Improving Drug Discovery by Utilizing Machine Learning Algorithms and Biological Activity Data of Target Proteins

Mathematics & Computer Science

Lagnajeet Panigrahi

Shrewsbury High School

Instructor: Catherine Phillips

75 Cypress Ave, Shrewsbury, MA 01545

Table of Contents

Acknowledgement of help Received

This project was an independent project conducted at home and school. No outside assistance from professors or mentors was used during the development of this project. The only guidance received was from Shrewsbury High School Chemistry Teacher, Catherine Phillips during the creation of the research paper and abstract about structure and what information needed to be included.

Abstract

Humans have been battling the age-old battle against illness for centuries. Only recently have humans made significant progress in averting illness through the creation of drugs and the field of drug discovery. To create a drug and deem it usable requires a rigorous process that examines one attribute in particular: safety. Unfortunately, ninety percent of drug development fails despite the many successful strategies used. Current chemical methodologies rely on a hit-and-miss approach where large amounts of drugs are analyzed for their properties by hand. This process is expensive, time-consuming, and often inaccurate. This project aims to address this through the implementation of various machine-learning algorithms that can predict the IC50 and pIC50 values of possible drug candidates. These values are essential in determining the quantity of a drug needed to inhibit a biological process by half but also are important in determining the toxicity of a drug and how it impacts patients. Using regression-based machine learning models and bioactivity data of compounds and target proteins, these values can be predicted and outputted. Upon completion of this project, we developed multiple regression models for the target protein of the SARS coronavirus. After statistical analysis, the best model was chosen: DecisionTreeRegressor. This model had a root mean squared error score of 0.34 and an R-squared score of 0.9019. This implies that this model fits the situation and makes accurate predictions. We conclude that drug discovery can become quicker, more accurate, and cost-effective through the implementation of machine learning algorithms.

Keywords: Drug discovery, IC50, pIC50, machine learning, regression, bioactivity, r-squared score, root mean squared error score

Introduction/Research

In this expanding world where illness and disease are increasingly common, efficient, accurate, and cost-effective approaches to drug discovery are becoming more necessary. Inaccuracies and other sources of error can greatly delay drug development which can have disastrous impacts on individuals around the world. It was calculated that there were 29 life-years lost in North America alone per one hour of delay in drug approval (Helwick, 2015) and 90 percent of clinical drug development fails (Sun et al., 2022). A major step in drug development is determining the toxicity of the drug, its potency, and overall, its safety. Current methods require careful experimentation and individual assessment of large amounts of potential drug candidates by hand (Blanco-González et al., 2023). Although current methods are good, they are prone to errors, are costly, and are time-consuming (Blanco-González et al., 2023). Consequently, together, the high likelihood of failure combined with the inherent problems with current methods create a system that fails to perform effectively and efficiently. This is detrimental for both developers— as they lose time and money— and patients who do not receive proper treatment at the correct time. To overcome this, a new and innovative system is required: one that can provide information about the potency and toxicity of a certain molecule or drug and how it absorbs into the body in a fast, cost-effective, and accurate manner. The proposed method that this project will employ to solve this problem is the use of regression-based machine learning algorithms in Python with preprocessed data to give information on drug toxicity and potency by predicting IC50 and pIC50 values. To build the model, any target protein can be utilized (the only difference between models of different target proteins is what they are trained and tested on). For this project, the SARS coronavirus 3C-like proteinase target will be used. The model's success will be determined by statistical analysis.

This type of project will greatly impact the scientific community as it will not only provide a model to predict IC50 and pIC50 values but pave the way for the use of machine learning and related technologies for drug discovery. This project will build on past research on this problem that utilized different types of models and architectures.

Modern-day drug discovery is a process in which new drugs and therapeutic entities are created based on computational, experimental, translational, and clinical models (Zhou et al, 2017). The drug creation process can be broken up into six main steps: target identification, target validation, “hit” identification, lead discovery, “hit-to-lead” (H2L) stage, and lead optimization (Alvarellos, 2023). Target identification is the stage where the target protein for the disease or illness that the drug will target is chosen (Alvarellos, 2023). During target validation, the selected biological molecule is analyzed to confirm whether or not it plays a role in disease development and whether it needs to be targeted (Alvarellos, 2023). During “hit” identification, molecules that are active, inactive, or intermediate at the target biological molecule (Alvarellos, 2023). In other words, at this stage, it is determined which molecules have impacts on the biological molecule of the disease. In lead discovery, each molecule is screened for its properties pertaining to safety, efficacy, and potency (Alvarellos, 2023). In the H2L stage, scientists optimize the characteristics of the molecule they want based on results from the lead discovery stage and test to ensure they have reached their goals (Alvarellos, 2023). Finally, during the lead optimization stage, the most promising candidates get further optimized (Alvarellos, 2023).
The problem with these conventional methods is that they are very difficult, lengthy, costly, and inaccurate (Zhou et al, 2017). This problem is exemplified when considering that each clinical trial can cost anywhere from 30 million dollars to 310 million dollars and fewer than ten percent of these trials succeed (Alvarellos, 2023). Imperfect representations by humans, errors in

experiments to determine the properties of drugs, and false positives all contribute to the failure of many clinical trials (Alvarellos, 2023). Machine learning and data-driven drug discovery have shown promise and could be a possible solution due to their characteristics: unbiased data, data standardization through preprocessing, and overall because of consistency and high performance if done right (Alvarellos, 2023). In this project, for the most part, the steps of H2L and lead discovery will be addressed and improved through machine learning as these steps are the most critical and most prone to error. A newer, more effective method called drug design that employs machine learning is a possible solution to the discussed problem and is the avenue this project will be taking. Drug design employs machine learning algorithms, computational methods, and bioinformatic methods to analyze drugs and molecules based on the knowledge of a biological target (Zhou et al, 2017). During the lead discovery and H2L stages, many factors are taken into consideration, but in this project, the factors of potency and drug toxicity will be taken into account. These metrics directly provide information about how well a patient will respond to the drug in question and what the drug's effect will be (Zhou et al, 2017). This project aims to be able to take a molecule, analyze it, and output values that indicate potency and toxicity effectively, accurately, and on time.

Drug toxicity and potency can be described by IC50 values and pIC50 values. IC50 stands for inhibitory concentration at fifty percent (Visikol, 2022). pIC50 values are the same values in concept, except they have had a negative logarithmic function applied (Navre, 2019). This is because the IC50 values follow a logarithmic phenomenon and pattern like pH and pOH values (Navre, 2019). In addition, once converted to pIC50, the data becomes easier to read and interpret. These values can be indicative of how much of a drug is needed to create an impact, what the drug's potency is, and its systemic toxicity (Visikol, 2022). For example, a lower IC50

value can indicate higher drug potency as less of the drug is required to inhibit a biological process and is less toxic because less of it is needed (Berrouet et al., 2022). There are many methods of calculating IC50 values such as using raw data from various assays and using curve fitting software and point-by-point analysis (isirv Antiviral group, n.d.). For the purposes of this project, we will be using a different methodology. IC50 and pIC50 values can be calculated through the SMILES notation of a molecule or drug and other bioactivity data (whether the drug or molecule is active or inactive at the target biological organism) (Liu et al., 2019). SMILES is the Simplified Molecular Input Line Entry System that converts chemical formulas into a notation composed of strings that are easily readable by a computer (US EPA, 2012). The data used in this project will be sourced from ChEMBL. ChEMBL is a curated database of molecules with drug-like properties (ChEMBL, n.d.). Since this machine learning project will be tested with SARS coronavirus data, the bioactivity data will contain bioactivity data and SMILES notation of certain molecules that have been tested with the target protein of SARS coronavirus. This will be the data that will be processed and will be used by the machine learning algorithm during training and testing.

Machine learning will be leveraged in this project with the data aforementioned. Machine learning is a branch of artificial intelligence and computer science that utilizes data to create algorithms that learn, predict, and improve (IBM, 2023). Within the field of machine learning, there is supervised learning and unsupervised learning. The difference between these types of learning for a machine learning model is that supervised learning uses labeled data outputs and inputs while unsupervised learning does not (Seldon, 2021). This project will employ supervised learning as the data is labeled by SMILES notation and bioactivity and use regression-based machine learning models. Regression-based machine learning models take in data and use

mathematical functions to establish relationships between certain inputs and outputs to eventually create a classifier and be able to predict outcomes (Kumara, 2023). This project will test many different machine learning regression models using LazyRegressor and then determine the best one to model drug design for the scenario presented within the project. LazyRegressor is a library within Python that can automate machine learning training, testing, and analysis of models (Patil, 2021).

This project is fully dependent on the use of the programming language Python and the data sourced from ChEMBL. As a result, this project employs the use of many libraries and applications. This project will be programmed on Google Colab rather than conventional IDEs such as Pycharm or Visual Studio Code. Google Colab is an online cloud-based Jupyter Notebook environment (Sharma, 2024). There are many benefits to using this as opposed to other IDEs. The benefit of utilizing Google Colab is that it is cloud-based, so all of the machine learning training, testing, and simple running of code is run on GPUs and TPUs hosted by Google (Sharma, 2020). This allows for the model to be trained quickly and efficiently while also not being hindered by the performance of a personal device. Next, this project requires the use of many Python libraries. The most prominent libraries used in this project were Pandas, Numpy, RDkit, Matplotlib, PaDEL, Seaborn, and LazyRegressor.

The main objective of this computer science project is to use regression-based machine learning models to use bioactivity data and SMILES notation of drugs and molecules to predict the IC50 and pIC50 values. This will give valuable insight into the properties of the drug such as toxicity and potency in a cost-effective, accurate, and timely manner. The model will be tested for how well it can predict and fit the situation. This will undoubtedly allow for quicker drug development for scientists and allow for the saving of millions of lives.

Experimental Design

Location

This experiment will be executed indoors at home or school with access to the internet. It will take place on a 16-inch Apple Macbook Pro. This laptop is equipped with an Apple Silicon M2 Pro chip, 16 gigabytes of RAM, and 1 terabyte of SSD storage.

Constraints

Since this experiment is solely based on programming, there are no independent variables, dependent variables, or controls to consider. Instead of this, the project will be tested on accuracy metrics. One limitation to consider is the amount of data available. Many programs related to machine learning—such as the regression models in this project— require a training dataset to allow the program to learn how to identify what is being shown. A testing set is also required to calculate the accuracy of this program and to ensure that the program is functioning correctly. This creates a great demand for large datasets. Since datasets for bioactivity are not widely available (and when they are, they are very small), the program will not be able to use the most optimal amount of data, thereby unable to reach its full potential.

Safety

This entire project revolves around using a computer to program a machine learning model. The testing and training will also occur on a laptop. Therefore, there are no safety concerns to consider for the execution of this experiment.

Materials

For this experiment, a laptop or a type of personal computer able to run Google Colab is required. In this experiment, a 16-inch Apple Macbook Pro equipped with an Apple Silicon M2

Pro chip, 16 gigabytes of RAM, and 1 terabyte of SSD storage will be used. Since this experiment is performed using Google Colab, most computers will work. Next, the SARS Coronavirus 3C-like proteinase data needs to be downloaded from the ChEMBL Database. A Google account will be necessary to utilize Google Colab, but no installation is required as it can be accessed without any downloads. Finally, the necessary libraries need to be installed: Pandas, Numpy, RDkit, Matplotlib, PaDEL, Seaborn, SkLearn, ChEMBL web source, and LazyRegressor.

Procedure 1: Get Bioactivity Data

Once the ChEMBL web source is installed, it can be accessed within the code editor, and SARS Coronavirus 3C-like proteinase data can be downloaded for future use. Using the target.search function the data can be found for SARS Coronavirus 3C-like proteinase and saved as a CSV file. The dataset may have some missing data or other problems. To filter these out, the data frame drop function can be used. Next, create a new data frame with only the molecule ChEMBL ID, canonical smiles, and standard value data by using the selection function. The rest of the data is not necessary for this project. Next, we must identify each molecule's bioactivity as inactive, active, or intermediate. If the standard value for a molecule is above 10000 it can be considered active, if it is under 1000 it can be considered inactive, and any values in between are intermediates. Now save the new data as a CSV file that can be accessed in the next steps.

Procedure 2: Exploratory Data Analysis/Descriptor (Lipiniski) Calculation

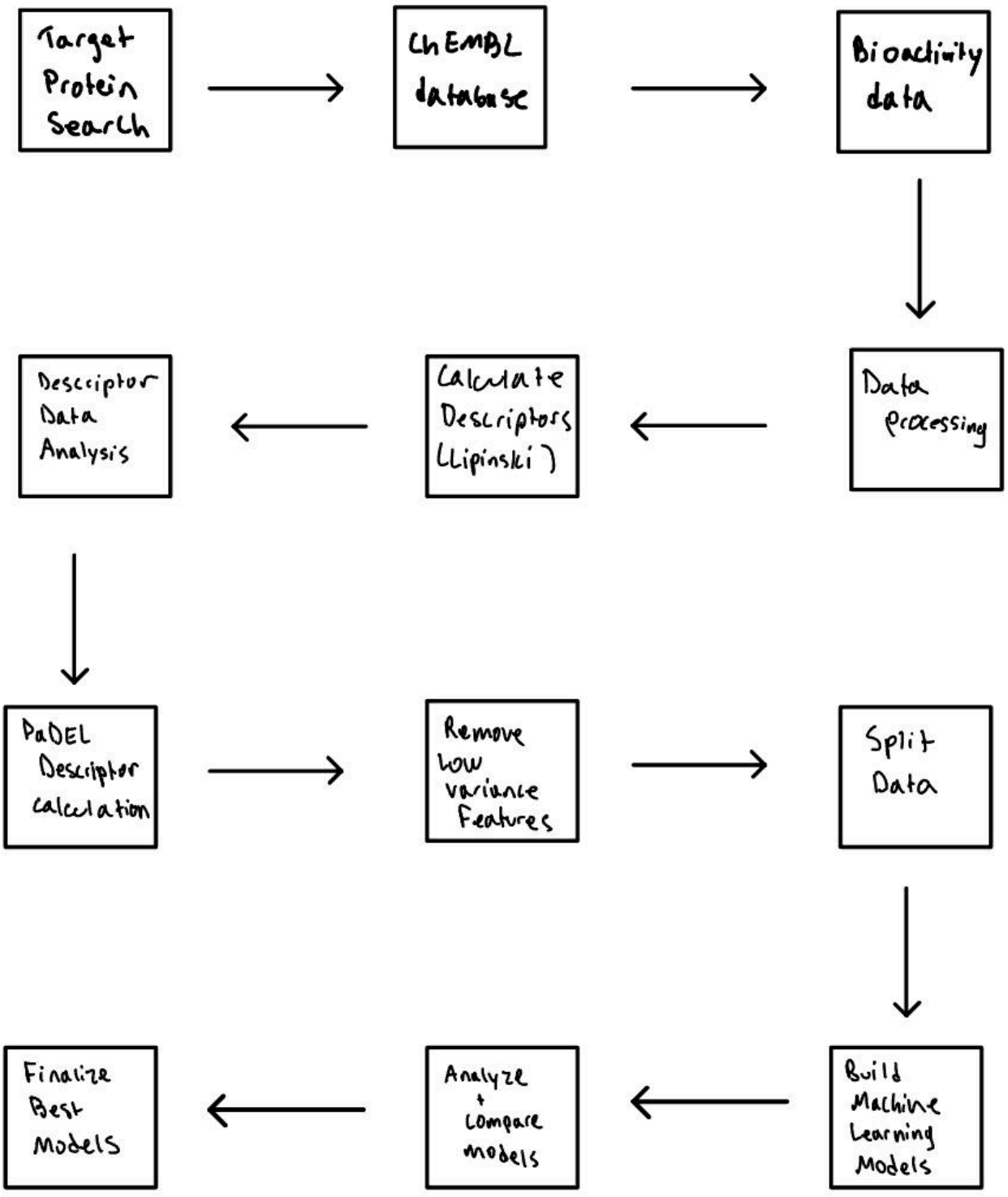
Once Rdkit has been installed and the previous steps have been completed import chem and lipinski from Rdkit. Next, use the Lipinski descriptor calculator outlined by Codeocean.com to calculate the descriptors (Codeocean.com, n.d.). This code can also be found in Appendix 2. Combine the outputted data with the CSV file from procedure 1. This dataset can be found in

Appendix 1. Next, to perform statistical analysis to determine which descriptors are indicative of active versus inactive, a Mann-Whitney U Test will be performed for statistical significance (Laerd, 2012). The alpha value will be 0.05. The code for this test can be found in Appendix 3. Perform the data analysis. Finally, the combined data will be converted into descriptor fingerprints using PaDEL for the regression model read as x and y values. Use the bash function to do this.

Procedure 3: Regression Models Creation

First import the created dataset from procedure 2 to use in the model creation. Next, remove the low variance data points by using the variance threshold function. Split the dataset into training and testing sets with 80 percent of the data going into training and the rest into testing using the split function. Define the x and y for training and testing and then use the LazyRegressor function to train and test models with the dataset. Output the results and choose the best model to model the pIC50 and IC50 values for molecules for SARS Coronavirus 3C-like proteinase.

Experiment Visualized:



Data Collection/Statistical Tests:

Descriptor data analysis

It is essential to know which of the descriptors calculated influence whether or not a molecule or drug is active or inactive. A Mann-Whitney U test is one method to determine this. It will determine the significance of the descriptor by calculating a p-value and seeing if it is smaller than the alpha value (alpha=0.05) that will be used. If the p-value is smaller than the alpha value, there is a statistically significant difference. Example data table:

	Descriptor	Statistics	p	alpha	Interpretation
0	LogP	-	-	0.05	Fail to reject null hypothesis or reject null hypothesis

Machine learning regression model data analysis:

R-squared values and root mean squared error (RMSE) scores will be used to evaluate the model's performance. The R-squared value will describe how well the regression model predicts the data on a scale from 0 to 1 where 1 is perfect. In other words, it shows how well the model fits the data. The RMSE score displays the average difference between the model's prediction and the actual value. The goal of this project is to achieve an R-squared value of at least 0.9 and a maximum RMSE value of 0.5.

Model	R-Squared Score	RMSE Score
-	-	-

Data Analysis/Results

Figure 1a. Amount Active and Inactive Molecules in dataset

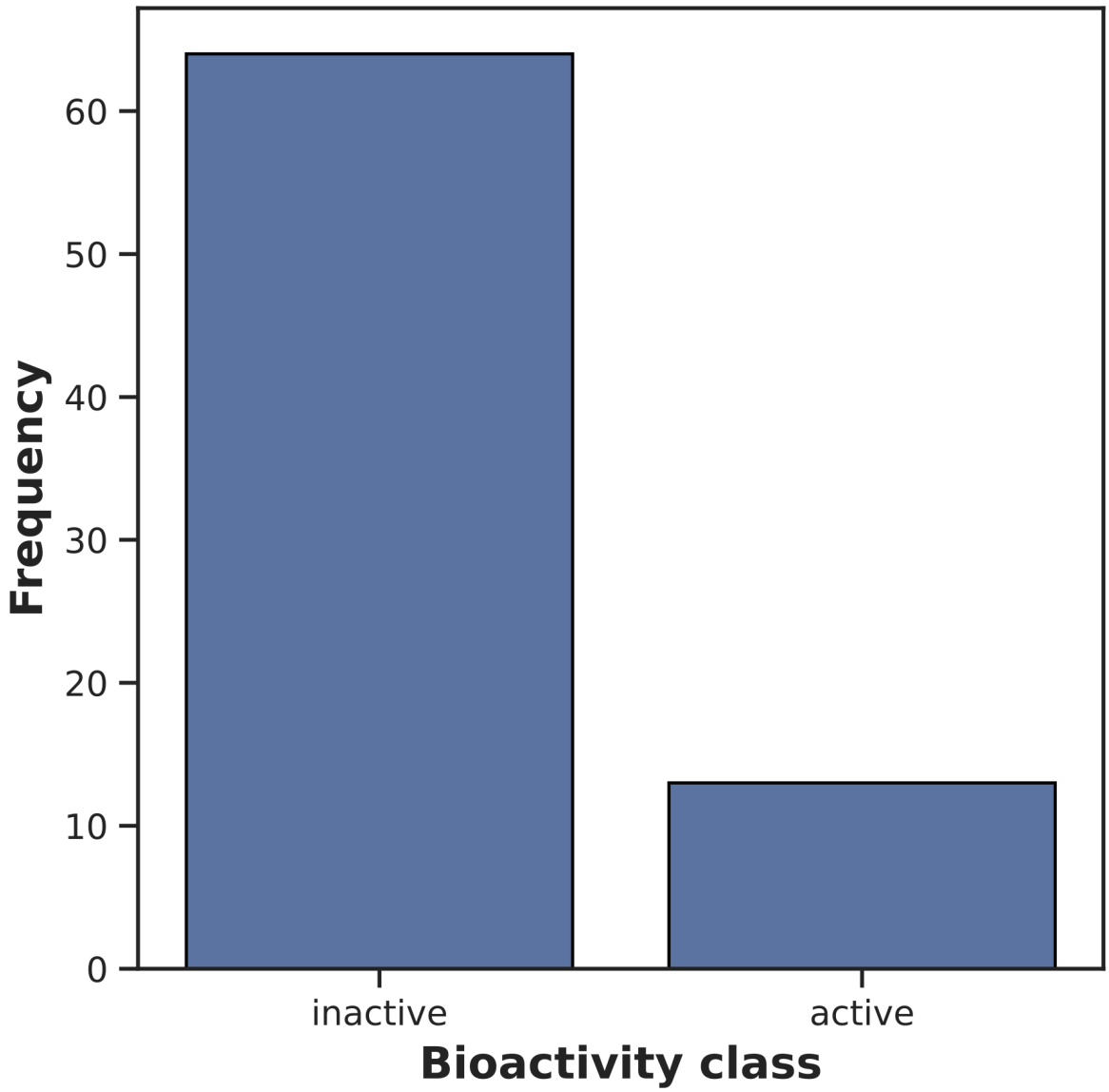


Figure 1a displays the quantity of inactive and active molecules within the dataset. 13 molecules are active at the target protein and 65 are inactive molecules. There was a total of 78 molecules.

Table 1a. Statistical Significance of pIC50

	Descriptor	Statistics	p	alpha	Interpretation
0	pIC50	832.0	1.58809447522824E-08	0.05	Different distribution (reject H0)

This table displays the result of the Mann-Whitney U test for pIC50. There was a statistically significant difference between the pIC50 value for active and inactive molecules and drugs. This can be used as a distinguisher in the machine learning regression model.

Table 1b. Statistical Significance for Number of Hydrogen Bond Donors

	Descriptor	Statistics	p	alpha	Interpretation
0	NumHDonors	178.5	0.0006202477871294170	0.05	Different distribution (reject H0)

This table displays the result of the Mann-Whitney U test for the number of hydrogen bond donors. There was a statistically significant difference between the number of hydrogen bond donors for active and inactive molecules and drugs. This can be used as a distinguisher in the machine learning regression model.

Table 1c. Statistical Significance for Number of Hydrogen Bond Accepters

	Descriptor	Statistics	p	alpha	Interpretation
0	NumHAcceptors	248.0	0.020963972338247300	0.05	Different distribution (reject H0)

This table displays the result of the Mann-Whitney U test for the number of hydrogen bond acceptors. There was a statistically significant difference between the number of hydrogen bond

acceptors for active and inactive molecules and drugs. This can be used as a distinguisher in the machine learning regression model.

Table 1d. Statistical Significance of Molecular Weight

	Descriptor	Statistics	p	alpha	Interpretation
0	MW	249.0	0.023566956483549300	0.05	Different distribution (reject H0)

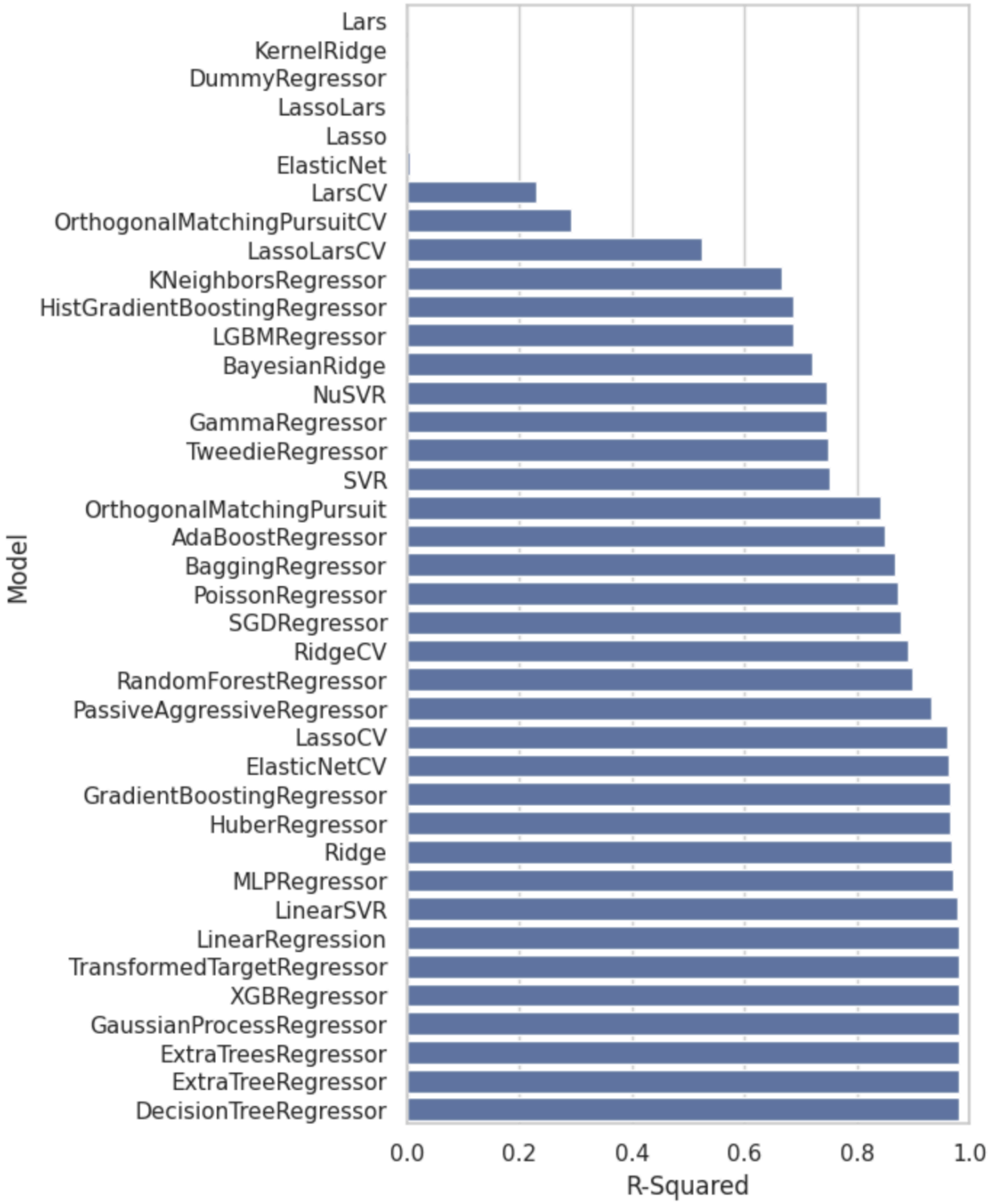
This table displays the result of the Mann-Whitney U test for molecular weight. There was a statistically significant difference between the molecular weight of active and inactive molecules and drugs. This can be used as a distinguisher in the machine learning regression model.

Table 1e. Statistical Significance of LogP

	Descriptor	Statistics	p	alpha	Interpretation
0	LogP	425.0	0.9079804356663140	0.05	Same distribution (fail to reject H0)

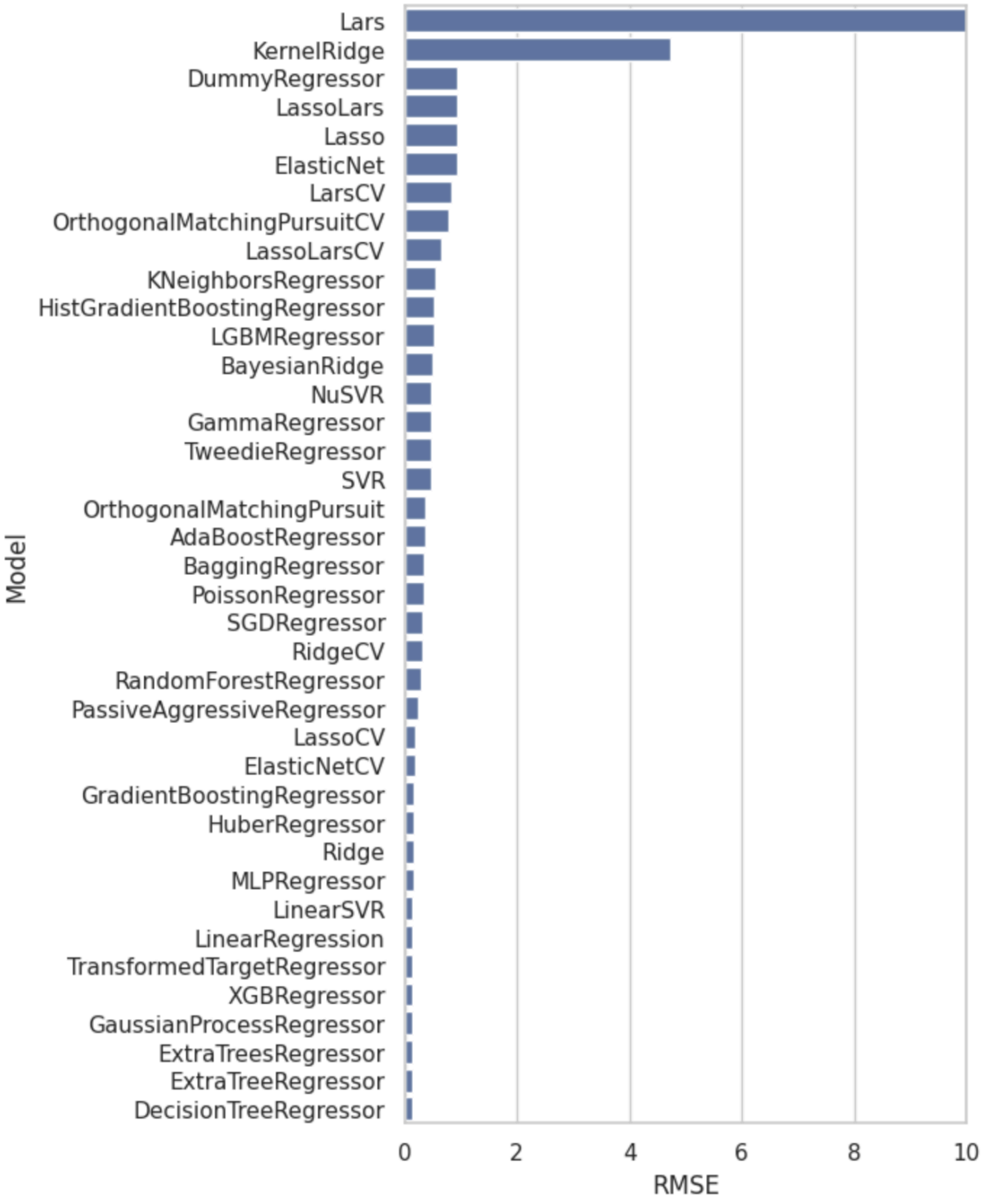
This table displays the result of the Mann-Whitney U test for druglikeness (LogP). There was not a statistically significant difference between the LogP of active and inactive molecules and drugs. This cannot be used as a distinguisher in the machine learning regression model.

Figure 2a. R-Sqaured values of the machine learning models during training



DecisionTreeRegressor performed the best on this data.

Figure 2b. RMSE values of the machine learning models during training

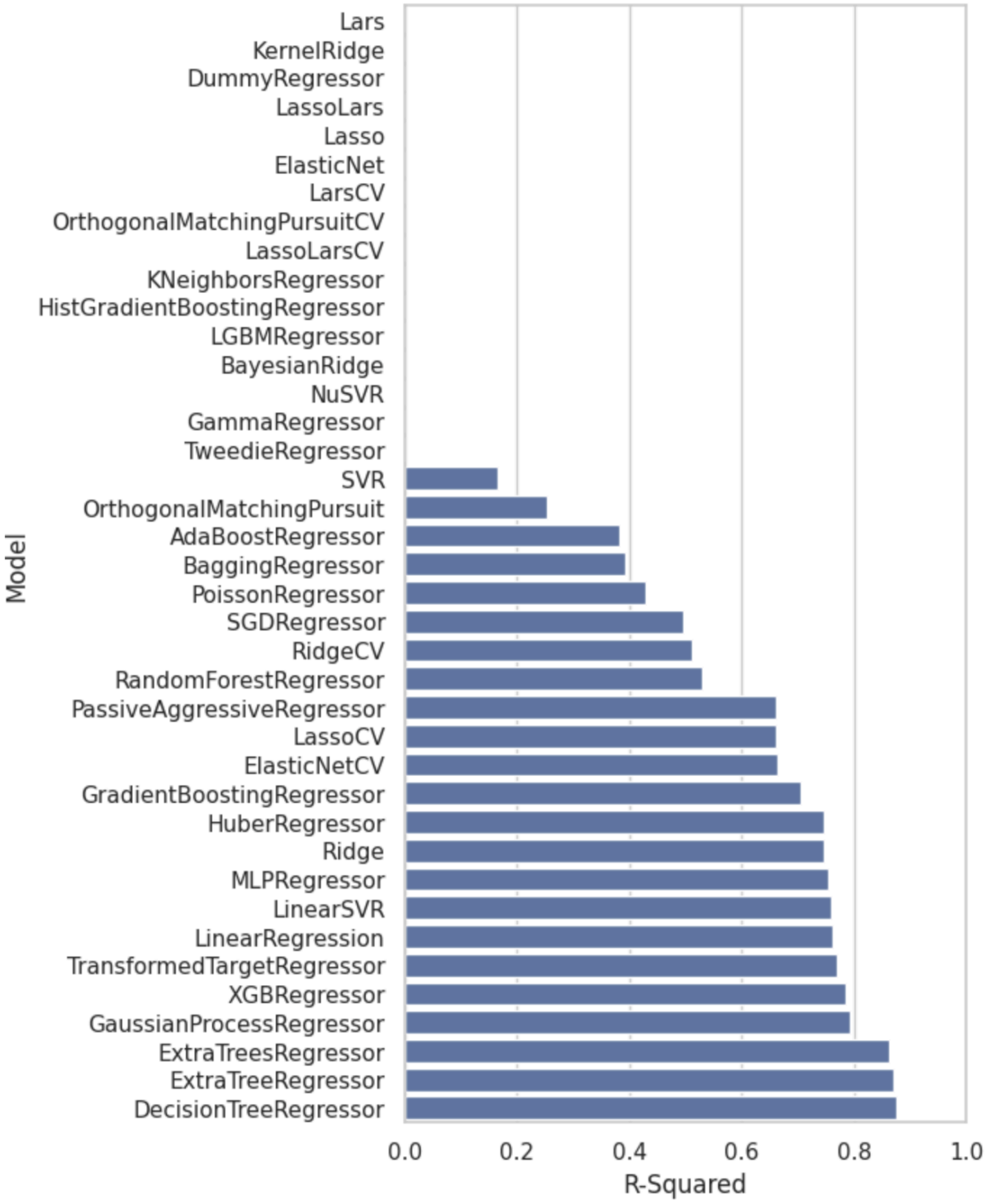


DecisionTreeRegressor performed the best on this data.

Table 2a. R-squared and RMSE values of the best model during training

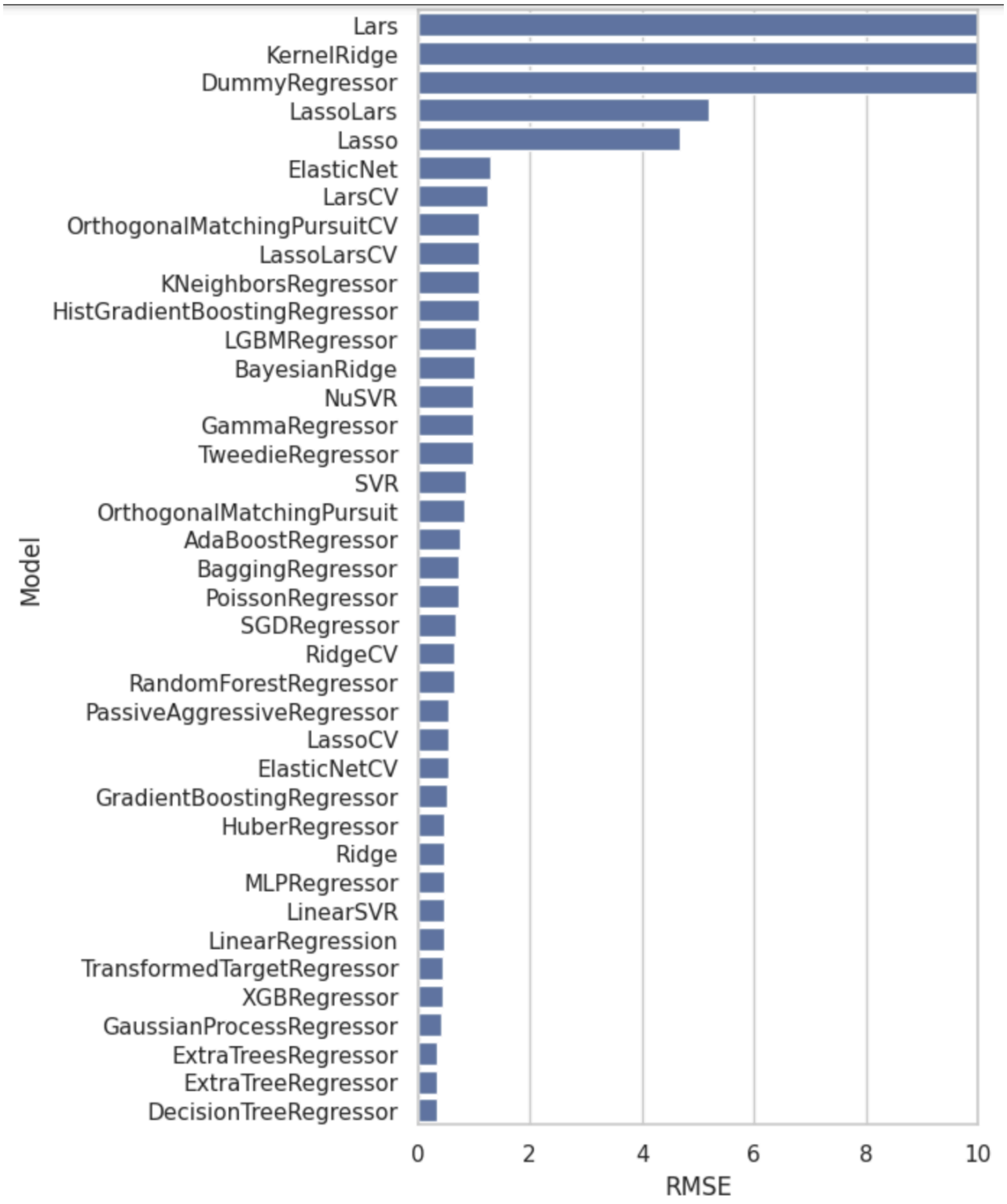
Model	R-Squared Score	RMSE Score
DecisionTreeRegressor	0.98	0.13

Figure 3a. R-Sqaured values of the machine learning models during testing



DecisionTreeRegressor performed the best on this data.

Figure 3b. RMSE values of the machine learning models during testing



DecisionTreeRegressor performed the best on this data.

Table 3a. R-squared and RMSE values of the best model during testing

Model	R-Squared Score	RMSE Score
DecisionTreeRegressor	0.9019	0.34

Discussion

As scientists face more difficulties in drug development whether it be during the analysis of properties or simply with time and cost of their research, this research aims to solve those problems by creating a regression-based machine learning model that utilizes bioactivity data and the molecule or drug's SMILES string to predict pIC50 and IC50 values. These pIC50 and IC50 values can be used to interpret a drug's potency and thus toxicity. This information is crucial in the H2L and lead discovery stages. This project will also save money and time in these stages of drug discovery.

Based on the results of the research, it can be said that models such as this regression-based machine learning model for SARS Coronavirus 3C-like proteinase are a plausible route to take in the journey of improving conventional drug discovery practices. This model was trained on chemical footprints determined by the descriptors. The statistical tests show that all descriptors except for druglikeness (LogP) contribute (they create a statistically significant difference as their p-values are less than 0.05) to whether or not a molecule will be active or inactive and thereby have an impact on pIC50 and IC50 values. Next, the dataset was split into training and testing sets to begin the creation, training, and testing of the machine learning models. Using LazyRegressor and the dataset, many models were created and analyzed by their RMSE and R-squared values. The best model after testing was determined to be the

DecisionTreeRegressor machine learning model. This model had the lowest RMSE value (0.34) and the highest R-squared value (0.9019). Having a large R-squared value is favored as that implies that the model fits the situation well and the predictions are accurate because the predictions are closer to the line of best fit (the actual values). RMSE is the average distance that all of the predictions are from the actual values. Having lower RMSE values is better because it would imply that the predictions are not very far off from the actual values. Together the low RMSE value and high R-squared value convey that the developed model is accurate and fast whilst also being cost-effective.

One limitation that greatly hindered the performance of this model was the amount of data available for the target protein of SARS Coronavirus 3C-like proteinase. Machine learning models and artificial intelligence models in general require immense amounts of data to go off to create better predictions and learn properly during training. This is especially true if there is significant variance within the data. The database utilized in this project had 78 data points, which is far too few to reach accuracies above 95 to 97 percent. Hence, this model did not reach higher accuracies. Overall, at this point, since the developed model has not reached an accuracy of higher than 95 percent, the models developed in this project can be used as more of an aid and supplement to scientists to make preliminary judgments and to confirm that the data they receive is accurate.

Implications and Applications

An important idea to note is that the regression model developed within this project is not only limited to molecules for the SARS Coronavirus 3C-like proteinase target protein. These models are very versatile in their uses. The same process can be applied to any target protein and

drug data. The only change that would be necessary is changing the dataset. By doing this, high-performance models for all drugs and target proteins can be developed.

Future Research

In the future, similar regression-based machine learning models can be expanded to other fields and even areas of drug discovery rather than focusing on the target protein SARS Coronavirus 3C-like proteinase. Work can also be done towards optimizing the current model or even testing new models with the same situation to look for ways to improve accuracy and decrease prediction error. The main action that can be taken to improve the accuracy of these types of models is gathering more data. Machine learning algorithms rely heavily on training and testing data. Without ample data, it is not possible to create models that reach accuracies of 97 percent higher. Finally, to improve the usability of these models, web applications can be formed to streamline the process of entering data and outputting a prediction.

Conclusion

In the world we live in today, the field of drug discovery is riddled with difficulties whether it be with the property analysis of drugs or the immense cost of clinical trials. The ultimate goal of this project was to develop and test regression-based machine learning models that could make predictions about drugs and molecules designed for SARS Coronavirus 3C-like proteinase while also being accurate, fast, and cost-effective. The models would predict the potency and toxicity by outputting an IC50 value and pIC50 value for the molecule. These models made predictions based on SMILES notation and bioactivity data of the molecules and drugs from the database ChEMBL. This data was first processed to remove any unnecessary data

and to ensure it was readable by the machine learning models. Then, the data was split into training and testing data and imported into LazyRegressor to start training the various types of regression models. In the end, the best regression-based machine learning model, DecisionTreeRegressor achieved an RMSE value of 0.34 and a R-squared value of 0.9019. This shows that the model was fairly accurate while also being cost-effective and quick. The results could have been improved with the inclusion of more data about SARS Coronavirus 3C-like proteinase. Despite the lack of data and average accuracy of less than 95 percent, we deem this project to be a success. This is because although it did not reach a very high accuracy it was still fairly accurate considering the constraints. In addition, this project proved that using these regression-based machine learning models is a possible avenue in improving the field of drug discovery. At the very least, the model developed in this project can be used as more of an aid and supplement to scientists to make preliminary judgments and to confirm that the data they receive is accurate. By continuing to build on the work conducted in this project we can continue to improve the field of drug discovery and save millions of lives.

Appendix

Appendix 1
Precessed dataset used for machine learning regression model training and testing

molecule_chembl_id	canonical_smiles	class	MW	LogP	NumHDonors	NumHAcceptors	pIC50
CHEMBL187579	Cc1noc(C)c1CN1C(=O)C(=O)c2cc(C#N)ccc21	intermediate	281.271	1.89262	0.0	5.0	5.142667503568730
CHEMBL188487	O=C1C(=O)N(Cc2ccc(F)cc2Cl)c2ccc(I)cc21	intermediate	415.589	3.8132000000000000	0.0	2.0	5.026872146400300
CHEMBL185698	O=C1C(=O)N(CC2COc3ccccc3O2)c2ccc(I)cc21	inactive	421.19000000000000	2.6605000000000000	0.0	4.0	4.869666231 5049900
CHEMBL426082	O=C1C(=O)N(Cc2cc3ccccc3s2)c2ccccc21	inactive	293.34700000000000	3.6308000000000000	0.0	3.0	4.882397308 3099200
CHEMBL187717	O=C1C(=O)N(Cc2cc3ccccc3s2)c2c1ccc c2[N+](=O)[O-]	intermediate	338.34400000000000	3.5390000000000000	0.0	5.0	5.698970004 336020

CHEMBL365134	O=C1C(=O)N(Cc2cc3ccccc3s2)c2c(Br)ccc21	active	372.24300000000000	4.393300000000000	0.0	3.0	6.008773924307510
CHEMBL187598	O=C1C(=O)N(Cc2cc3ccccc3s2)c2ccc(F)cc21	intermediate	311.33700000000000	3.769900000000000	0.0	3.0	5.31695296176115
CHEMBL190743	O=C1C(=O)N(Cc2cc3ccccc3s2)c2ccc(I)cc21	active	419.24300000000000	4.235400000000000	0.0	3.0	6.022276394711150
CHEMBL365469	O=C1C(=O)N(Cc2cc3ccccc3s2)c2cccc(Cl)c21	inactive	327.79200000000000	4.284200000000000	0.0	3.0	4.950781977329820
CHEMBL188983	O=C1C(=O)N(C/C=C/c2cc3ccccc3s2)c2ccc(I)cc21	inactive	445.28100000000000	4.748600000000000	0.0	3.0	4.628932137728260
CHEMBL191575	O=C(Nc1ccc(Cl)cc1)c1ccc(CN2C(=O)C(=O)c3cc(I)ccc32)s1	inactive	522.75100000000000	4.987900000000000	1.0	4.0	4.900664722314040
CHEMBL370923	O=C1C(=O)N(Cc2ccc(C(=O)N3CCCCC3)s2)c2ccc(I)cc21	inactive	480.327	3.708300000000000	0.0	4.0	4.756961951313710
CHEMBL194398	CCOC(=O)/C=C/[C@H](C[C@@H]1CCNC1=O)NC(=O)[C@@H](CC(=O)[C@@H](NC(=O)c1cc(C)on1)C(C)C)Cc1ccccc1	inactive	580.68200000000000	2.685820000000000	3.0	8.0	4.346787486224660
CHEMBL196635	CCOC(=O)/C=C/[C@H](C[C@@H]1CCNC1=O)NC(=O)[C@H](CC=C(C)C)CC(=O)[C@@H](NC(=O)c1cc(C)on1)C(C)C	inactive	558.67600000000000	2.799420000000000	3.0	8.0	4.154901959985740
CHEMBL209287	CCCCN1C(=O)C(=O)c2cc(I)ccc21	inactive	329.137	2.620600000000000	0.0	2.0	4.180456064458130
CHEMBL358279	NC(=O)c1ccc2c(c1)C(=O)C(=O)N2Cc1ccc2ccccc2c1	active	330.34300000000000	2.668200000000000	1.0	3.0	6.431798275933010
CHEMBL348660	NC(=O)c1ccc2c(c1)C(=O)C(=O)N2Cc1ccccc1	inactive	280.28300000000000	1.515000000000000	1.0	3.0	4.903089986991940
CHEMBL379727	CCCCN1C(=O)C(=O)c2cc(C(N)=O)ccc21	inactive	246.26600000000000	1.114900000000000	1.0	3.0	4.721246399047170
CHEMBL210525	CCCN1C(=O)C(=O)c2cc(C(N)=O)ccc21	inactive	232.23900000000000	0.7248	1.0	3.0	4.6020599913279600
CHEMBL148483	CN1C(=O)C(=O)c2cc(C(N)=O)ccc21	inactive	204.18500000000000	-0.055400000000000030	1.0	3.0	4.148741651 2809200
CHEMBL383725	O=C1C(=O)N(Cc2ccc3ccccc3c2)c2ccc(I)cc21	intermediate	413.21400000000000	4.173900000000000	0.0	2.0	5.958607314841780
CHEMBL118596	O=C1C(=O)N(Cc2ccccc2)c2ccc(I)cc21	inactive	363.15400000000000	3.020700000000000	0.0	2.0	4.301029995663980
CHEMBL208732	O=C(CSc1nccc(-c2csc(-c3ccccc3)n2)n1)Nc1cc(Cl)cc(Cl)c1	intermediate	473.41000000000000	6.304700000000000	1.0	6.0	5.5200002474083600
CHEMBL210146	COc1cccc(-c2nc(SCC(=O)Nc3ccc([N+](=O)[O-])cc3)nc(O)c2C#N)c1	inactive	437.43700000000000	3.368480000000000	2.0	9.0	3.5200000029339800
CHEMBL207458	CCOC(=O)c1cnc(SCC(=O)Nc2ccc([N+](=O)[O-])cc2)nc1N	inactive	377.38200000000000	1.874500000000000	2.0	9.0	3.6000000054478700
CHEMBL207484	COC(OC)c1cc(O)nc(SCC(=O)Nc2ccc(C(F)(F)F)cc2)n1	inactive	403.38200000000000	3.223200000000000	2.0	7.0	3.690000009501580
CHEMBL207207	Cc1cc(O)nc(SCC(=O)Nc2cc(Cl)ccc2Oc2ccccc2)n1	inactive	401.87500000000000	4.667120000000000	2.0	6.0	4.0
CHEMBL210487	CCC(Sc1nc(O)c(C#N)c(-c2cccc(OC)c2)n1)C(=O)Nc1ccc(C(C)=O)cc1	inactive	462.53100000000000	4.441480000000000	2.0	8.0	4.219999989963100

CHEMBL380470	COc1ccc(NC(=O)CSc2nc(O)cc(-c3ccccc3)n2)cc1OC	inactive	397.45600000000000	3.597200000000000	2.0	7.0	4.339999990132750
CHEMBL210612	CC(C)c1ccc(NC(=O)CSc2nccc(-c3cccs3)n2)cc1	inactive	369.51500000000000	5.059300000000000	1.0	5.0	4.389999976337710
CHEMBL209667	O=C(CSc1nccc(-c2cc(-c3ccccc3Cl)no2)n1)Nc1ccc(Cl)cc1	inactive	457.34200000000000	5.836200000000000	1.0	6.0	4.820000071285180
CHEMBL210097	O=C(CSc1nccc(-c2cc(-c3ccc(Cl)cc3Cl)no2)n1)Nc1ccc(C(F)(F)F)cc1	inactive	525.33900000000000	6.855000000000000	1.0	6.0	4.820000071285180
CHEMBL378674	CSc1sc(-c2nc(C)cs2)c(C)c1-c1ccnc(SCC(=O)Nc2ccccc2Cl)n1	inactive	519.14200000000000	7.051540000000010	1.0	8.0	4.920000156997060
CHEMBL210216	Cc1nc(SCC(=O)Nc2ccc([N+](=O)[O-])cc2)nc(O)c1C	inactive	334.35700000000000	2.43804	2.0	7.0	3.0
CHEMBL210195	Cc1cc(C(F)(F)F)nc(SCC(=O)Nc2ccc(Cl)cc2F)n1	inactive	379.766	4.327120000000000	1.0	4.0	3.300000003143150
CHEMBL210437	COc1ccc(-c2ccnc(SCC(=O)Nc3ccc(C(C)C)cc3)n2)cc1	inactive	393.51200000000000	5.006400000000000	1.0	5.0	3.389999997659040
CHEMBL378677	CC(C)(C)c1ccc(NC(=O)CSc2nc(O)c(C#N)c(C3CCCCC3)n2)cc1	inactive	424.57000000000000	5.129880000000000	2.0	6.0	3.449999999006189
CHEMBL210972	CC(Sc1nc(O)c(C#N)c(-c2ccccc2)n1)C(=O)Nc1ccc(Cl)cc1	inactive	410.88600000000000	4.493580000000000	2.0	6.0	3.5200000029339800
CHEMBL210145	CCOc1ccc(N2C(=O)CC(Sc3nc(C)cc(C)n3)C2=O)cc1	inactive	357.43500000000000	2.916340000000000	0.0	6.0	3.5200000029339800
CHEMBL377225	N#Cc1c(O)nc(SCC(=O)Nc2cccc(C(F)(F)F)c2)nc1-c1ccccc1	inactive	430.41100000000000	4.470480000000000	2.0	6.0	3.690000009501580
CHEMBL210823	CCCCc1ccc(NC(=O)CSc2nc(O)c(C#N)c(C3CCCCC3)n2)c(C)c1	inactive	438.59700000000000	5.483400000000010	2.0	6.0	3.690000009501580
CHEMBL207381	CC(C)c1ccc(NC(=O)CSc2nccc(-c3csc(COc4ccccc4Cl)n3)n2)cc1	inactive	511.07200000000000	6.686700000000010	1.0	7.0	3.690000009501580
CHEMBL210511	Cc1cc(O)nc(SCC(=O)Nc2ccc(Oc3ccc(Cl)cc3)cc2)n1	inactive	401.87500000000000	4.667120000000000	2.0	6.0	3.690000009501580
CHEMBL210632	COc1cccc(-c2nc(SCC(=O)Nc3ccc(C(C)=O)cc3)nc(O)c2C#N)c1	inactive	434.47700000000000	3.662880000000000	2.0	8.0	4.219999989963100
CHEMBL210497	COc1cccc(-c2nc(SCC(=O)Nc3ccc(S(N)(=O)=O)cc3)nc(O)c2C#N)c1	inactive	471.52000000000000	2.10768	3.0	9.0	4.389999976337710
CHEMBL208584	CCCc1cc(O)nc(SCC(=O)Nc2ccc(Cl)cc2)n1	inactive	337.83200000000000	3.518900000000000	2.0	5.0	4.519999959791460
CHEMBL208763	O=C(CSc1nccc(-c2cc(-c3ccccc3)no2)n1)Nc1ccc(Cl)cc1	inactive	422.89700000000000	5.182800000000000	1.0	6.0	4.820000071285180
CHEMBL209227	Cc1nc(-c2nc(-c3ccnc(SCC(=O)Nc4ccc(Cl)cc4)n3)cs2)cs1	inactive	460.00900000000000	5.416220000000000	1.0	8.0	4.849999859991130
CHEMBL210092	CSc1sc(-c2nc(C)cs2)c(C)c1-c1ccnc(SCC(=O)Nc2ccc(Cl)cc2)n1	inactive	519.14200000000000	7.051540000000010	1.0	8.0	4.950000175844590
CHEMBL377150	Cn1nc(-c2ccc(-c3ccnc(SCC(=O)Nc4ccc(Cl)cc4)n3)s2)cc1C(F)(F)F	inactive	509.966	6.008600000000000	1.0	7.0	5.0
CHEMBL212454	O=C(Oc1ccc(S(=O)(=O)c2ccc(OC(=O)C(Cl)=C(Cl)Cl)cc2)cc1)C(Cl)=C(Cl)Cl	active	565.041	6.101200000000000	0.0	6.0	6.045757490560680
CHEMBL213581	Nc1ncc(S(=O)(=O)c2ccc(Cl)cc2)c(N)n1	intermediate	284.728	1.127200000000000	2.0	6.0	5.221848749616360

CHEMBL380403	O=[N+]([O-])c1cc(C(F)(F)F)ccc1S(=O)(=O)c1ccc(Cl)cc1	inactive	365.716	4.099800000000000	0.0	4.0	4.920818753952380
CHEMBL212504	Cc1nc(S(=O)(=O)c2ccccc2)c(C#N)c(C)c1[N+](=O)[O-]	inactive	317.326000000000	2.311120000000000	0.0	6.0	4.886056647693160
CHEMBL211969	Cc1ccc(S(=O)(=O)c2nc(C)c([N+](=O)[O-])c(C)c2C#N)cc1	inactive	331.353000000000000	2.619540000000000	0.0	6.0	4.886056647693160
CHEMBL384739	O=[N+]([O-])c1ccc(S(=O)(=O)c2ccc(Cl)cc2)[n+]([O-])c1	inactive	314.706	1.714400000000000	0.0	5.0	4.823908740944320
CHEMBL215732	CCOC(=O)/C(C#N)=C/Nc1ccc(S(=O)(=O)c2ccc(/N=C/C(C#N)=C(\O)OCC)cc2)cc1	inactive	494.529000000000	3.933860000000000	2.0	10.0	4.795880017344080
CHEMBL212240	O=C(O)c1ccc(S(=O)(=O)c2cc(Br)c(O)c(Br)c2)cc1	inactive	436.077000000000000	3.448200000000000	2.0	4.0	4.795880017344080
CHEMBL377253	CC(=O)c1ccccc1S(=O)(=O)c1ccccc1C(=O)O	inactive	304.323000000000000	2.420200000000000	1.0	4.0	4.795880017344080
CHEMBL215397	O=[N+]([O-])c1ccc(S(=O)(=O)c2ccc([N+](=O)[O-])cc2)cc1	inactive	308.271	2.3358	0.0	6.0	4.602059991327960 0
CHEMBL378342	CCOC(=O)C(=CNc1ccc(S(=O)(=O)c2ccc(/N=C/C(C(=O)OCC)=C(\O)OCC)cc2)cc1)C(=O)OCC	inactive	588.635000000000	4.012900000000000	2.0	12.0	4.494850021680090
CHEMBL379642	O=C(Sc1nnc(C(F)(F)F)[nH]1)c1ccc(C#Cc2ccccc2)o1	intermediate	363.320000000000	3.748800000000000	1.0	5.0	5.522878745280340
CHEMBL427404	Cc1noc(NC(=O)c2ccc(-c3cc(C(F)(F)F)nn3C)s2)c1[N+](=O)[O-]	intermediate	401.326000000000	3.624320000000000	1.0	8.0	5.301029995663980
CHEMBL212190	Cc1oc(C(C)(C)C)cc1-c1cc(NS(=O)(=O)c2cccs2)[nH]n1	inactive	365.48	4.137920000000000	2.0	5.0	5.0
CHEMBL378700	CSc1[nH]nc(NC(=O)c2cccs2)c1S(=O)(=O)c1ccccc1	inactive	379.488000000000	3.278200000000000	2.0	6.0	4.823908740944320
CHEMBL212019	CC1(C)CC(=O)c2c(NCc3ccco3)sc(C#N)c2C1	inactive	300.383000000000	3.979980000000000	1.0	5.0	4.795880017344080
CHEMBL212399	CSc1nn(-c2c([N+](=O)[O-])c(C)nn2C)c(-c2cccs2)c1C#N	inactive	360.424000000000000	3.144500000000000	0.0	9.0	4.744727494 8966900
CHEMBL215733	O=S(=O)(Cc1[nH]c(-c2ccc(Cl)s2)c[s+]1)c1cccs1.[Br-]	inactive	442.834000000000000	1.7786	1.0	4.0	4.744727494 8966900
CHEMBL375130	Cc1nn(C)c(NCc2ccc(-c3cccs3)s2)c1[N+](=O)[O-]	inactive	334.426000000000	4.038820000000000	1.0	7.0	4.698970004336020
CHEMBL214372	O=C(Cc1nccs1)c1nccs1	inactive	210.283000000000000	2.025	0.0	5.0	4.397940008 6720400
CHEMBL212218	Cc1cc(S(=O)(=O)c2c([N+](=O)[O-])cc(C(F)(F)F)cc2[N+](=O)[O-])c(Cl)cc1Cl	active	459.185000000000	4.969820000000000	0.0	6.0	6.522878745280340
CHEMBL222840	O=C(Oc1cncc(Cl)c1)c1ccco1	active	223.615	2.5472	0.0	4.0	7.221848749616360
CHEMBL222769	O=C(Oc1cncc(Cl)c1)c1ccc(-c2ccc(Cl)cc2)o1	active	334.158	4.867600000000000	0.0	4.0	7.200659450546420
CHEMBL222735	COc1cccc(C(=O)Oc2cncc(Cl)c2)c1	active	263.68	2.962800000000000	0.0	4.0	6.468521082957750
CHEMBL222628	O=C(Oc1cncc(Cl)c1)c1cscn1	active	240.671000000000000	2.410700000000000	0.0	5.0	6.568636235841010

CHEMBL222893	O=C(Oc1cncc(Cl)c1)c1cc2ccccc2s1	active	289.74300000000000	4.16890000000000	0.0	4.0	7.022276394711150
CHEMBL225515	O=C(Oc1cncc(Cl)c1)c1cc2ccccc2[nH]1	active	272.69100000000000	3.43550000000000	1.0	3.0	7.187086643357140
CHEMBL222234	O=C(Oc1cncc(Br)c1)c1ccco1	active	268.066	2.65630000000000	0.0	4.0	7.301029995663980
CHEMBL426898	O=C(Oc1cncc(Cl)c1)c1cc2ccccc2o1	active	273.67500000000000	3.70040000000000	0.0	4.0	6.769551078621730
CHEMBL187266	Cc1coc2c1C(=O)C(=O)c1c-2ccc2c1CCCC2(C)C	inactive	294.35	4.24792000000000	0.0	3.0	4.050122295963130
CHEMBL215254	Cc1coc2c1C(=O)C(=O)c1c-2ccc2c1CCCC2(C)CO	inactive	310.34900000000000	3.22032000000000	1.0	4.0	4.605548319173780
CHEMBL2146517	COC(=O)[C@@]1(C)CCCc2c1ccc1c2C(=O)C(=O)c2c(C)coc2-1	inactive	338.35900000000000	3.40102000000000	0.0	5.0	4.675717544702310
CHEMBL187460	C[C@H]1COC2=C1C(=O)C(=O)c1c2ccc2c1CCCC2(C)C	inactive	296.366	3.44330000000000	0.0	3.0	3.6445484798734800
CHEMBL363535	Cc1coc2c1C(=O)C(=O)c1c-2ccc2c(C)cccc12	inactive	276.291	4.09564000000000	0.0	3.0	4.412289034981090
CHEMBL227075	Cc1cccc2c3c(ccc12)C1=C(C(=O)C3=O)[C@@H](C)CO1	inactive	278.307	3.29102000000000	0.0	3.0	4.841637507904750
CHEMBL45830	CC(C)C1=Cc2ccc3c(c2C(=O)C1=O)CCCC3(C)C	inactive	282.38300000000000	4.10530000000000	0.0	2.0	4.675717544702310

Appendix 2

Code for Lipinski descriptor calculation

```
def lipinski(smiles, verbose=False):

moldata= []
for elem in smiles:
mol=Chem.MolFromSmiles(elem)
moldata.append(mol)
baseData= np.arange(1,1)
i=0
for mol in moldata:
desc_MolWt = Descriptors.MolWt(mol)
desc_MolLogP = Descriptors.MolLogP(mol)
desc_NumHDonors = Lipinski.NumHDonors(mol)
desc_NumHAcceptors = Lipinski.NumHAcceptors(mol)
row = np.array([desc_MolWt,
desc_MolLogP,
desc_NumHDonors,
desc_NumHAcceptors])
if(i==0):
```

```
baseData=row
else:
baseData=np.vstack([baseData, row])
i=i+1
columnNames=["MW","LogP","NumHDonors","NumHAcceptors"]
descriptors = pd.DataFrame(data=baseData,columns=columnNames)
return descriptors
```

Appendix 3

Mann-Whitney U Test for statistical significance code. For analysis use the function "mannwhitney("")." For example for pIC50 the function would be mannwhitney("pIC50").

```
def mannwhitney(descriptor, verbose=False):

from numpy.random import seed
from numpy.random import randn
from scipy.stats import mannwhitneyu

# seed the random number generator
seed(1)

# actives and inactives
selection = [descriptor, 'bioactivity_class']
df = df_2class[selection]
active = df[df.bioactivity_class == 'active']
active = active[descriptor]

selection = [descriptor, 'bioactivity_class']
df = df_2class[selection]
inactive = df[df.bioactivity_class == 'inactive']
inactive = inactive[descriptor]

# compare samples
stat, p = mannwhitneyu(active, inactive)
#print('Statistics=%.3f, p=%.3f' % (stat, p))

# interpret
alpha = 0.05
if p > alpha:
interpretation = 'Same distribution (fail to reject H0)'
```

```
else:
interpretation = 'Different distribution (reject H0)'
results = pd.DataFrame({'Descriptor':descriptor,
'Statistics':stat,
'p':p,
'alpha':alpha,
'Interpretation':interpretation}, index=[0])
filename = 'mannwhitneyu_' + descriptor + '.csv'
results.to_csv(filename)

return results
```

References

Alexandre Blanco-González, Cabezón, A., Seco-González, A., Conde-Torres, D., Antelo-Riveiro, P., Ángel Piñeiro, & Garcia-Fandino, R. (2023). The Role of AI in Drug Discovery: Challenges, Opportunities, and Strategies. *The Role of AI in Drug Discovery: Challenges, Opportunities, and Strategies*, *16*(6), 891–891. https://doi.org/10.3390/ph16060891

Alvarellos, M. (2023, October 5). *What are the current challenges of drug discovery?* Www.lifebit.ai. https://www.lifebit.ai/blog/current-challenges-of-drug-discovery

Aykul, S., & Martinez-Hackert, E. (2016). Determination of half-maximal inhibitory concentration using biosensor-based protein interaction analysis. *Analytical Biochemistry*, *508*(1), 97–103. https://doi.org/10.1016/j.ab.2016.06.025

Berrouet, C., Dorilas, N., Rejniak, K. A., & Tuncer, N. (2020). Comparison of Drug Inhibitory Effects (IC50) in Monolayer and Spheroid Cultures. *Bulletin of Mathematical Biology*, *82*(6). https://doi.org/10.1007/s11538-020-00746-7

ChEMBL. (n.d.). *ChEMBL Database*. Www.ebi.ac.uk. https://www.ebi.ac.uk/chembl/

Code Ocean. (n.d.). *Code Ocean*. Codeocean.com. Retrieved February 1, 2024, from https://codeocean.com/explore/capsules?query=tag:data-curation

IBM. (2023). *What is Machine Learning?* IBM. https://www.ibm.com/topics/machine-learning

isirv Antiviral group. (n.d.). *Analysis of IC50 data*. Www.isirv.org. Retrieved January 31, 2024, from https://www.isirv.org/site/index.php/methodology/analysis-of-ic50-data#:~:text=IC50%20values%20can%20be

Kurama, V. (2019, September 4). *Regression in Machine Learning: What it is and Examples of Different Models*. Built In. https://builtin.com/data-science/regression-machine-learning

Laerd Statistics . (2013). *Mann-Whitney U Test in SPSS Statistics*. Laerd Statistics. https://statistics.laerd.com/spss-tutorials/mann-whitney-u-test-using-spss-statistics.php

Liu, P., Li, H., Li, S., & Leung, K.-S. (2019). Improving prediction of phenotypic drug response on cancer cell lines using deep convolutional network. *BMC Bioinformatics*, *20*(1). https://doi.org/10.1186/s12859-019-2910-6

Navre, M. (2019, November 21). *Why using pIC50 instead of IC50 will change your life*. Www.collaborativedrug.com. https://www.collaborativedrug.com/cdd-blog/why-using-pic50-instead-of-ic50-will-change-your-life#:~:text=A%20closely%20related%20concept%20is

October 25, C. H., & 2015. (2015, October 15). *Delays in Drug Approval Are Deadly, Highlighting the Need for Improved Regulatory Efficiency - The ASCO Post*. Ascopost.com. https://ascopost.com/issues/october-25-2015/delays-in-drug-approval-are-deadly-

highlighting-the-need-for-improved-regulatory-efficiency/#:~:text=Worldwide%2C%201%20life%2Dyear%20was

Patil, R. (2021, May 6). *Lazy Predict - Best Suitable Model for You*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/05/lazy-predict-best-suitable-model-for-you/

Seldon. (2021, October 16). *Supervised vs Unsupervised Learning Explained*. Seldon. https://www.seldon.io/supervised-vs-unsupervised-learning-explained

Sharma, A. (2020, March 22). *A Comprehensive Guide to Google Colab*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2020/03/google-colab-machine-learning-deep-learning/#:~:text=Benefits%20of%20Google%20Colab

Sun, D., Gao, W., Hu, H., & Zhou, S. (2022). Why 90% of clinical drug development fails and how to improve it? *Acta Pharmaceutica Sinica B*, *12*(7). https://doi.org/10.1016/j.apsb.2022.02.002

US EPA. (2012). *Sustainable Futures / P2 Framework Manual 2012 EPA-748-B12-001 Appendix F. SMILES Notation Tutorial*. https://www.epa.gov/sites/default/files/2015-05/documents/appendf.pdf

Visikol. (2022). *The Importance of IC50 Determination | Visikol*. Visikol. https://visikol.com/blog/2022/06/07/the-importance-of-ic50-determination/

Zhou, S.-F., & Zhong, W.-Z. (2017). Drug Design and Discovery: Principles and Applications. *Molecules*, *22*(2), 279. https://doi.org/10.3390/molecules22020279