Chemistry Capstone Project: *An exploration of model complexity in the drug discovery field.*

Authors: Phil Michaels and David Dunstan
University of Michigan: Masters of Applied Data Science Program
August 2022

Introduction:

It is March of 2020, and a highly contagious virus is spreading rapidly across the world and killing without any known treatments. Scientists race to first understand the biology of the virus, decipher its DNA and protein structures and then identify opportunities for small molecules to interrupt or inhibit them. This is the exact scenario that many drug discovery scientists find themselves in everyday, although instead of COVID-19 they are looking to treat HIV or Alczheimers, or cancer. The need of patients drives the drug discovery industry towards trying to find these novel treatments with the utmost speed. The potential for machine learning holds massive promise to help better predict which molecules can be used to treat these diseases.

The field of drug discovery is complex due to the interplay between biological systems, disease and drug candidates. The search begins with an efficiently designed high-throughput screen to identify potential hit compounds from large library collections. Next, these hits are expanded and chemically iterated upon to optimize their potency as well as their properties so they can be absorbed properly into the body, reduce the risk of toxic side-effects and allow them to be produced efficiently. These optimized hits, called leads, are then moved into further testing where animal models are used to confirm the results before moving towards the clinic.

The opportunity to apply modeling at all of these stages offers the potential to increase the speed, reduce the costs, and minimize the need for animal studies. Advances in high-throughput experimentation and automation are enabling access to larger and more complete datasets than ever before. A typical drug-discovery workflow involves virtually designing a set of potential drug-like compounds and then applying machine learning models based on both historical and target-specific data. These models help to predict which compounds are most likely to be successful while minimizing the risk of poor properties. Scientists in the lab then produce the selected compounds and profile them in a range of assays that empirically test the predictions. The process is iterative in that the new data fuels re-training of the models and subsequent selections until the desired outcome is achieved or sufficient data is produced to convince the scientists that alternative molecules are needed due to an adverse finding.

Project Goal:

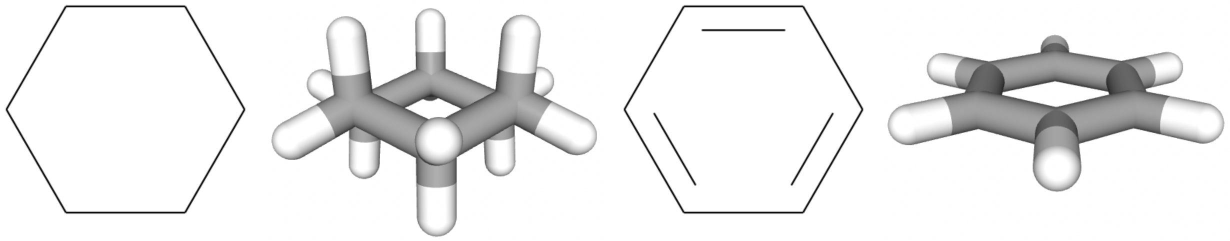
Given the increasing role that modeling is playing in chemistry and drug discovery, we wanted to assess whether the trend towards building more complex modeling approaches is contributing to increased performance when compared with more simple baselines. While there are many ways to evaluate performance of a model, we took inspiration from a recent Nature paper in which the authors suggest that the use of standardized data sets would be a way to help better compare new modeling techniques. (https://www.nature.com/articles/s41570-022-00391-9) We selected a subset of these datasets from Moleculenet (https://moleculenet.org/datasets-1) that would be typically relevant to an early drug discovery space. We then compared the results of these various models to each other across several different metrics. Another consideration for the modeling selections was also training time and the ease setup and configuration.

While there are a multitude of complex models that have been published, we chose to explore a message passing neural network package that has been developed by MIT, called Chemprop. (https://chemprop.readthedocs.io/en/latest/) In the initial paper, the authors describe and compare the Chemprop approach to a series of other models, however they looked primarily at a single metric and did not factor in other challenges that arise from utilization of complex models. (https://pubs.acs.org/doi/10.1021/acs.jcim.9b00237) Furthermore, we feel that there is opportunity to dive deeper into the predictions between the models to try and understand where performance is breaking down. For the simple models, we chose to explore a range of scikit-learn model deployments. We also utilized scikit-learn based metrics for evaluation.

Introduction to Cheminformatics

Cheminformatics is the application of computational or computer based approaches, often referred to as *in silico* methods, to chemistry and chemical structures. One of the key challenges in the cheminformatics space is the featurization, or description of a molecule such that a computer can understand it. In nature, molecules are a complex combination of electronic interactions that play out on a quantum scale in 3-dimensional space. However, despite dramatic increases in computational power the field has yet to achieve this level of modeling at scale (https://wires.onlinelibrary.wiley.com/doi/10.1002/wcms.1290) Therefore, we must rely on simpler representations in order to describe a molecule.

In many chemistry applications, structures are drawn to a 2D image, where atoms (assumed carbon unless otherwise specified) are connected by simple lines to form bonds. While this type of image can be effective for communication of structure, it often masks subtleties of how the molecules look in 3D space, or behave from an electronic perspective. For example, when comparing cyclohexane to benzene, the 2D structures are quite similar, with 6 carbon atoms connected by either single or double bonds, in 3D they appear to be drastically different.



Cyclohexane (2D) Cyclohexane (3D) Benzene (2D) Benzene (3D)

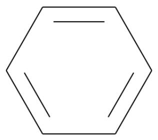
Cyclohexane SMILES: C1CCCCC1 Benzene SMILES: c1ccccc1

Figure 1: Cyclohexane compared to benzene

To get started with describing molecules computationally, we can utilize SMILES (Simplified Molecular Input Line Entry System) which uses alphanumeric characters and punctuation to describe chemical structures. While there has been some work in using SMILES and natural language processing techniques from a generative modeling perspective, the SMILES strings are not typically used for chemical property modeling.(https://link.springer.com/article/10.1007/s00894-021-04674-8) As we can see in Figure 1, despite a very similar 6 carbon atom chemical formula, the SMILES strings are quite similar, except for the capitalization. However, SMILES representations of molecules are quite prevalent for capturing the molecular structure within datasets due to their small size and textual based representation that can be easily saved to comma separated value (CSV) or text (txt) files. The 3D structure also shows the most differentiation, with cyclohexane having more 3D character than the very flat benzene ring.

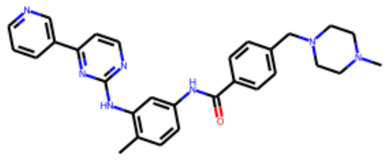
For property modeling, it is much more common to utilize fingerprint based approaches, where a molecule is broken down into simpler subsets of chemical structure and then vectorized, using either a one-hot or count based approaches. One common fingerprinting method is Morgan Fingerprints, or Extended Connectivity Fingerprint (ECFP), where a radius and bit-vector length are pre-selected and then mapped to molecular features. They are popular because they can be easily calculated, can represent a vast range of structural information and can also be interpreted back to structural features within the molecules. (https://pubs.acs.org/doi/10.1021/ci100050t) Unlike some algorithms for natural language processing such as TF-idf, the connectivity fingerprinting does not depend on the dataset of chemical structures, so the risk of data leakage into models during the featurization process is not there as there is no fitting step when generating the fingerprints.
The rough algorithm for computation of these Morgan fingerprints is the examination of an atom within a structure and then mapping the local substructure at a given radius to the bit-vector. The morgan fingerprint algorithm as well as many other relevant cheminformatics tools are all available for us in the RDkit package. (https://www.rdkit.org) For our example molecules above, if we take a radius of 2 atoms and a bit vector size of 100 (smaller than those typically utilized in models due to collisions), the representations are shown in Figure 2.

Benzene:



```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
```

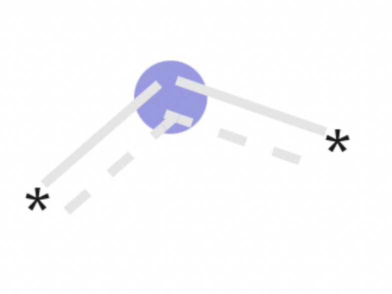
Gleevec:



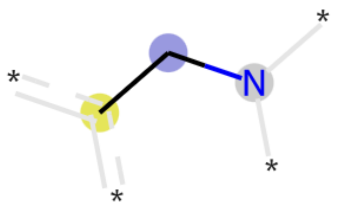
```
array([1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0,
       0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1,
       0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0,
       1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0,
       1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1])
```

Figure 2: The radius 2, 100 bit representation of Gleevec and benzene.

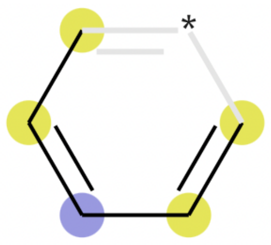
From Figure 2, we can see that the relatively simple structure of benzene is also represented by a much simpler bit-vector compared with the oncology drug Gleevec. Furthermore, we can see that these molecules share values at position 13, 69 and 84. The structural representation for bit 69, is shown in Figure 3, which is a carbon sharing two aromatic double bonds, which both molecules have. However, for bit 13, we can see that this is now a collision as two different substructures are being mapped to the same position, which is why longer bit vectors of 1024 or 2048 bits are often used.



Bit 69 (present in both molecules)



Bit 13 in Gleevec



Bit 13 in Benzene

Figure 3. Bit vector representations for bit 69 for both molecules and then bit 13 for Gleevec and then bit 13 for benzene.

Once bit-vector representations of the molecules have been generated, many other data science tasks are then possible. For instance, we can compute the Tanimoto (Jaccard) similarity between these bit vectors as follows, where a simple nitrogen substitution on pyridine yields a similarity of 0.33, while the more elaborated oncology drug, Gleevec is only 0.06, as shown in Figure 4. For this project we decided to utilize the Morgan fingerprint as our primary means to featurize the molecules. An area for future work could be to expand this approach to utilize multiple different fingerprint types, as well as compare some of the proprietary ones which are only available within licensed software to determine if they provide improved model performance.

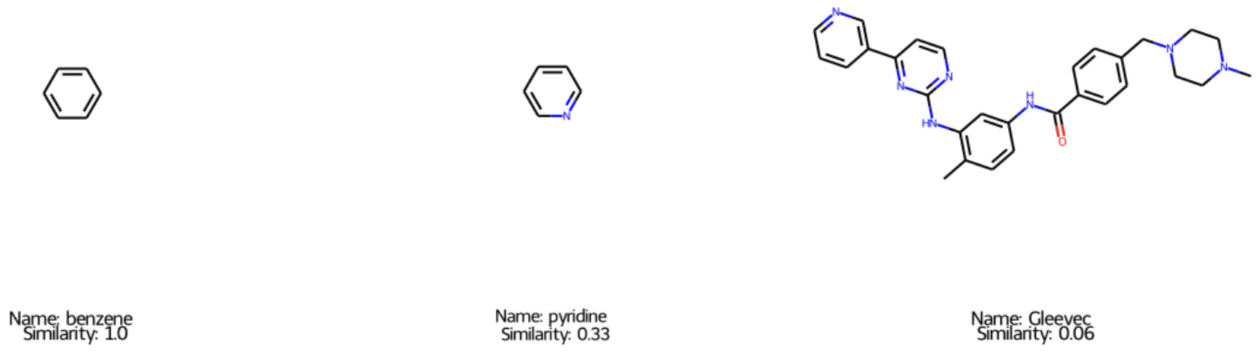


Figure 4. The similarity metrics for three molecules compared with benzene.

Datasets:

For the comparison of the various model types, we decided to select several datasets that would be relevant to the early drug discovery space. Considering the conclusions of Bender et al. (https://www.nature.com/articles/s41570-022-00391-9 same nature as above), we selected a subset of datasets which were suggested for evaluation. As a result, the datasets that we have are all publicly available and have been utilized in multiple machine learning evaluations and projects. Therefore, there are few ethical concerns for the utilization of these datasets within our project. Any ethical considerations for this project are then the result of the mis-application of the resulting models, or from any erroneous results, where our models may not be performing as expected. We therefore caution that all results here are made in an attempt to compare different model types to one another and not to utilize the resulting models for prediction of physicochemical or activity related predictions in a drug-discovery context.

We identified several datasets from Moleculenet (https://moleculenet.org/datasets-1) which is an OpenSource project from Stanford university, with the aim of comparing performance of different model types. These datasets were also employed by Bendet et al. suggesting that they have general acceptance as benchmarks within the field. However, we did diverge slightly from these other works in that we also converted all of the datasets to a classification target in order to better compare the resulting models to one another, and to limit the scope of the project. For the datasets that were sources as regression targets, we utilized generally accepted literature guidance to assign a binary target value. While there are a multitude of regression modeling problems that would be beneficial for drug discovery, we felt that classification metrics would provide a better degree of interpretability and homogeneity. Overall, we feel that these datasets represent a good range of different size, class balance, and difficulty.

BACE Dataset: The BACE dataset is a set of compounds with their BACE (an enzyme involved in the formation of neurodegenerative amyloid plaques) activity. In order to convert this problem to a classification task, a generally accepted cut-off of pIC50 values > 6 was employed to separate "active" compounds, which inhibit the target, from "inactive" compounds, which do not show any effect. (https://www.collaborativedrug.com/what-is-pic50-2/)

Dataset Size: Small (1513 molecules)
Class Balance: Balanced (1012 positive class : 501 negative class)

Tox 21 Dataset: The Tox21 dataset is a collection of compounds and their activity across a range of different toxicity relevant assays. (https://tripod.nih.gov/tox/assays) While there a multitude of different assays here, we decided to focus the dataset on the '*NR-AhR*' assay which is a liver toxicity assay, due to the data completeness, relevance to early drug discovery, and broad applicability to a number of different disease areas.

Dataset Size: Medium (7831 molecules)
Class Balance: Unbalanced (768 positive class : 5781 negative class)

Clintox Dataset: The Clintox dataset is a list from the FDA of drugs and whether or not they have shown toxicity in patients. This dataset, while small, represents some of the most relevant data for human patients.

Dataset Size: Small (1484 molecules)
Class Balance: Unbalanced (112 positive class : 1372 negative class)

Solubility Delaney Dataset: The solubility dataset is a measure of how much of a given compound can be dissolved in water, which is an important metric to optimize for any drug to ensure it is able to be properly absorbed. In order to convert this to a classification problem, a solubility measure of 0.1M was utilized to determine if a compound should be classified as soluble or insoluble. (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3399483/)

Dataset Size: Small (1128 molecules)
Class Balance: Unbalanced (185 positive class : 943 negative class)

Deepchem Lipophilicity Dataset: The lipophilicity (how strongly a molecule is attracted to grease compared with water) is also an important property for drug discovery. In order to be properly absorbed into a cell, as well as avoiding potential clearance or toxicity issues, a molecule should have a lipophilicity between -0.4 and 5.6.

(https://en.wikipedia.org/wiki/Lipinski%27s_rule_of_five) Therefore, we applied these cut-offs to convert this initial dataset into a classification one. Interestingly, after classification, this dataset was imbalanced, but in favor of the positive class.

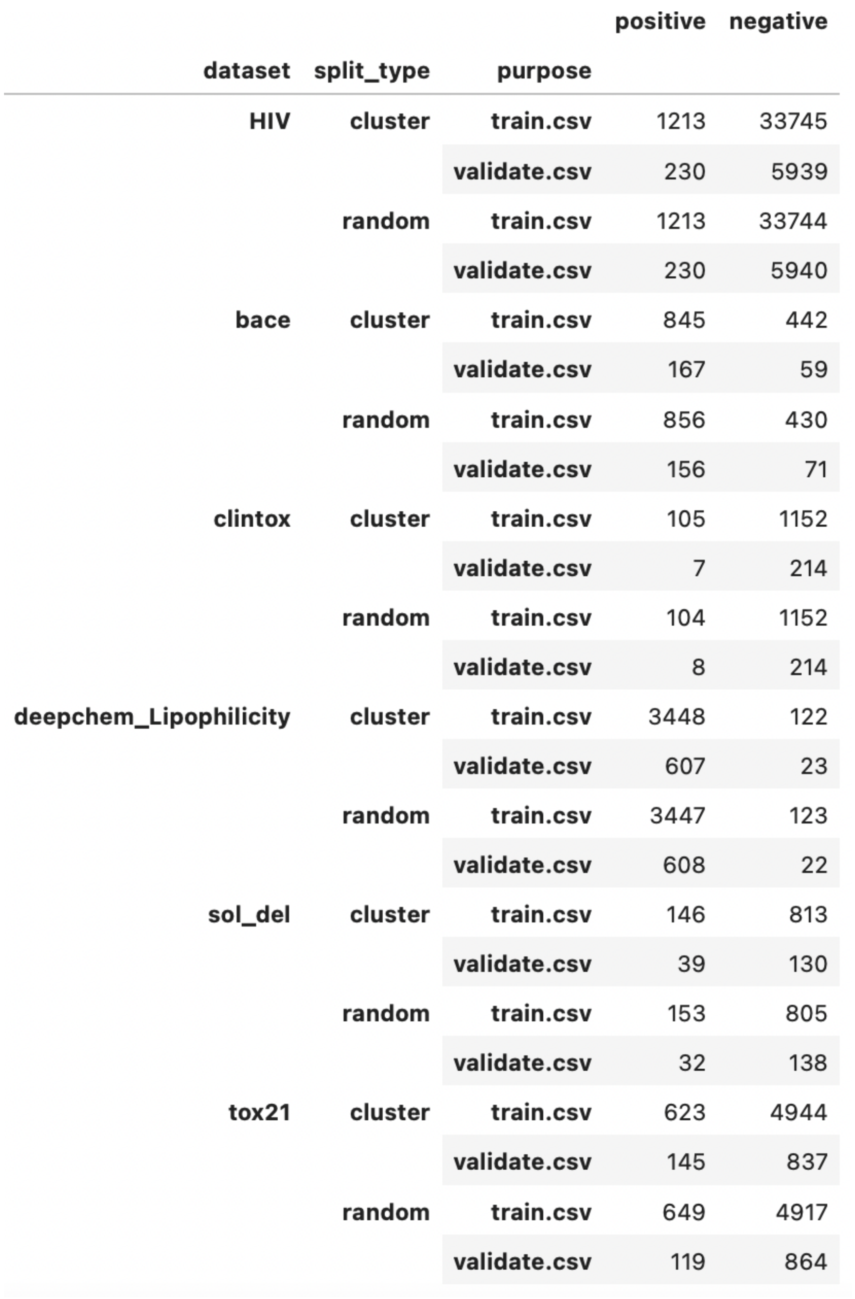
Dataset Size: Medium (4,200 molecules)
Class Balance: Unbalanced (4,055 positive class : 145 negative class)

HIV Dataset: The HIV dataset is a set of molecules and their reported activity to inhibit HIV replication in a biochemical assay. The dataset was already a classification problem, but represents a relatively large dataset for early drug discovery.

Dataset Size: Very Large (41,127 molecules)
Class Balance: Unbalanced (1443 positive class : 39684 negative class)

Dataset Splitting: For the comparison of chemical modeling approaches, we opted to utilize a cross-validation approach to model training and evaluation. Therefore, each dataset was only split into a training set and a withheld validation set for final performance evaluation. A key challenge in the area of chemical modeling is the idea of a chemical scaffold. When training a machine learning model, the model attempts to learn the various features of the molecules, however if there molecules within a dataset are very homogeneous in terms of a chemical structure, with only a few minor modifications, we risk the model learning to search for a broad structure, instead of learning generalizable features. While not true data-leakage, it can limit the applicability of a model to new chemical structures. Typically within a drug discovery project there is a desire to have models that generalize well in order to facilitate exploration and testing of new scaffold types, and therefore limit the risk of unexpected later findings, such as toxicity.

In order to address the problem of scaffold balance we adopted an unsupervised method of K-nearest neighbors (KNN) clustering to help split the dataset such that the same chemical scaffolds are not fully represented in both the training and validation data. We also performed a more traditional random split for comparison. It should be noted that there are also more elaborated approaches, such as the scaffold based split available in Chemprop (https://chemprop.readthedocs.io/en/latest/_modules/chemprop/data/scaffold.html) and while potentially effective, we found that the KNN approach was quick to execute and effective. One risk that we were aware of was that the KNN approach might alter the class balance of the various datasets relative to the random split. However, from Figure 5, we can observe that the class balance is roughly the same between approaches and the validation sets share the same class balance as the training set. In order to assign the number of clusters, we explored some metrics such as silhouette score, but found that dividing the total data set size by 30 produced clusters that were small enough to enable roughly even class balance, while still producing clusters that were grouped by chemical substructure.



dataset	split_type	purpose	positive	negative
HIV	cluster	train.csv	1213	33745
		validate.csv	230	5939
	random	train.csv	1213	33744
		validate.csv	230	5940
bace	cluster	train.csv	845	442
		validate.csv	167	59
	random	train.csv	856	430
		validate.csv	156	71
clintox	cluster	train.csv	105	1152
		validate.csv	7	214
	random	train.csv	104	1152
		validate.csv	8	214
deepchem_Lipophilicity	cluster	train.csv	3448	122
		validate.csv	607	23
	random	train.csv	3447	123
		validate.csv	608	22
sol_del	cluster	train.csv	146	813
		validate.csv	39	130
	random	train.csv	153	805
		validate.csv	32	138
tox21	cluster	train.csv	623	4944
		validate.csv	145	837
	random	train.csv	649	4917
		validate.csv	119	864

Figure 5. The final class balance metrics for the various data splits.

In comparing the various clusters we also wanted to evaluate how well the clustering algorithm was doing at identifying groups of active molecules. In Figure 6, we can observe the red line indicating where the overall dataset fraction of actives is and then the relative fraction for each cluster. The presence of over 38 of 50 total clusters above or below this line indicates that it is doing a good job in grouping like-compounds together and therefore ensures that chemical scaffolds are not randomly split between both training and validation sets. (Should maybe look at a similarity metric within cluster vs between?)

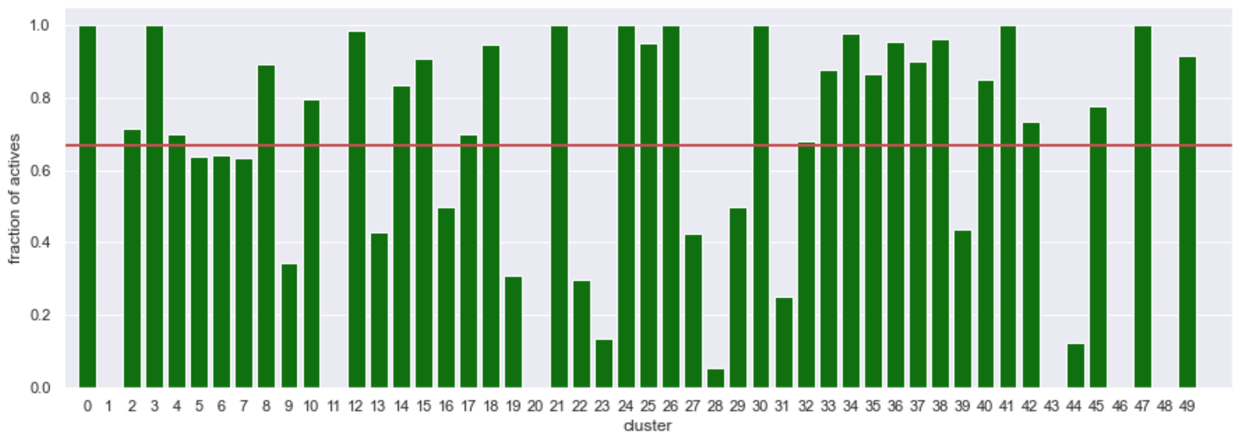


Figure 6. The fraction of active molecules in each cluster, with the red-line indicating the overall class ratio in the dataset.

The other evaluation we can perform on initial datasets is to look at how the molecular fingerprints might cluster with respect to the relevant activity data. In order to assess this in a generic way, we generated the fingerprints and then employed the Uniform Manifold Approximation (Umap) to reduce the bit vectors to two dimensions for visualization. We decided to utilize Umap over PCA since it does a better job at preserving local structure and is more interpretable than TSNE. (https://umap-learn.readthedocs.io/en/latest/) From the resulting visualization in Figure 7, we can get a rough gauge of how difficult the various classification tasks might be. For instance, we can observe for the BACE dataset that many of the local clusters are separated based on activity, while for the HIV dataset, the active compounds appear to be randomly spread in the chemical space. Furthermore, BACE, TOX21, Clintox and Solubility all appear to have clear clusters and structure within the data, whereas HIV and Lipophilicity both appear to have mainly a single cluster. Therefore, we make the broad generalization that the HIV and Lipophilicity datasets are more difficult than the others.

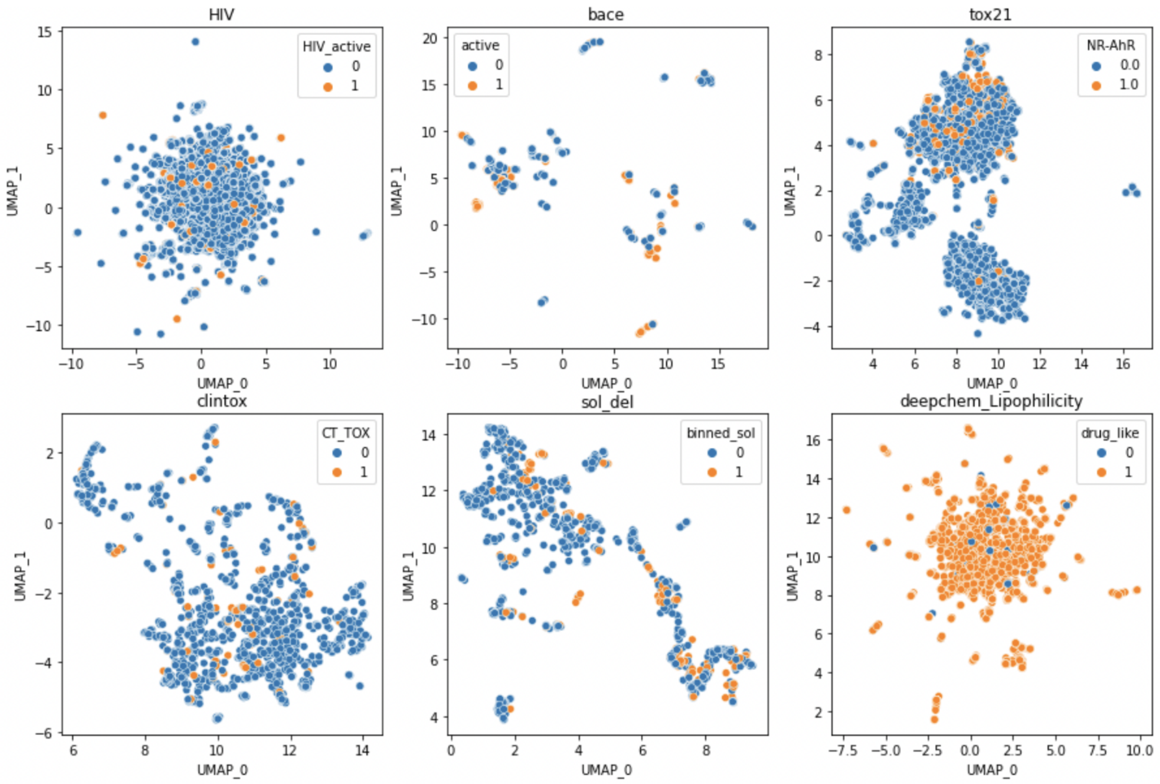


Figure 7. The Umap dimensionality reduction visualizations for each of the datasets.

Modeling:

To explore the question of how much value does model complexity add in the context of early drug-discovery programs, we needed to select the models to compare. There is not necessarily a concrete definition of what separates simple and complex models, and instead they operate on more of a gradient. We applied general heuristics in that if the model is too complex it might overfit the data, and fail to generalize. While, if a model is too simple, it might underfit the data and fail to give optimal performance. Interpretability is also something which can often favor simpler models, however there are also techniques for doing this in more complex modeling space. Lastly, the computational costs should also be considered, with simpler models typically being faster to train and easier to optimize hyperparameters for than more complex ones.(https://towardsdatascience.com/simplicity-vs-complexity-in-machine-learning-finding-the-right-balance-c9000d1726fb) For this project we chose to evaluate several of the readily available Scikit-learn models against the MIT Chemprop model.

Chemprop: In order to evaluate the complex modeling space, we opted to explore a publicly available package instead of trying to develop a novel framework on our own. Due to the time of the project, it did not seem feasible to build something which would function better than the work resulting from collaborations spanning several years between academia and industry. While there are many packages available such as Few-Shot Learning for Molecules (FS-MOl) from a Microsoft Collaboration, to DeepChem a scientific learning package from Stanford (DeepChem), we opted to go with the MIT package of Chemprop. For this project we felt that Chemprop provided a good framework that balanced both ease of use, implementation as well as having in-built features such as data splitting and early stopping which would help ensure optimal performance. Furthermore, evaluations of Chemprop against other modeling methods are present in the literature, which help give further comparison and insight into our results. (https://pubs.acs.org/doi/full/10.1021/acs.jcim.9b00237)

The Chemprop package offers many different approaches to training the models, however the core is based around a directed message passing neural network (D-MPNN). This network works by passing a directed message across a molecular "network" made up of bonds as edges and atoms as nodes. The resulting representation is then utilized to create a learned representation of the molecule which the model then uses to train against the property prediction task. In addition, the Chemprop package also allows for a fixed feature representation, such as a Morgan Fingerprint, or generic physicochemical properties and descriptors to be utilized in conjunction. For this project we utilized a Morgan Fingerprint representation in addition to the D-MPNN for the model to enable a better comparison to the simpler models.

Simple Models: There are a vast array of different models that can be applied to classification problems, and many packages that offer versions of them. For this project, we chose to explore the Scikit-learn implementations since they are generally accepted in the machine learning industry and also offer many helpful functions such as cross validation and data splitting. Furthermore, the Scikit-learn package has implementations of many common model types which helped provide a range of models for us to compare.(https://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html) For this project we decided to utilize Logistic regression and K-nearest neighbors (KNN) as some of the most simple models. We also trained a Dummy Classifier that would only predict the most frequent class in order to provide a true baseline. We then looked at both Random Forest and Gradient Boosted Decision Trees as different decision tree type classifiers, that are utilized often within the chemistry community and are less sensitive to data normalization, while still remaining interpretable. Lastly, we also fit a Support Vector Machine model (SVM) as these were quite popular in many chemistry applications during the early 2000's, before Neural network surpassed them in popularity. (https://pubs.rsc.org/en/content/articlehtml/2011/cp/c1cp00051a?casa_token=GUiEmnj2PZ4AAAAA:o8NX-EqVpxmT-8jxdJFCO6QUwf7Hnj-VGwVF8QRpBWqdpWkwXLUXxN0DgnJvXQ4kutk0MevvA_xTHw) (https://www.sciencedirect.com/science/article/abs/pii/S0169743908001998)

For the simple models, we featured all of the molecules using the Morgan Fingerprint generation algorithm available in the RDkit package. While we explored different radii and bit numbers initially we found that a radius of 2 and a bit length of 1024 produced reasonable results without generating excessively sparse vectors. While it is possible that these parameters could be further tuned in future work, we were unable to thoroughly evaluate the fingerprint parameters across all of the datasets due to computational time.

Modeling Methods:

While the simple models could be trained in a reasonable amount of time on our local computers, the Chemprop model did take a significant amount of computational time. We therefore opted to utilize a GPU to help speed this up. From an initial test on the BACE dataset, which was one of our smaller ones, training the chemprop model locally on a MacBook Air took about 1.5 hours, while on the University of Michigan Great Lakes Cluster it took less than 5 minutes. When factoring in the cross validation and multiple training attempts, the Chemprop training would have been prohibitive to do locally but was relatively painless on the cluster. While many tools now exist to access GPU resources, this highlights one key disadvantage of the more complex models that could limit their accessibility.

In order to get a sense of how well each model was generalizing, while being able to make the most efficient use of the smaller datasets, we opted to use cross-validation. Cross-validation is where the dataset is divided into several equal parts and one part is held out as a test set during a round of training. Training and this testing is then repeated with each split until all have been used once. In effect, the difference between each test set scores, can give an empirical measure of how well the model is generalizing to new data. For this project, we opted to perform 5-fold cross validation to keep the training times reasonable. This was performed either using the Scikit-learn function, or as a command line feature in the Chemprop package. After performing cross-validation, we then trained the various models on the entire training set and made predictions on both the training and validation sets.

Evaluation and Results:

Some potential ideas to explore:

Introduction to metrics for evaluation - data set differences, class balance, positive class prevalence, dataset size

Cross validation results - training - generalizability

Scoring on validation data

Overfitting - by comparison to validation data

In depth dive for Bace data set

SMOTE technique for dealing with imbalanced datasets??

Modeling Result Overview:

A major consideration for an experiment of this complexity, with multiple datasets and multiple models, is which scoring metric is most appropriate for model optimization on a given dataset. Scikit-learn offers over twenty different classification metrics to choose from. With this project being simplified as a binary classification problem, we decided to look at 5 of the more common metrics, accuracy, f1 score, ROC-AUC, log loss, and Matthews correlation coefficient. Each of these have their own pros and cons. Accuracy score is the most well known metric and is calculated as Equation here

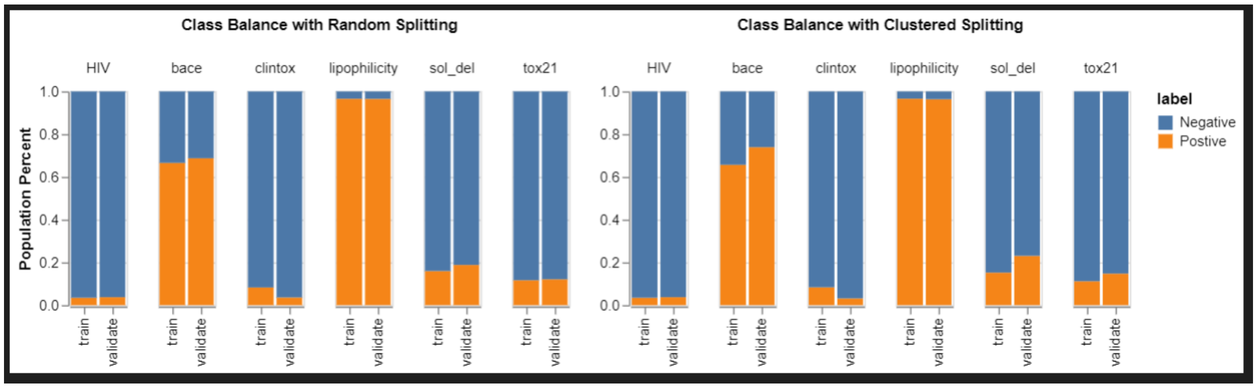


Figure 8. A comparison of class balance in train/validate sets based on random splitting or KNN cluster based splitting. Interactive vis exists - also fix dark-mode background

Include summary of metrics used -

Accuracy Score: Only considers the number of correct predictions, and does not consider how the model fails (probability, false negative vs false positive, etc.., has challenges with class imbalance. This inflates model results, especially in cases where there is class imbalance.

F1 Score: useful when FP and FN are equally costly. The F1 score computes the harmonic mean of precision and recall scores. It returns a high value when both precision and recall have high scores, a low value when both are low, and a medium value when one is high and the other is low. F1 can be especially useful when comparing different classification models, where we just want the best overall model parameters before our inevitable trade-off choice of precision vs recall, in which case we might use F-beta score, a parameter that takes into account both precision and recall, while still allowing the ability to bias results in one direction.. F1 score focuses on positive predictions, but does not focus on the negative prediction metrics.

https://towardsdatascience.com/the-f1-score-bec2bbc38aa6

ROC-AUC: Receiver Operator Characteristic, plots prediction *scores against True Pos vs False Pos determines signal from noise - measures the ability of model to distinguish between classes. Very similar to balanced accuracy results where each class gets equal priority. This helps understand model results better when there is class imbalance. - still not really sure when to use F1-score vs ROC-AUC and what comparing these two tells us. One uses scores and one of uses predicted class counts -> below article suggests that for F1 score to be better, we need to choose a threshold -> I see that ROC-AUC might not be good for imbalanced datasets.

https://neptune.ai/blog/f1-score-accuracy-roc-auc-pr-auc

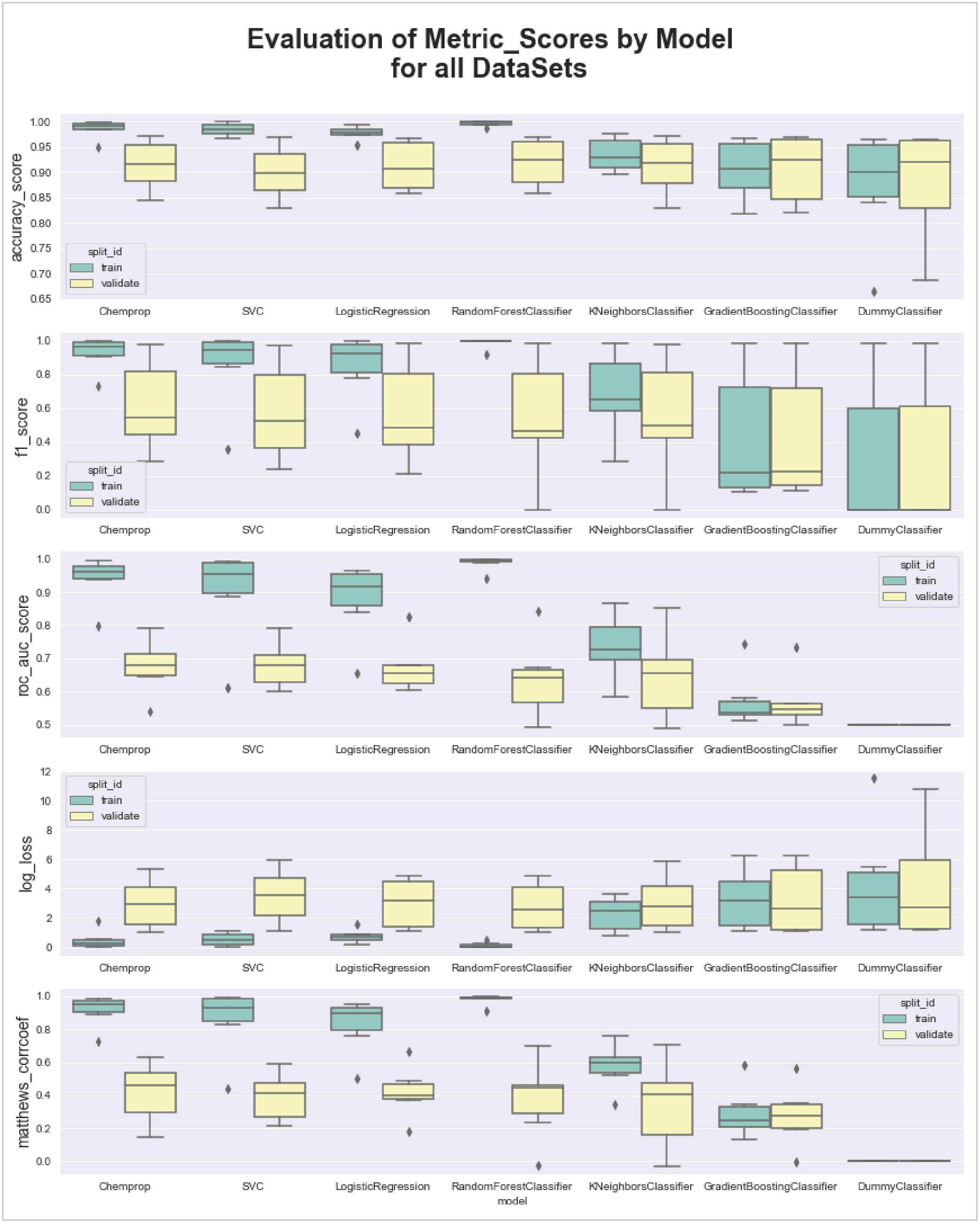
Matthews Correlation Coefficient: This metric is considered a more comprehensive accuracy measurement for imbalanced datasets than even F1 score because it takes into account both positive and negative predictions as well as the subset prediction accuracy of true vs false labels within each prediction class. Matthews correlation coefficient is also agnostic to class swapping, meaning that it doesn't matter which class is labeled as 1 or 0. (This does in fact seem to be the strictest scorer (when considering metrics out where 1 is the top score. What is the range of possible scores for this class?)

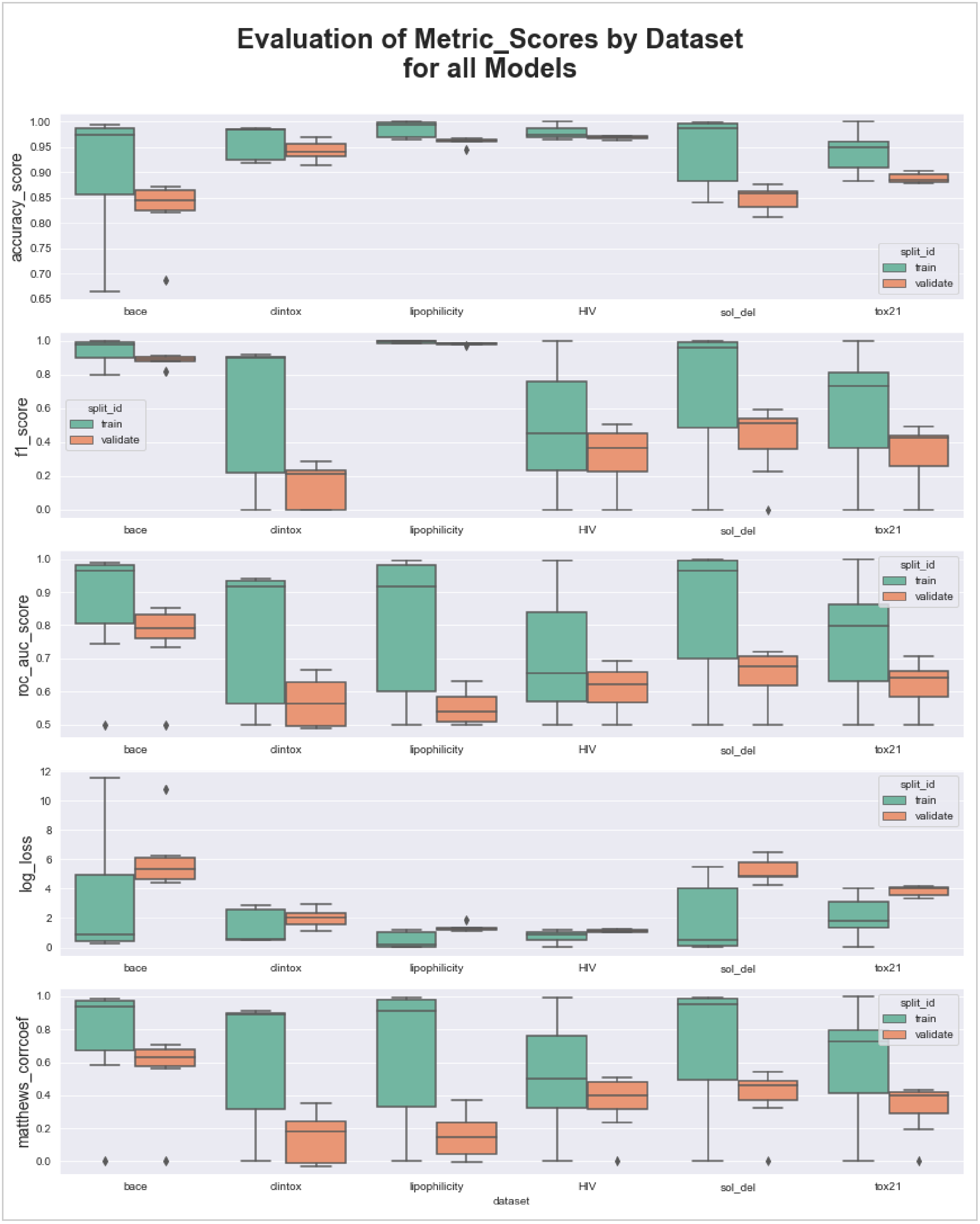
https://thedatascientist.com/metrics-matthews-correlation-coefficient/

Log-Loss:

Things to look at:

- Classes prone to overfitting based on a comparison of train and validation data
- Validation scores only and which metrics are most useful
- Cluster vs random splitting - where cluster is used to be more representative of a real world data situation
- Cross validation results??


Evaluation of Metric_Scores by Model
for all DataSets
accuracy_score
f1_score
roc_auc_score
log_loss
matthews_corrcoef
split_id
train
validate
Chemprop
SVC
LogisticRegression
RandomForestClassifier
KNeighborsClassifier
GradientBoostingClassifier
DummyClassifier
model



BACE Results Deep Dive:

In order to explore the differences between modeling techniques further, we chose to look closely at the BACE dataset due to its relatively small size and the prediction of on-target potency is often the most critical task for many drug-discovery programs. In order to maximize performance, the hyperparameter optimization was performed using the Chemprop algorithm that involves a Baysian optimization approach to search over the various neural network parameter space. The hyperparameter optimization suggested that increasing the drop-out from 0 to 0.3 helped performance, suggesting that a reduction in the un-optimized model might have been overfitting. Interestingly, the other parameters of hidden-size and network depth both increased from 300 to 900 and 3 to 6 layers respectively. This suggests that adding additional model complexity could help with the performance. It should be noted that the hyperparameter optimization was performed to improve the roc-auc metric, so further exploration might yield other parameters that improve other metrics.

Following the updated training, if we explore the predictions for each of the models relative to the actual labels, shown in Figure 9, it is clear that many of the models are not performing particularly well. The Dummy Most Frequent classifier appears in dark green for most samples because it simply assigns with complete confidence the positive label for this dataset. Furthermore, the simpler models of Logistic Regression, Random Forest and Gradient Boosted Trees all show a lighter color, indicating that the models are not particularly confident in their decisions. The KNN classifier is clearly not a good choice in this situation as it has large sets that are not correctly predicted. Interestingly, the complex Chemprop models appear to do a better overall job based on the heatmap, however there are some clear groupings of molecules where all of the models appear to perform poorly.

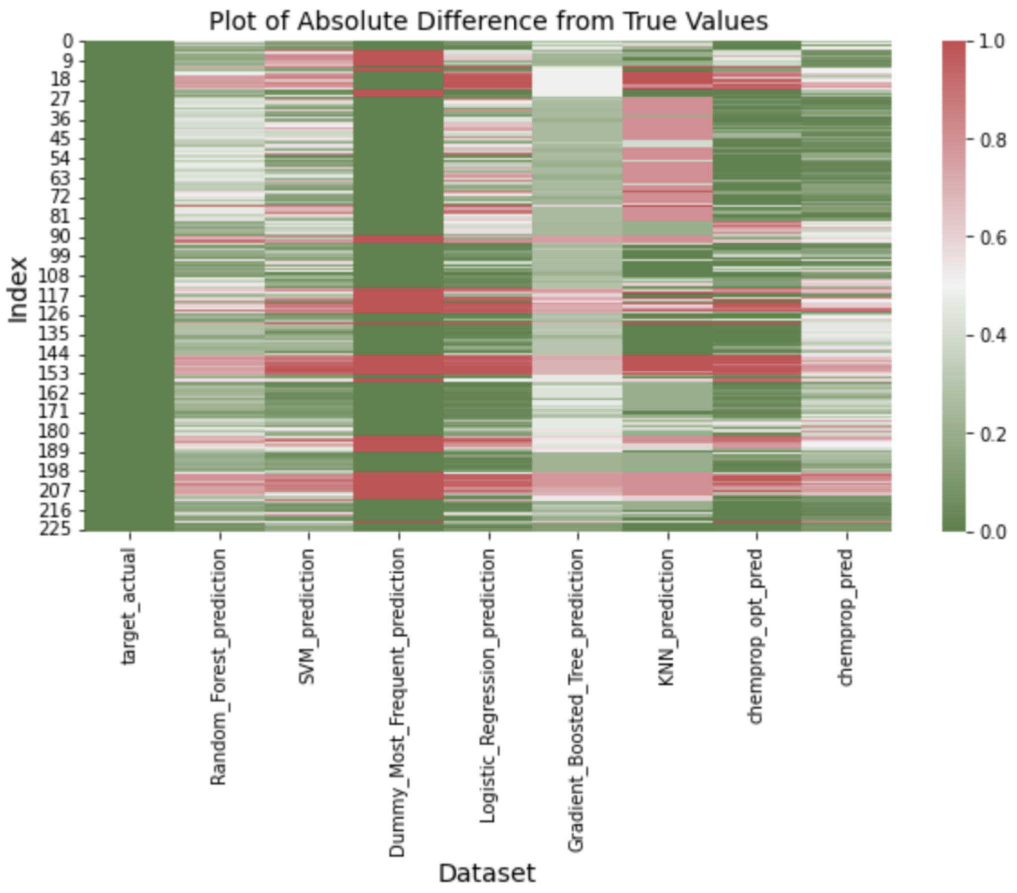


Figure 8. A heat map showing the absolute difference between the predicted probabilities and the actual target labels. Darker green colors indicate less difference from the true value, while red colors indicate a stronger in-correct prediction.

The other aspect of optimization that we then explored for the various different models, was how the decision threshold affected the true positive and false positive rate. To evaluate this across the different models, we plotted the ROC curve (Figure 9) for the different models. This curve can help determine how sensitive the prediction accuracy is to changing the threshold at which a prediction is considered to be positive or negative. The area under each curve also gives a measure of model performance, with larger area indicating a higher performing classifier. From this visualization, it can be seen that both Chemprop models are relatively high performing, and from the simple models, the Gradient Boosted Tree is the best. Therefore, we selected the Optimized Chemprop model and the Gradient Boosted Tree as our models for further evaluation. This curve also helped us select a descrimination threshold for further scoring. To accomplish this we calculated the J-statistic for each threshold value and selected the maximum.
(https://machinelearningmastery.com/threshold-moving-for-imbalanced-classification/) However, these optimal thresholds were quite high for some of the models, such as the optimized chemprop which made it difficult to compare the models. Therefore, an intermediate descrimination threshold of 0.6 was selected, which seemed to strike a reasonable balance of

true positives, which are of more interest in the drug discovery field to avoid missing potentially useful compounds.

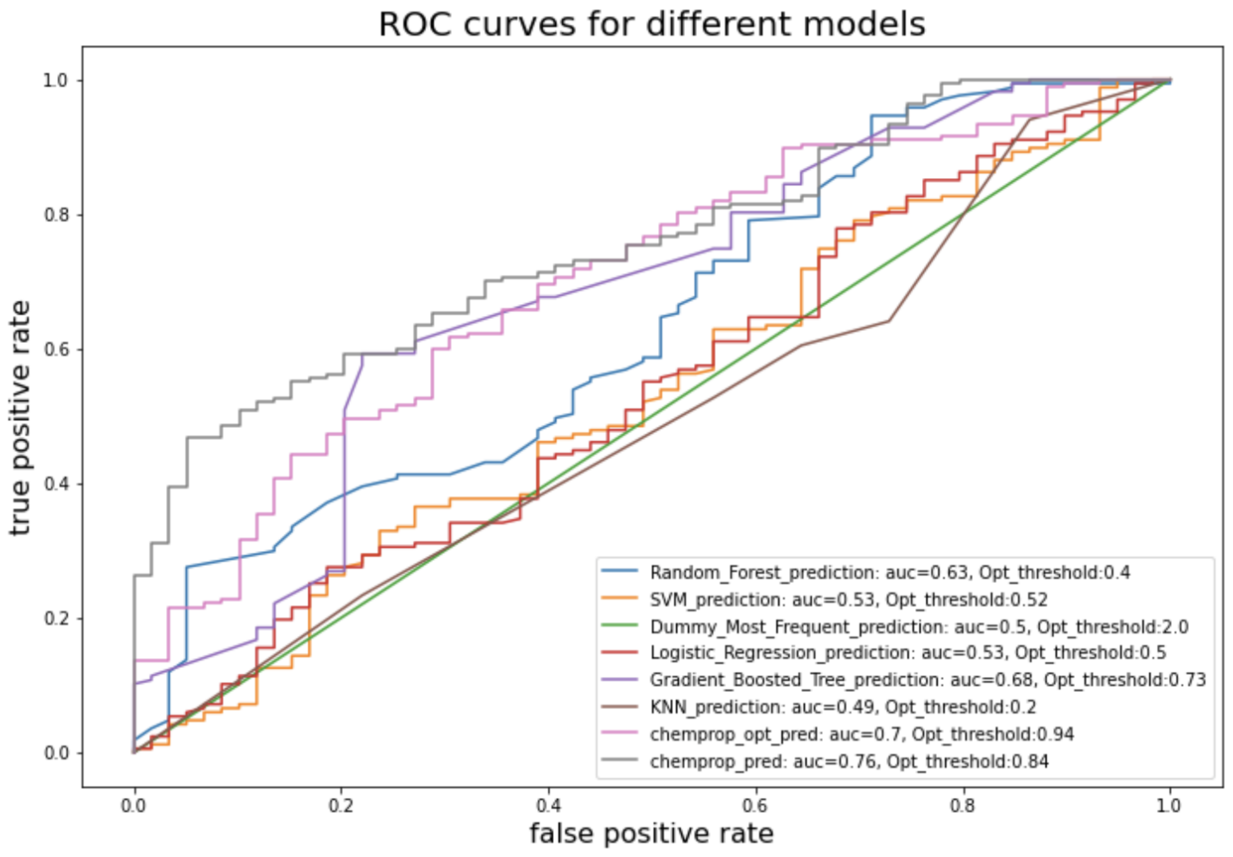
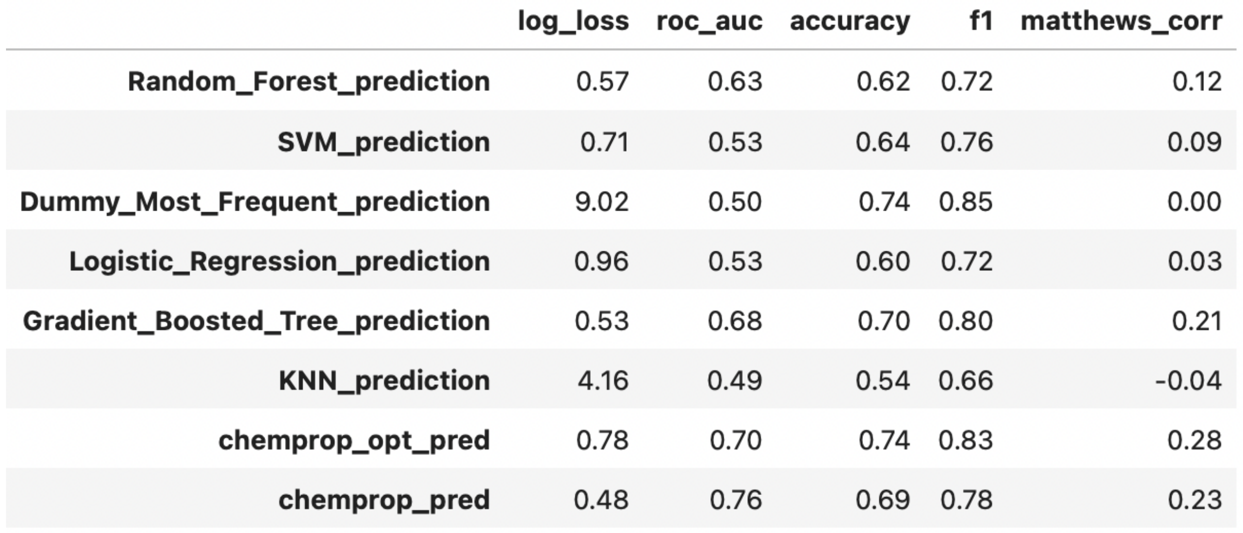


Figure 9. The Receiver Operating Characteristic curve for the models on the BACE dataset.

Once the descrimination threshold was determined, the series of scoring metrics were calculated in order to explore the model performance. From Figure 10. We can see that overall the scores are not particularly impressive. In part, since there is some class imbalance, the dummy prediction is relatively high scoring in terms of accuracy and f1. However, we can see that the Matthews correlation coefficient is quite poor for the Dummy model. The optimized Chemprop model shows similar performance to the Dummy classifier, along with a higher Matthews correlation coefficient and drastically lower Log Loss, meaning that the model is performing better than baseline. We can also observe from the scores that the hyper parameter optimization for the Chemprop model helped in some aspects such as accuracy and f1 score, but hurt in others, such as Log Loss or roc-auc. From the results it is not clear how much value the hyperparameter optimization added.

The score matrix also allows us to compare some performance metrics between the simple models. Overall, the simple models did not appear to perform drastically better than the baseline Dummy model. However, the Gradient Boosted Decision Tree, did show some of the best overall scores as well as a Matthews Correlation coefficient that was on par to the complex Chemprop models.



	log_loss	roc_auc	accuracy	f1	matthews_corr
Random_Forest_prediction	0.57	0.63	0.62	0.72	0.12
SVM_prediction	0.71	0.53	0.64	0.76	0.09
Dummy_Most_Frequent_prediction	9.02	0.50	0.74	0.85	0.00
Logistic_Regression_prediction	0.96	0.53	0.60	0.72	0.03
Gradient_Boosted_Tree_prediction	0.53	0.68	0.70	0.80	0.21
KNN_prediction	4.16	0.49	0.54	0.66	-0.04
chemprop_opt_pred	0.78	0.70	0.74	0.83	0.28
chemprop_pred	0.48	0.76	0.69	0.78	0.23

Figure 10. Table showing the different BACE models and their respective scores.

Since many of the models were not particularly high-performing we wanted to explore the interpretation of these models to see if we could understand what could be happening as well as try to understand if there are any fundamental differences between the Gradient Boosted Tree and the optimized Chemprop model. In examining some of the molecules that were predicted correctly or incorrectly, the example shown in Figure 11 is informative. The top molecule is inactive in the data, while the bottom two both belong to the positive class. Interestingly the Gradient Boosted Tree model predicts all three to be active, while the Chemprop-optimized model predicts them all to be inactive. Given the only difference in the three structures is the location of the oxygen atom and the size of the ring on the left side, it is perhaps understandable that these would have very similar fingerprint based representations and therefore would be predicted similarly. This example highlights one of the key challenges in using 2D representations, where in 3D, the size of the ring might completely prohibit the molecule from binding the protein correctly.

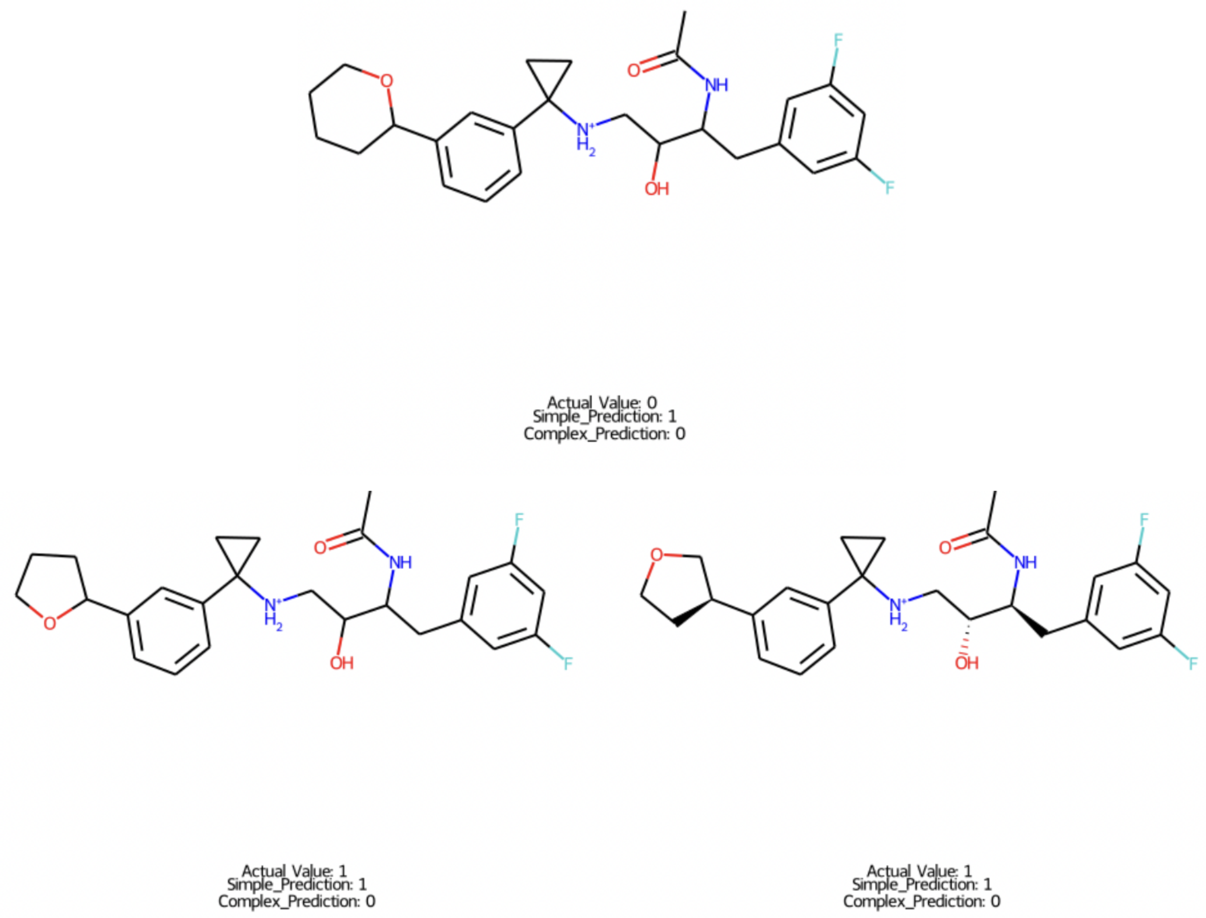
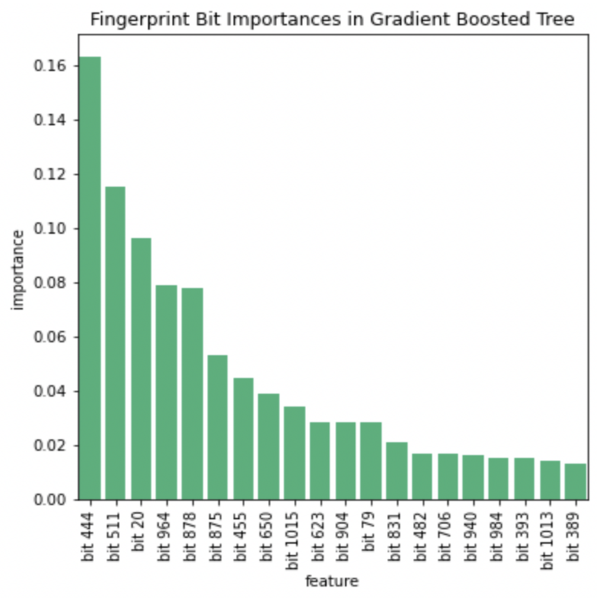


Figure 11. Example molecules from BACE dataset showing some of the subtlety that is often present in chemical structure to biological activity relationships.

Lastly, we wanted to try and understand to what extent the model interpretability might compare between the simple and complex models. In the chemical fingerprint based modeling space, interpretation is a very open and active question, with many papers only coming out in the last few years.(https://jcheminf.biomedcentral.com/articles/10.1186/s13321-021-00519-x) (https://link.springer.com/chapter/10.1007/978-3-319-56850-8_1) However, many of these approaches involve more elaborate featurization and descriptors and typically utilize genetic algorithms or Monte Carlo simulations. For our example, we wanted to see how we could relate this to the circular fingerprints.

In order to explore the simple model interpretation. We extracted the feature importances and then ranked the top 20 values. These were then associated with certain positions in the fingerprint array using RDkit. (http://rdkit.blogspot.com/2018/10/using-new-fingerprint-bit-rendering-code.html) It should be noted that there is potential for some bit collisions, so the substructures might not be exactly the same for every molecule, but many of them resulting here were relatively simple and common in the space, which makes these seem less likely. From Figure 12, we can see the top bits and then the associated substructures of these. Interestingly, some of these are quite simple, such as bit number 650, 623 and 904 which are simply a single oxygen, nitrogen or fluorine atom respectively. The highest importance bit was 444, which is simply a basic amine. Other bits are various parts of common substructures or ring systems. This result likely helps explain some of the above observations that the model is more learning general scaffolds, and without additional data might have a hard time distinguishing between subtle atomic features from a fingerprint alone. Bit 511 was present in this dataset, and was perhaps one of the more specific bits in that it requires a specific 5-membered ring to contain a nitrogen atom. The results from this exploration seem to suggest a higher importance of the features holistically over an individual few bits. The small nature of these might also suggest that exploring larger radius fingerprints could also contribute to a less localized interpretation of the chemical structure.



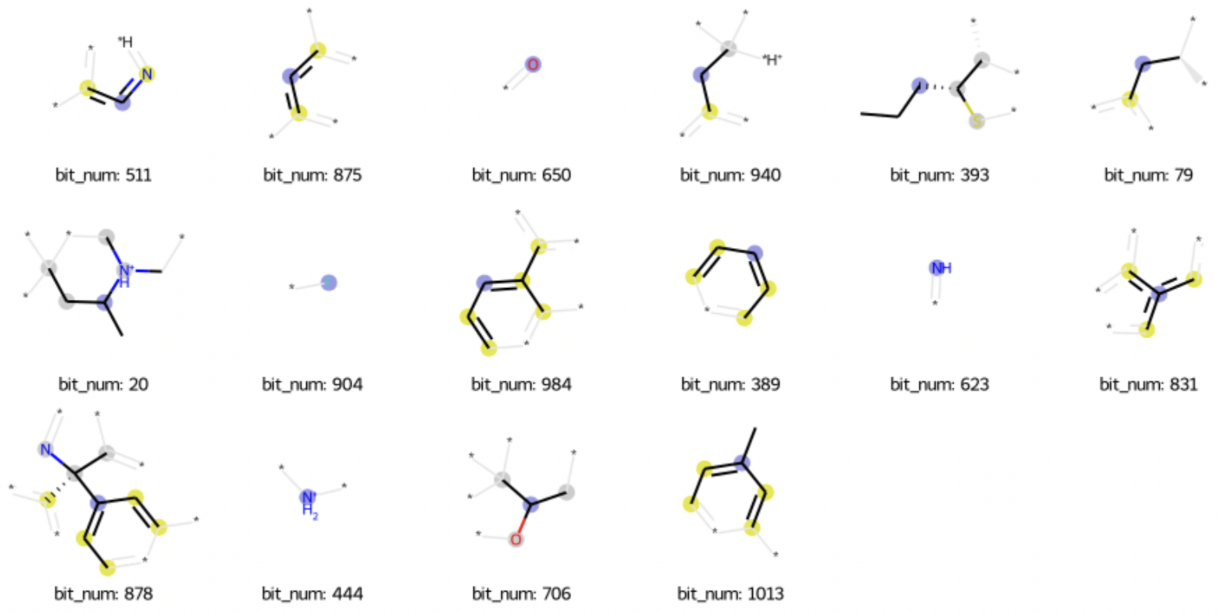


Figure 12. The feature importances and the associated fingerprint bits for the top gradient boosted tree predictions.

To explore how interpretability would work in a complex model space, we utilized the Chemprop interpret function. This function utilizes a Monte Carlo simulation over the different models and substructure types to generate a minimum substructure leading to a positive result as well as a measure of how much that substructure contributes to the prediction. In practice, this function was of limited usefulness in our hands as the simulation took almost 30 minutes to run on each compound and as shown in Figure 13, the substructure that was contributing significant activity to the prediction, was also present in several of the other molecules that were predicted to be inactive. It is possible that this is a result of the model not being particularly performant, or that there were other more negatively correlated features on these molecules as well. On the whole from our work here, we did not find the fingerprint based interpretability of either the simple or complex models to add significant insight to the problem of BACE activity prediction.

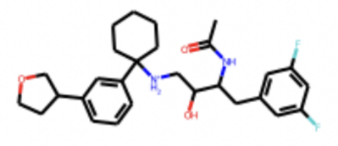
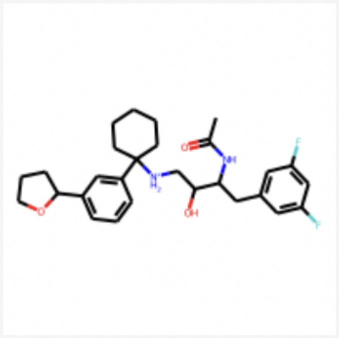
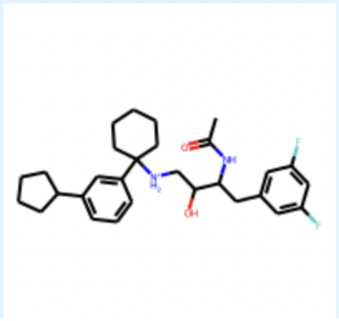
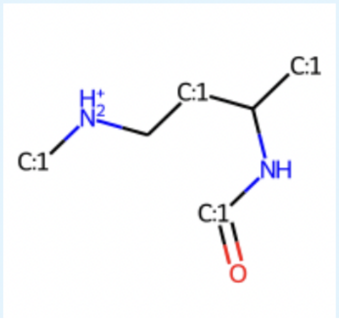
	Structure	prediction	substructure	substructure_activity
0		0.104	NaN	NaN
1		0.193	NaN	NaN
6		0.633		0.961

Figure 13. The results of a Chemprop interpretation of some of the molecules.

Conclusions and Future Directions:

From our results…
Discussion:
Answer to question - is the complex model worth it?

While these results are interesting, we recognize that there are many future directions that would be relevant to explore. From this work, we were only able to perform minimal hyperparameter tuning, and only on a single dataset. It would be interesting to see if the results from the BACE dataset, where Chemprop performance did not improve significantly with the tuning, holds true for other examples and class balances. While some parameters were explored for the simple models, these could also benefit from additional tuning of the parameters as well. In particular, the depth and minimum sample parameters in the tree-based methods can often modulate overfitting and might help improve performance. The support vector classifier could also benefit from tuning the kernel function and the regularization parameter since its performance was lower than expected overall.

Another opportunity for future work would be to expand the number of complex models. While the setup and configuration of these can take time, it would be interesting to understand how these more complex models relate to one another. For instance, other NN modeling approaches such as Deepchem (https://deepchem.io) or Few-Shot Mol (https://openreview.net/forum?id=701FtuyLlAd) would be relevant to compare. These could also be compared to 3-dimensional docking (https://www.schrodinger.com/platform/smdd/drug-discovery-workflows#hit-identification) or quantum mechanical based Free Energy Perturbation (https://www.cresset-group.com/about/news/fep-drug-discovery-toolbox/) for target related datasets to compare. The software for the 3D methods are often proprietary and can be costly, so working out access would be important here.

Lastly, the featurization of the molecules is of the utmost importance to model performance. There are some opportunities to leverage calculated molecular properties, such as through the descriptastorus package (https://github.com/bp-kelley/descriptastorus), which might help improve generalizability. Moreover, other fingerprinting techniques and count based fingerprinting (https://www.rdkit.org/UGM/2012/Landrum_RDKit_UGM.Fingerprints.Final.pptx.pdf) could also be informative to compare. While Chemprop works towards a deep-learning approach to the chemical representation with the graph-based approach, it could be informative to consider undirected graph based approaches as well. While difficult computationally, quantum mechanical descriptions based on molecular dynamics are also becoming increasingly popular.

While there are many directions to explore further, it could be of interest to consider various ensemble methods where one does not need to look for a “best” model but instead can capture the benefits of several modeling approaches at once. This is further supported by the iterative and uncertain nature of drug discovery where many experiments are still likely to be run, so even if models can enrich the target molecules in ones that are higher value it will help discover new drugs faster and with reduced cost.