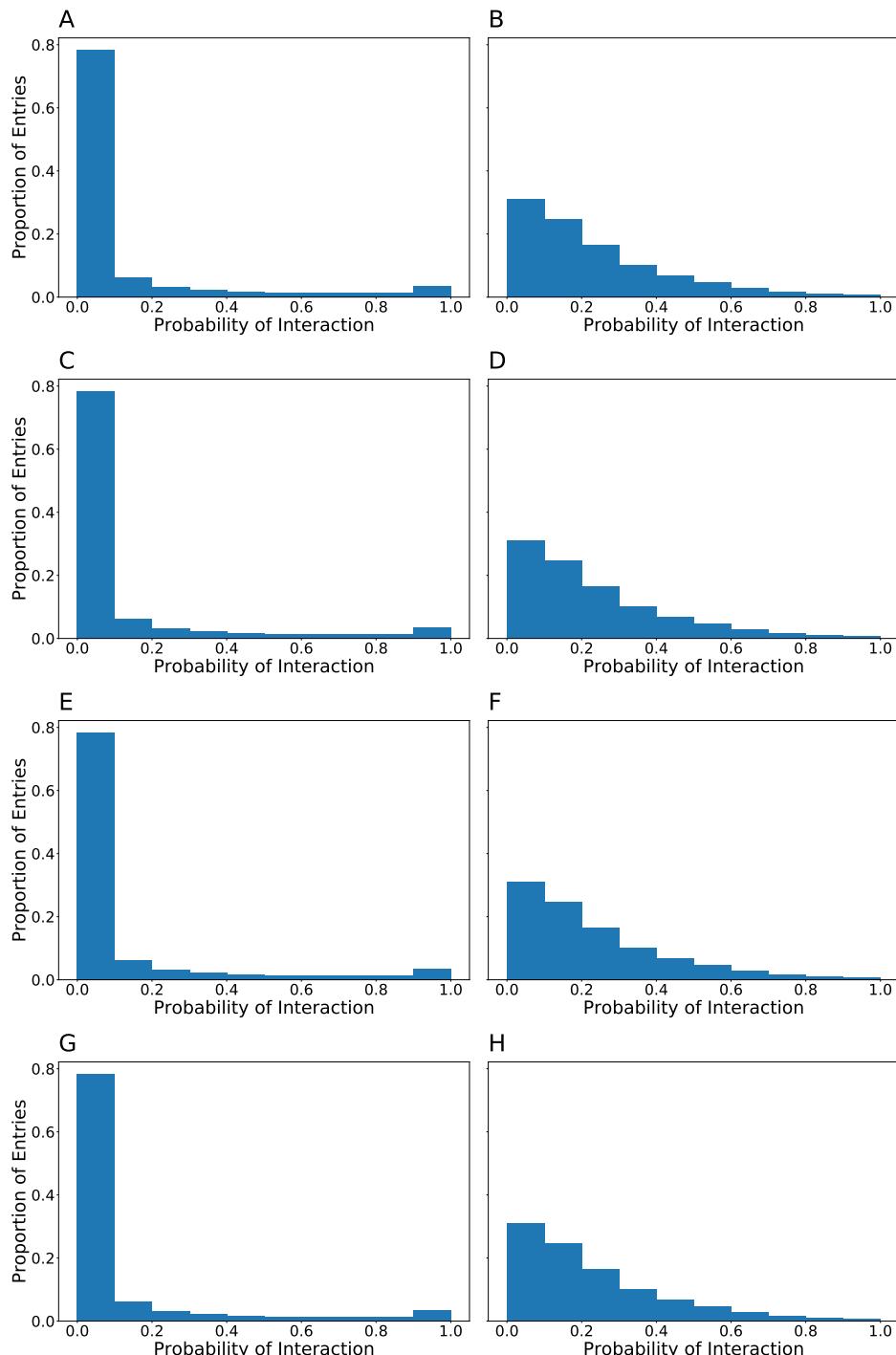


Figure 4.B.12: Histogram of Added Information to Training Matrix for (a-b) Nuclear Receptor, (c-d) Ion Channel, (e-f) Kinase, and (g-h) Other Enzyme Datasets when Preprocessed by (a, c, e, g) Logistic Regression and (b, d, f, h) Random Forest. This shows the predicted probabilities of interactions that are added to the training matrix by the appropriate single protein/ligand binding model. Most of the interactions are predicted to be inactive, but some are predicted to be active. We hypothesize that this additional data helps explain some of the improvement in the unseen target and drug and unseen target problems.

Chapter 5

Conclusion

In this thesis, we have examined some of the pitfalls and issues that arise in machine learning applied to protein/ligand binding as well as ways to avoid them. Chapter 2 showed that many standard ML models do not generalise well and that debiasing algorithms that eliminate clustering in datasets do not aid in generalisation. Chapter 3 developed methods to measure a model’s ability to attribute or explain its predictions and found that many models consistently learn the wrong binding logic; we demonstrated that this was likely due to spurious correlations in the background data that confounded our models. Chapter 4 introduced a new method of preprocessing using single protein models to improve the accuracy of DTI models when generalising to data about previously unseen proteins.

One of the key problems for ML virtual screening models today is generalisation, in both protein and ligand space. In chemical space, our results suggest a few potential avenues for further research. We could adopt applicability domains to sidestep the question of generalisation entirely. Alternatively, we could gain a better understanding of the background correlation that is likely confounding our models’ attempts at generalising and thereby develop more diverse fragment libraries. Our results suggest that incorporation of fragment-matched decoys, i.e. molecules that have some but not all similar fragments, will help models learn more effectively. Similarly, lowering the background correlation rate due to clustering around scaffolds may help our models generalise better. In contrast to these, the RMT-based approaches and debiasing approaches previously used seem to be ineffective for this problem.

Related to the problem of generalisation in chemical space is the issue of attribution.

Chapter 5 Conclusion

Our results indicate that understanding background correlation is crucial to eliminating misattributions in our models, as is a more diverse background library so models see a larger variety of molecules. These steps should make it more reasonable for medicinal chemists to draw mechanistic conclusions from the output of ML virtual screening models.

Lastly, we have the issue of generalisation in protein space. Our results suggest that the key is to use simple, robust single protein models to make predictions throughout chemical space and then generalise in protein space. This step significantly augments performance on the unseen target and drug problem, beyond any change in the DTI model used. However, our high performance was only demonstrated on randomly split data, not accounting for bias in either chemical or protein space. Further insights are needed to better incorporate information about the protein targets and create better-performing DTI models to fully tackle this problem.

Bibliography

- [1] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity Checks for Saliency Maps. *Advances in Neural Information Processing Systems*, 31, 2018.
- [2] Han Altae-Tran, Bharath Ramsundar, Aneesh S. Pappu, and Vijay Pande. Low Data Drug Discovery with One-Shot Learning. *ACS Central Science*, 3(4):283–293, 2017.
- [3] Pedro J. Ballester and John B. O. Mitchell. A Machine Learning Approach to Predicting Protein-Ligand Binding Affinity with Applications to Molecular Docking. *Bioinformatics*, 26(9):1169–1175, 2010.
- [4] Pedro J Ballester, Adrian Schreyer, and Tom L Blundell. Does a More Precise Chemical Description of Protein-Ligand Complexes Lead to More Accurate Prediction of Binding Affinity? *Journal of Chemical Information and Modeling*, 54(3):944–55, 2014.
- [5] Tolga Bolukbasi, Kai-Wei Chang, James Zou, Venkatesh Saligrama, and Adam Kalai. Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings. *Advances in Neural Information Processing Systems*, pages 4356–4364, 2016.
- [6] Henrik Boström. Calibrating random forests. *Proceedings - 7th International Conference on Machine Learning and Applications, ICMLA 2008*, pages 121–126, 2008.
- [7] Brandon Carter, Jonas Mueller, Siddhartha Jain, and David Gifford. What Made You Do This? Understanding Black-Box Decisions with Sufficient Input Subsets. *Proceedings of Machine Learning Research*, 89:567–576, 2019.

Bibliography

- [8] Adrià Cereto-Massagué, María José Ojeda, Cristina Valls, Miquel Mulero, Santiago Garcia-Vallvé, and Gerard Pujadas. Molecular Fingerprint Similarity Search in Virtual Screening. *Methods*, 71:58–63, 2015.
- [9] Ruolan Chen, Xiangrong Liu, Shuting Jin, Jiawei Lin, and Juan Liu. Machine Learning for Drug-Target Interaction Prediction. *Molecules*, 23(9):1–15, 2018.
- [10] Yu-Chian Chen. Beware of Docking! *Trends in Pharmacological Sciences*, 36(2):78–95, 2015.
- [11] Ann E Cleves and Ajay N Jain. Effects of Inductive Bias on Computational Evaluations of Ligand-Based Modeling and on Drug Discovery. *Journal of computer-aided molecular design*, 22(3-4):147–59, 2008.
- [12] Murat Can Cobanoglu, Chang Liu, Feizhuo Hu, Zoltan N. Oltvai, and Ivet Bahar. Predicting Drug-Target Interactions Using Probabilistic Matrix Factorization Predicting Drug Target Interactions Using Probabilistic Matrix. *Journal of Chemical Information and Modeling*, 53(12):3399–3409, 2013.
- [13] Lucy J. Colwell. Statistical and Machine Learning Approaches to Predicting Protein-Ligand Interactions. *Current Opinion in Structural Biology*, 49:123–128, 2018.
- [14] Isidro Cortés-Ciriano, Qurrat Ul Ain, Vigneshwari Subramanian, Eelke B. Lenselink, Oscar Mendez-Lucio, Adriaan P. IJzerman, Gerd Wohlfahrt, Peteris Prusis, Therese E. Malliaivin, Gerard J.P. van Westen, and Andreas Bender. Polypharmacology Modelling using Proteochemometrics (PCM): Recent Methodological Developments, Applications to Target Families, and Future Prospects. *Medicinal Chemistry Communications*, 6(1):24–50, 2015.
- [15] Mark Davies, Michał Nowotka, George Papadatos, Nathan Dedman, Anna Gaulton, Francis Atkinson, Louisa Bellis, and John P. Overington. ChEMBL Web Services: Streamlining Access to Drug Discovery Data and Utilities. *Nucleic Acids Research*, 43(W1):W612–W620, 2015.
- [16] Joseph L. Durant, Burton A. Leland, Douglas R. Henry, and James G. Nourse.

Bibliography

- Reoptimization of MDL Keys for Use in Drug Discovery. *Journal of Chemical Information and Computer Sciences*, 42(6):1273–1280, 2002.
- [17] David K. Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alan Aspuru-Guzik, and Ryan P. Adams. Convolutional Networks on Graphs for Learning Molecular Fingerprints. *Advances in Neural Information Processing Systems*, 28:2224–2232, 2015.
- [18] Sean Eddy. HMMER 3.1b2.
- [19] Ali Ezzat, Min Wu, Xiao-Li Li, and Chee-Keong Kwoh. Computational Prediction of Drug-Target Interactions using Chemogenomic Approaches: An Empirical Survey. *Briefings in Bioinformatics*, pages 1–21, 2018.
- [20] Ali Ezzat, Peilin Zhao, Min Wu, Xiao-Li Li, and Chee-Keong Kwoh. Drug-Target Interaction Prediction with Graph Regularized Matrix Factorization. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 14(3):646–656, 2017.
- [21] Anna Gaulton, Anne Hersey, Micha L. Nowotka, A. Patricia Bento, Jon Chambers, David Mendez, Prudence Mutowo, Francis Atkinson, Louisa J. Bellis, Elena Cibrian-Uhalte, Mark Davies, Nathan Dedman, Anneli Karlsson, Maia Paula Margarinos, John P. Overington, George Papadatos, Ines Smit, and Andrew R. Leach. The ChEMBL database in 2017. *Nucleic Acids Research*, 45(D1):D945–D954, 2017.
- [22] Erik Gawehn, Jan A. Hiss, and Gisbert Schneider. Deep Learning in Drug Discovery. *Molecular Informatics*, 35(1):3–14, 2016.
- [23] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural Message Passing for Quantum Chemistry. *Proceedings of the 34th International Conference on Machine Learning*, 70:1263–1272, 2017.
- [24] Joseph Gomes, Bharath Ramsundar, Evan N. Feinberg, and Vijay S. Pande. Atomic Convolutional Networks for Predicting Protein-Ligand Binding Affinity. *arXiv Preprint*, (arXiv:1703.10603), 2017.
- [25] Mehmet Gönen. Predicting Drug-Target Interactions from Chemical and Genomic Kernels using Bayesian Matrix Factorization. *Bioinformatics*, 28(18):2304–2310, 2012.

Bibliography

- [26] Andrew C. Good and Tudor I. Oprea. Optimization of CAMD Techniques 3. Virtual Screening Enrichment Studies: A Help or Hindrance in Tool Selection? *Journal of Computer-Aided Molecular Design*, 22(3-4):169–178, 2008.
- [27] Sam Grinter and Xiaoqin Zou. Challenges, Applications, and Recent Advances of Protein-Ligand Docking in Structure-Based Drug Design. *Molecules*, 19(7):10150–10176, 2014.
- [28] Martin Gütlein and Stefan Kramer. Filtered Circular Fingerprints Improve Either Prediction Or Runtime Performance while Retaining Interpretability. *Journal of Cheminformatics*, 8(1):1–16, 2016.
- [29] Kazunari Hattori, Hiroaki Wakabayashi, and Kenta Tamaki. Predicting Key Example Compounds in Competitors' Patent Applications Using Structural Information Alone. *Journal of Chemical Information and Modeling*, 48(1):135–142, 2008.
- [30] R P Hertzberg and A J Pope. High-Throughput Screening: New Technology for the 21st Century. *Current Opinion in Chemical Biology*, 4(4):445–51, 2000.
- [31] Fergus Imrie, Anthony R. Bradley, Mihaela van der Schaar, and Charlotte M. Deane. Protein Family-Specific Models Using Deep Neural Networks and Transfer Learning Improve Virtual Screening and Highlight the Need for More Data. *Journal of Chemical Information and Modeling*, 58(11):2319–2330, 2018.
- [32] John J. Irwin, Teague Sterling, Michael M. Mysinger, Erin S. Bolstad, and Ryan G. Coleman. ZINC: A Free Tool to Discover Chemistry for Biology. *Journal of Chemical Information and Modeling*, 52(7):1757–1768, 2012.
- [33] Ajay N Jain and Ann E Cleves. Does Your Model Weigh the Same as a Duck? *Journal of Computer-Aided Molecular Design*, 26(1):57–67, 2012.
- [34] Eric Jones, Travis Oliphant, Pearu Peterson, and Others. SciPy: Open Source Scientific Tools for Python, 2001.
- [35] Steven Kearnes, Kevin McCloskey, Marc Berndl, Vijay Pande, and Patrick Riley. Molecular Graph Convolutions: Moving Beyond Fingerprints. *Journal of Computer-Aided Molecular Design*, 30(8):595–608, 2016.

Bibliography

- [36] Sunghwan Kim, Paul A. Thiessen, Evan E. Bolton, Jie Chen, Gang Fu, Asta Gindulyte, Lianyi Han, Jane He, Siqian He, Benjamin A. Shoemaker, Jiyao Wang, Bo Yu, Jian Zhang, and Stephen H. Bryant. PubChem Substance and Compound Databases. *Nucleic Acids Research*, 44(D1):D1202–D1213, 2016.
- [37] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial Machine Learning at Scale. *International Conference on Learning Representations*, 2017.
- [38] Lukasz Kurgan and Fatemeh Miri Disfani. Structural Protein Descriptors in 1-Dimension and their Sequence-Based Predictions. *Current Protein & Peptide Science*, 12(6):470–89, 2011.
- [39] Greg Landrum. RDKit: Open-Source Cheminformatics, 2006.
- [40] Alpha A. Lee, Michael P. Brenner, and Lucy J. Colwell. Predicting Protein-Ligand Affinity with a Random Matrix Framework. *Proceedings of the National Academy of Sciences*, 113(48):13564–13569, 2016.
- [41] Alpha A Lee, Qingyi Yang, Asser Bassyouni, Christopher R Butler, Xinjun Hou, Stephen Jenkinson, and David A Price. Ligand Biological Activity Predicted by Cleaning Positive and Negative Chemical Correlations. *Proceedings of the National Academy of Sciences of the United States of America*, 116(9):3373–3378, 2019.
- [42] Ingoo Lee, Jongsoo Keum, and Hojung Nam. DeepConv-DTI: Prediction of Drug-Target Interactions via Deep Learning with Convolution on Protein Sequences. *PLoS Computational Biology*, 15(6):e1007129, 2019.
- [43] Shengchao Liu, Moayad Alnammi, Spencer S Erickson, Andrew F Voter, Gene E Ananiev, James L Keck, F Michael Hoffmann, Scott A Wildman, Anthony Gitter, and Anthony Gitter. Practical Model Selection for Prospective Virtual Screening. *Journal of Chemical Information and Modeling*, 59(1):282–293, 2019.
- [44] Junshui Ma, Robert P. Sheridan, Andy Liaw, George E. Dahl, and Vladimir Svetnik. Deep Neural Nets as a Method for Quantitative Structure-Activity Relationships. *Journal of Chemical Information and Modeling*, 55(2):263–274, 2015.

Bibliography

- [45] Andrew B MacConnell, Alexander K Price, and Brian M Paegel. An Integrated Microfluidic Processor for DNA-Encoded Combinatorial Library Functional Screening. *ACS Combinatorial Science*, 19(3):181–192, 2017.
- [46] Gerald Maggiora, Martin Vogt, Dagmar Stumpfe, and Jürgen Bajorath. Molecular Similarity in Medicinal Chemistry. *Journal of Medicinal Chemistry*, 57(8):3186–3204, 2014.
- [47] Kevin McCloskey, Ankur Taly, Federico Monti, Michael P. Brenner, and Lucy Colwell. Using Attribution to Decode Binding Mechanism in Neural Network Models for Chemistry. *Proceedings of the National Academy of Sciences of the United States of America*, 116(24):11624–11629, 2019.
- [48] Lewis H. Mervin, Krishna C. Bulusu, Leen Kalash, Avid M. Afzal, Fredrik Svensson, Mike A. Firth, Ian Barrett, Ola Engkvist, and Andreas Bender. Orthologue Chemical Space and Its Influence on Target Prediction. *Bioinformatics*, 34(1):72–79, 2018.
- [49] Michael M. Mysinger, Michael Carchia, John. J. Irwin, and Brian K. Shoichet. Directory of Useful Decoys, Enhanced (DUD-E): Better Ligands and Decoys for Better Benchmarking. *Journal of Medicinal Chemistry*, 55(14):6582–6594, 2012.
- [50] Steven M. Paul, Daniel S. Mytelka, Christopher T. Dunwiddie, Charles C. Persinger, Bernard H. Munos, Stacy R. Lindborg, and Aaron L. Schacht. How to Improve R&D Productivity: the Pharmaceutical Industry’s Grand Challenge. *Nature Reviews Drug Discovery*, 9(3):203–214, 2010.
- [51] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [52] Tianyi Qiu, Jingxuan Qiu, Jun Feng, Dingfeng Wu, Yiyan Yang, Kailin Tang, Zhiwei Cao, and Ruixin Zhu. The Recent Progress in Proteochemometric Modelling:

Bibliography

- Focusing on Target Descriptors, Cross-Term Descriptors and Application Scope. *Briefings in Bioinformatics*, 18(1):125–136, 2017.
- [53] Matthew Ragoza, Joshua Hochuli, Elisa Idrobo, Jocelyn Sunseri, and David Ryan Koes. ProteinLigand Scoring with Convolutional Neural Networks. *Journal of Chemical Information and Modeling*, 57(4):942–957, 2017.
- [54] Bharath Ramsundar, Steven Kearnes, Patrick Riley, Dale Webster, David Konerding, and Vijay Pande. Massively Multitask Networks for Drug Discovery. *arXiv Preprint*, (arXiv:1502.02072), 2015.
- [55] Bharath Ramsundar, Bowen Liu, Zhenqin Wu, Andreas Verras, Matthew Tudor, Robert P. Sheridan, and Vijay Pande. Is Multitask Deep Learning Practical for Pharma? *Journal of Chemical Information and Modeling*, 57(8):2068–2076, 2017.
- [56] H. B. Rao, F. Zhu, G. B. Yang, Z. R. Li, and Y. Z. Chen. Update of PROFEAT: a Web Server for Computing Structural and Physicochemical Features of Proteins and Peptides from Amino Acid Sequence. *Nucleic Acids Research*, 39(Web Server Issue):W385–W390, 2011.
- [57] Ahmet Sureyya Rifaioglu, Heval Atas, Maria Jesus Martin, Rengul Cetin-Atalay, Volkan Atalay, and Tunca Dogan. Recent Applications of Deep Learning and Machine Intelligence on In Silico Drug Discovery: Methods, Tools, and Databases. *Briefings in Bioinformatics*, pages 1–35, 2018.
- [58] David Rogers and Mathew Hahn. Extended-Connectivity Fingerprints. *Journal of Chemical Information and Modeling*, 50(5):742–754, 2010.
- [59] Sebastian G. Rohrer and Knut Baumann. Maximum Unbiased Validation (MUV) Data Sets for Virtual Screening Based on PubChem Bioactivity Data. *Journal of Chemical Information and Modeling*, 49(2):169–184, 2009.
- [60] Gisbert Schneider. Automating Drug Discovery. *Nature Reviews Drug Discovery*, 17(2):97–113, 2018.
- [61] Robert P. Sheridan. Time-Split Cross-Validation as a Method for Estimating the Goodness of Prospective Prediction. *Journal of Chemical Information and Modeling*, 53(4):783–790, 2013.

Bibliography

- [62] Jochen Sieg, Florian Flachsenberg, and Matthias Rarey. In Need of Bias Control: Evaluating Chemical Data for Machine Learning in Structure-Based Virtual Screening. *Journal of Chemical Information and Modeling*, 59(3):947–961, 2019.
- [63] T.F. Smith and M.S. Waterman. Identification of Common Molecular Subsequences. *Journal of Molecular Biology*, 147(1):195–197, 1981.
- [64] Vikram Sundar and Lucy Colwell. Debiasing Algorithms for Protein Ligand Binding Data do not Improve Generalisation. *ChemRxiv*, 2019.
- [65] Vladimir Svetnik, Andy Liaw, Christopher Tong, J. Christopher Culberson, Robert P. Sheridan, and Bradley P. Feuston. Random Forest: A Classification and Regression Tool for Compound Classification and QSAR Modeling. *Journal of Chemical Information and Computer Science*, 43(6):1947–1958, 2003.
- [66] Thomas Unterthiner, Andreas Mayr, Günter Klambauer, Marvin Steljaert, Jörg Wenger, Hugo Ceulemans, and Sepp Hochreiter. Deep Learning as an Opportunity in Virtual Screening. *Proceedings of the Deep Learning Workshop at NIPS*, pages 1–9, 2014.
- [67] Twan van Laarhoven and Elena Marchiori. Predicting Drug-Target Interactions for New Drug Compounds Using a Weighted Nearest Neighbor Profile. *PLoS ONE*, 8(6):e66952, 2013.
- [68] Gerard JP van Westen, Remco F Swier, Jörg K Wegner, Adriaan P IJzerman, Herman WT van Vlijmen, and Andreas Bender. Benchmarking of Protein Descriptor Sets in Proteochemometric Modeling (part 1): Comparative Study of 13 Amino Acid Descriptor Sets. *Journal of Cheminformatics*, 5(1):41, 2013.
- [69] Alfredo Vellido Alcacena, Jose D. Martin Guerrero, and Paulo J.G. Lisboa. Making Machine Learning Models Interpretable. *ESANN 2012 Proceedings: European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, pages 163–172, 2012.
- [70] Marcel L. Verdonk, Valerio Berdini, Michael J. Hartshorn, Wijnand T. M. Mooij, Christopher W. Murray, Richard D. Taylor, and Paul Watson. Virtual Screening

Bibliography

- Using Protein-Ligand Docking: Avoiding Artificial Enrichment. *Journal of Chemical Information and Computer Sciences*, 44(3):793–806, 2004.
- [71] Izhar Wallach, Michael Dzamba, and Abraham Heifets. AtomNet: A Deep Convolutional Neural Network for Bioactivity Prediction in Structure-based Drug Discovery. *arXiv Preprint*, arXiv:1510, 2015.
- [72] Izhar Wallach and Abraham Heifets. Most Ligand-Based Classification Benchmarks Reward Memorization Rather than Generalization. *Journal of Chemical Information and Modeling*, 58(5):916–932, 2018.
- [73] Ming Wen, Zhimin Zhang, Shaoyu Niu, Haozhi Sha, Ruihan Yang, Yonghuan Yun, and Hongmei Lu. Deep-Learning-Based Drug-Target Interaction Prediction. *Journal of Proteome Research*, 16(4):1401–1409, 2017.
- [74] Jie Xia, Hongwei Jin, Zhenming Liu, Liangren Zhang, and Xiang Simon Wang. An Unbiased Method To Build Benchmarking Sets for Ligand-Based Virtual Screening and its Application To GPCRs. *Journal of Chemical Information and Modeling*, 54(5):1433–1450, 2014.
- [75] Yoshihiro Yamanishi, Michihiro Araki, Alex Gutteridge, Wataru Honda, and Minoru Kanehisa. Prediction of Drug-Target Interaction Networks from the Integration of Chemical and Genomic Spaces. *Bioinformatics*, 24(13):232–240, 2008.
- [76] Xiaodong Zheng, Hao Ding, Hiroshi Mamitsuka, and Shanfeng Zhu. Collaborative Matrix Factorization with Multiple Similarities for Predicting Drug-Target Interactions. *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '13*, pages 1025–1033, 2013.
- [77] David Zilian and Christoph A. Sottriffer. SFCscore(RF) : A Random Forest-Based Scoring Function for Improved Affinity Prediction of Protein-Ligand Complexes. *Journal of Chemical Information and Modeling*, 53(8):1923–1933, 2013.