Data Science Minor Project
Area code 01

Comparative Analysis of Machine Learning Scoring Functions in Molecular Docking and Structure-Based Virtual Screens

A Dissertation Submitted
in Partial Fulfilment of the Requirements
for

MINOR DEGREE

in

DATA SCIENCE

by

Ira Zibbu
(Roll No. IMS19106)

to

SCHOOL OF DATA SCIENCE
INDIAN INSTITUTE OF SCIENCE EDUCATION AND RESEARCH
THIRUVANANTHAPURAM - 695 551, INDIA

April 2023

DECLARATION

I, **Ira Zibbu (Roll No: IMS19106)**, hereby declare that, this report entitled **"Comparative Analysis of Machine Learning Scoring Functions in Molecular Docking and Structure-based Virtual Screens"** submitted to the Indian Institute of Science Education and Research Thiruvananthapuram towards the partial requirement of **Minor Degree** in **Data Science**, is an original work carried out by me under the supervision of **Dr. Ramanathan Natesh** and has not formed the basis for the award of any degree or diploma, in this or any other institution or university. I have sincerely tried to uphold academic ethics and honesty. Whenever a piece of external information or statement or result is used then, that has been duly acknowledged and cited.

Thiruvananthapuram - 695 551 **Ira Zibbu**

April 2023

CERTIFICATE

This is to certify that the work contained in this project report entitled **"Comparative Analysis of Machine Learning Scoring Functions in Molecular Docking and Structure-Based Virtual Screens"** submitted by **Ira Zibbu (Roll No: IMS19106)** to the Indian Institute of Science Education and Research, Thiruvananthapuram towards the partial requirement of **Minor Degree** in **Data Science** has been carried out by her under my supervision and that it has not been submitted elsewhere for the award of any degree.

Thiruvananthapuram - 695 551 — Dr. Ramanathan Natesh

April 2023 — Project Supervisor

ACKNOWLEDGEMENT

I would like to first and foremost express my gratitude to my project supervisor Dr. Ramanathan Natesh, School of Biology, IISER Thiruvananthapuram for his guidance, informative feedback and constant support throughout the duration of this project.

I also wish to thank the School of Data Science for providing me with this opportunity to work on this project. I am lastly grateful to the Indian Institute of Science Education and Research Thiruvananthapuram for providing the necessary resources and facilities to complete this project to the best of my ability.

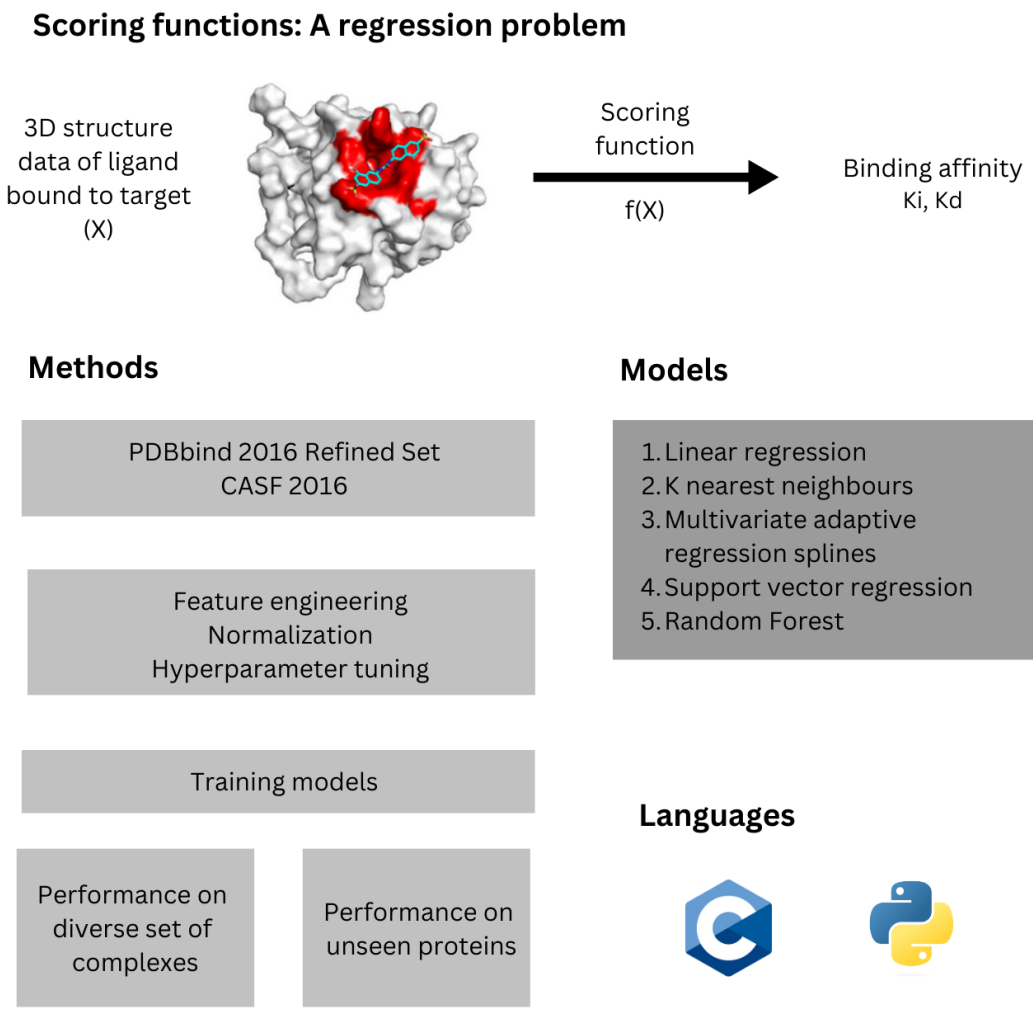
Thiruvananthapuram - 695 551 **Ira Zibbu**

ABSTRACT

Molecular docking is the main computational method underlying structure-based virtual screens (SBVS), and is a standard tool used in modern day drug discovery pipelines. Scoring functions (SFs) are mathematical predictive models underlying molecular docking which map the structure data of a protein-ligand complex to the binding affinity of the complex. Shortcomings associated with conventional SFs have led to a growing field of research on the use of machine learning (ML) models instead. Many ML SFs have been developed in the past, but insufficient effort has been devoted to comparing the performance of different models. Five regression ML models were compared on their ability to predict binding affinities for a diverse set of protein-ligand complexes. Performance of the models was also tested on unseen complexes absent in the training data. The results indicate that the best performing model varies with the nature of the test, and a there is no one-size-fits-all approach. In general, the non-parametric and non-linear ML models outperform linear regression, and the random forest ensemble model had the most robust performance. Model performance also usually drops when the query target is absent from the training set, and some proteins are not suited for binding affinity prediction. Overall, this project presents compelling data on the practical applications of ML SFs in molecular docking.

Keywords: Scoring functions, Machine Learning, Virtual Screens

GRAPHICAL ABSTRACT



Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Objective and Research Questions

. For a given target protein, very few ligands show significant binding, and despite advances in high-throughput screening of drug activity, a sufficient number of ligands cannot be screened to find suitable drugs. *In-silico* approaches such as molecular docking and structure-based virtual screens are emerging as a fast and cost-effective method to screen large libraries of ligands for activity against a given target. During ligand docking, the docking algorithm samples putative ligand active conformations and ligand binding modes to the target protein in the bound state. A scoring function (SF) is a mathematical or predictive model that outputs the binding affinity to represent binding free energy. The choice of scoring function can dramatically impact the outcomes of the docking process. Conventional SFs include force field, empirical or knowledge based SFs. The primary reason for the poor performance of conventional SFs is that they assume a predetermined functional form and do not adequately capture key non-covalent interactions. In contrast, machine learning scoring functions (ML SFs) are able to learn the scoring function from training data. Their inherent flexibility and non-parametric nature improve their performance. Many ML SFs have been developed in the past, but a uniform comparison of their performance has not been carried out on a large and recent data set. This project aims to fill this research gap, and answer the following questions:

1. Which ML models can be used to develop robust SFs for use in molecular

Data and code for this project an be found on https://github.com/ira-zibbu/Minor-Project-2013-ML-Scoring-Functions/

docking and SBVS?

2. Which ML models have the best scoring and ranking power?
3. How does the composition of the training data set affect the scoring power of ML SFs?

1.2 Computational Approaches to Drug Discovery and Design

Drug discovery is the process of identifying novel therapeutic compounds to treat or cure a disease. Drug discovery is a multi step process; the development of a single drug can take 12-15 years and $ 900 million - $ 2 billion [1]. The major challenges faced by modern drug discovery pipelines include high costs associated with the discovery process, long research and development (R & D) cycles and low rates of success in clinical trials. To address these challenges, computational approaches to drug discovery have been developed which can exploit the growing availability of biomedical and chemical data. In particular, these methods are valuable in the early stages of drug discovery [2].

Figure 1.1: An overview of modern drug discovery pipelines. ADMET, absorption, distribution, metabolism, excretion and toxicity. Adapted from [3]

The first step in the drug discovery pipeline is target identification. Here, a target refers to a biological entity that has been implicated in the cause or progression of the disease. Identification is typically carried out by studying genetic associations, analysis of omics data, or functional/phenotypic *in-vitro* or *in-vivo* screens (knock-downs, knockouts etc.) This is followed by target validation, where the target is demonstrated to have a functional role in producing the disease phenotype. Next, a variety of screening protocols are used for 'hit discovery', where candidate drugs with activity against the target are identified. Some examples of conventional

screening strategies are described below [4]

- High throughput screen (HTS): Large compound libraries are analysed in a biochemical assay for activity against the target. HTS perform a more exhaustive search for candidate drugs, but despite automation they are resource-intensive, time consuming and require the physical existence and synthesis of compound libraries.
- Focused screen: A restricted pool of compounds are screened based on prior knowledge of compounds against similar targets. Focused screens are cheaper and faster, but are less exhaustive and are prone to bias due assumptions made about the target.
- Physiological screen: A tissue-level approach to screen for drug activity against a target. These screens more accurately reflect physiological conditions of disease, but are of low-throughput.

Virtual screening is an attractive *in-silico* alternative to the physical screening techniques mentioned above. Virtual screens use molecular docking to interrogate a virtual compound library for binding with the target. They are fast and inexpensive, and do not require a laboratory setup to execute. Other computational techniques such as fragment screens and structural aided drug design are also available, but are outside the scope of this project. Beyond computational methods for screening, other *in-silico* techniques are also used in the drug discovery process such as quantitative structure activity relationship (QSAR) modelling, pharmacophore modelling, *de novo* drug design, ADMET (absorption, distribution, metabolism, excretion, and toxicity) property prediction. A description of these techniques and their applications is outside the purview of this project [5].

1.3 Virtual Screens

The primary goal of virtual screens is to filter a compound library down to a manageable number of compounds which can be feasibly physically screened. Advances in bioinformatics methods, high performance computing and curated biomedical databases have made virtual screens a powerful technique that decrease cost and time in early drug discovery. The protocol for virtual screen may vary depending on the drug discovery problem. There is usually a preparatory phase where a compound library is selected, ligands are prefiltered, and the ligand files are prepared [6]. Virtual screens can be divided into two main categories: [5]

- Structure based virtual screens (SBVS): SBVS (also known as target based virtual screening) searches and ranks ligands from a virtual compound library according to their ability to form complexes with a target biomolecule. SBVS requires the 3D structure of the target molecule [7].
- Ligand based virtual screens (LBVS): LBVS operates on the hypothesis that similar compounds have similar biological activity. It searches a compound library to identify drugs similar to those with known activity against the target [5].

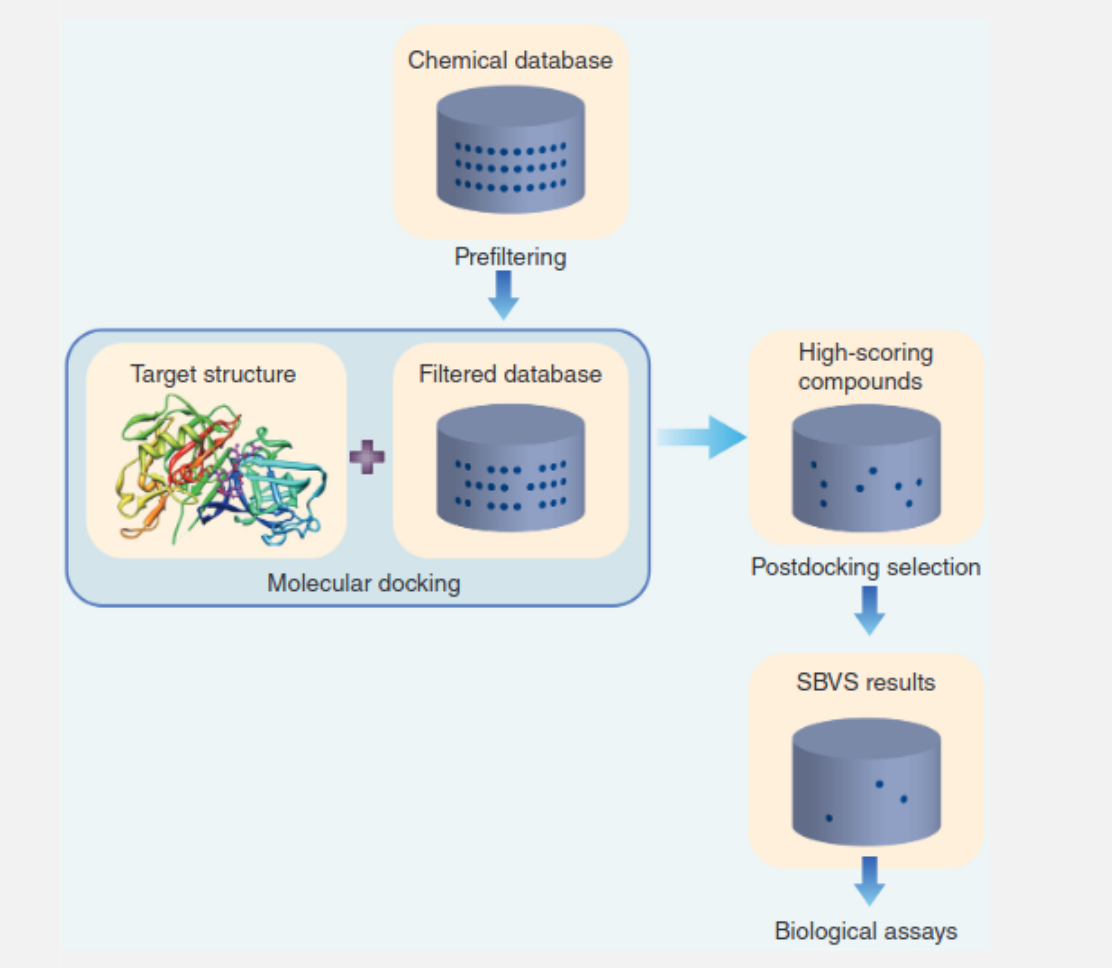


Figure 1.2: A schematic of structure-based virtual screens. A database of chemical compounds is prefiltered, and structural data of the target is used to find high-scoring compounds, which are subjected to biological assays for activity. Image source: [6]

Generally, since SBVS explicitly uses structural information about the target, it is regarded to be more accurate and effective than LBVS [8]. An SBVS campaign is counted as successful if one of the predicted ligands shows activity upon biological testing. This project focuses on SBVS, and in particular the use of molecular docking in SBVS. Molecular docking and SBVS have been successful in the past for hit discovery [6].

1.4 Molecular Docking

Molecular docking is a computational technique that predicts the bound conformation of a ligand to a protein and the binding affinity of this complex [9]. Docking has applications outside drug discovery in basic sciences such as predicting protein function and identifying allosteric sites [10]. The 3D structure of a ligand when it is bound is known as the **ligand active conformation**. The orientation of a ligand with respect to a target is called a **binding mode or binding pose**. In other words, for a particular protein-ligand pair, docking will produce ligand active conformations, binding pose and binding affinity. Some examples of popular docking programs are AutoDock Vina, GOLD, Glide and X-Score. There are two main components to molecular docking programs: the search algorithm and the scoring function.

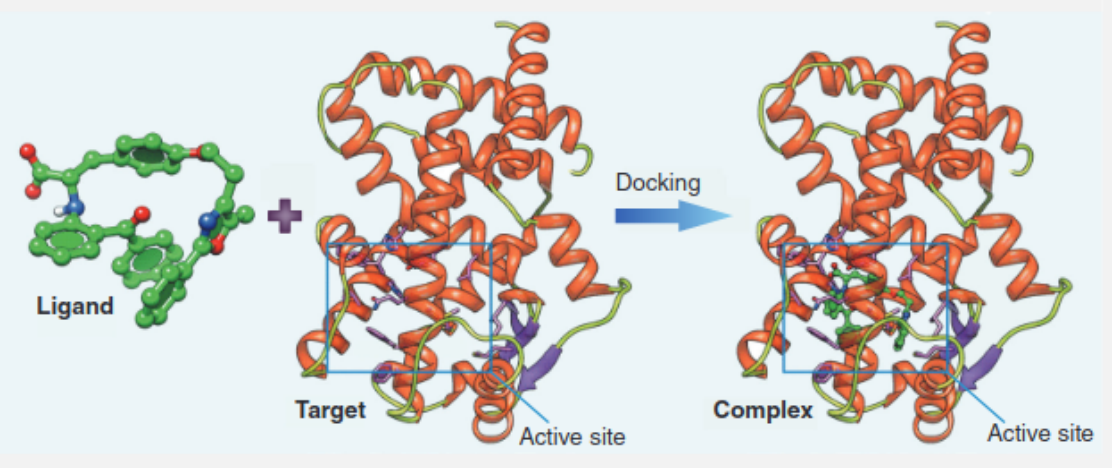


Figure 1.3: A schematic depicting the process of molecular docking. Image source: [6]

1.5 Search Algorithms

Search algorithms are concerned with the task of generating candidate binding poses and ligand conformations at active site. They do so by exploring different positions of ligands using translational and rotational degrees of freedom [7]. Most docking problems treat the ligand as flexible, and the target as rigid, to reduce computational complexity. There are three main types of search algorithms : systematic, stochastic and deterministic. A detailed description of these algorithms is outside the scope of this project.

1.6 Scoring Functions

Scoring functions are mathematical models that predict the binding affinity of a complex. Scoring functions are used in molecular docking to determine the energetics of binding, and therefore predict the most likely binding mode and conformation of the ligand. Scoring functions are also used to rank candidate ligands based on their binding affinity with the target. Therefore, they greatly influence the outcome of the docking and SBVS process, and many researchers consider scoring functions to be the critical factor that limits molecular docking programs [8] [5]. Scoring functions can be classified as conventional or machine learning-based.

1.6.1 Conventional Scoring Functions

There are three main classes of classical scoring functions, and they are briefly described below.

Force-Field

Force-field scoring functions are represented as the sum of energy terms arising from atomic interactions, where each energy term has the functional form of a particular force-field (eg: electrostatic fields, van der Waals force fields). Goldscore is an example of a force-field scoring function [7].

Knowledge-Based

Knowledge-based scoring functions calculate binding affinity by summing energy interaction terms. These interaction terms represent distance-dependent pairwise energy potentials that are statistically determined from large complex data sets.

ParaDocks and FRED are programs that use knowledge-based scoring functions [7] [6].

Empirical

Empirical scoring functions consist of a weighted sum of binding energy terms. These energy terms represent different physical effects that are involved in binding, such as hydrogen bonding, ionic bonding, non-polar interactions etc. Some examples of empirical scoring functions include Glide-Score [7].

1.6.2 Machine Learning Scoring Functions

There are two major shortcomings associated with conventional scoring functions. First, they impose a theory-inspired fixed functional form on the scoring function, which reduces the accuracy in instances where the assumptions underlying the function are not met. Second, most conventional scoring function are linear in nature, which fails to capture the non-linear nature of binding energetics. In contrast, ML models address these shortcomings by directly learning the scoring function from experimental data and capturing binding energetics that are difficult to explicitly model. Their non-parameteric nature also means that no or minimal assumptions are made about functional form of the scoring function. ML models are also able to exploit the growing availability of experimental protein-ligand data better than conventional scoring functions [11] [12].

Many ML scoring functions have been developed in the past, and have been shown to be as or more effective than conventional scoring functions [8] [12]. Some ML scoring functions that have been previously developed are given below: Different

Model	Year	Reference
k Nearest Neighbours	2006	[13]
Neural Network	2010	[14]
Random Forest	2010	[10]
Support Vector Machine	2011	[15]

Table 1.1: Existing ML scoring functions

ML scoring functions have different accuracies. Previous reviews by Ain et al. (2015), Li H. et al. (2020), Yang et al. (2019) have reported on the different ML scoring functions that have been developed at the time of the respective publishing [12] [11] [5]. However, an accurate comparison cannot be conducted in this

manner; the choice of training and test sets can greatly affect model performance, and for these models, the data sets were different or unknown. Ashwaty and Mahapatra (2015) conducted a comparative analysis of six ML scoring functions where all the models were trained and tested on the same sets, to ensure uniformity. This comparison was carried out on data published in 2010 [8]. Since then, a similar comparison has not been carried out to compare multiple ML models. Researchers have compared the performance of novel ML scoring functions against conventional scoring functions, but not compared ML models among themselves [16] [10]. Therefore, this project addresses this gap as a follow up to the work of Ashwaty and Mahapatra (2015), where a more recent, and larger data set was used for comparison. Following the general convention, scoring functions (conventional or ML) are validated on crystal structures of complexes with known binding affinity. This decouples the scoring process from the search algorithm (section 1.5), as co-crystallized ligands (by definition) have no ligand pose generation error [8] [11]. This format comprises of testing under idealised conditions, where the intrinsic properties of each model can be observed.

Chapter 2

Methods

2.1 Data sets

2.1.1 PDBbind

The development and validation of scoring functions requires the availability of high-resolution 3D protein-ligand complex structures and accurately determined binding affinities. The **Protein Data Bank (PDB)** (https://www.rcsb.org/) is an open-access database of the experimentally determined three-dimensional structural data of biomolecules such as proteins and nucleic acids. The structures of these molecules is typically determined by X-ray crystallography, NMR spectroscopy or cryo-electron microscopy [17]. As of April 2023, there are 203,607 structures deposited in the PDB, and it is a widely-used archive of structural information. However, the PDB does not directly identify and annotate protein-ligand complexes among the deposited structures, and structural data of a complex alone is insufficient to develop scoring functions as information of binding energetics is absent. To address this shortcoming, Shaomeng Wang and colleagues developed the **PDBbind database** (http://www.pdbbind.org.cn/) [18]. PDBbind is a comprehensive collection of protein-ligand structures and their binding affinity data. The size of the PDBbind database has been growing; in 2004 there were 1622 complexes and as of 2022, there are 23,396 complexes. It is compiled by identifying and collecting structural data of complexes from PDB, and obtaining binding affinity data from primary literature. Protein-nucleic acid, protein-carbohydrate, protein-oligopeptide and protein-small ligand complexes are present in PDBbind.

This project is restricted to analysis of the protein-small ligand complexes. Hence-

forth the term **ligand** refers to a small molecule with no fewer than six non-hydrogen atoms [18], and **target** refers to a protein against which drugs are being screened. Water molecules, organic solvents, or metal ions are not considered ligands, although they are frequently found in complex with proteins. Furthermore, only three quantities were accepted as binding affinity data: dissociation constants (K_d), inhibition constants (K_i) or concentration at 50% inhibition (IC_{50}). Kinetics parameters such as Michaelis constants (K_m) were not included.

Other databases with binding affinity data such as KiBank [12], CLiBE [19] and BindingDB [20] are also available but have not been used for this project.

2.1.2 PDBbind 2016 Refined Set

PDBbind provides a wealth of information on protein-ligand complexes, but there are inconsistencies in the database and poor-quality complex data is also present. To provide a subset of the PDBbind database consisting of complexes annotated with high quality structural and binding affinity data, Wang and colleagues created the **PDBbind Refined Set**. Complexes from PDBbind that meet the following requirements were included in the Refined Set [18]:

- The resolution of the crystal structures should be 2.5$\mathring{A}$ or better.
- Structures solved by NMR are not included as NMR generates an ensemble of structures [6].
- Only non-covalently bound ligands are included.
- The complex must be of one protein bound to one ligand. Proteins with multiple ligands are not included.
- Only complexes with K_d or K_i values were included. These are equilibrium constants that are independent of protein or ligand concentration. In contrast, IC_{50} values depend on the nature of the binding assay used and are a less reliable measure of binding affinity.
- The ligand should only contains {C,N,O,P,S,F,Cl,Br,I,H }atoms.

The Refined Set is updated every few years. This project utilizes the 2016 PDBbind Refined Set, which consists of 4057 complexes.

2.1.3 CASF 2016 Benchmark

To test software and algorithms, benchmark data sets are often created to enable consistent, reproducible and rigorous comparison. For example, the Community Structure-Activity Resource [21] and Drug and Data Resource (D3R) Grand Challenge [22] are benchmark data sets and exercises to compare molecular docking methods. Motivated to create a similar benchmark dedicated to scoring functions, Renxaio Wang et al. created the **Comparative Assessment of Scoring Functions (CASF) 2016 dataset** [23]. CASF 2016 is compiled from the PDBbind 2016 Refined Set to be a diverse, high-quality test set. The following process was used to build CASF 2016:

- Protein-ligand complexes from the PDBbind 2016 Refined Set were grouped into clusters based on a 90% sequence similarity cutoff.
- Clusters with fewer than five complexes were discarded.
- Among clusters, five complexes were chosen to be representative of the cluster. This included the complex with the highest binding affinity, lowest binding affinity and three complexes with binding affinities between the minimum and maximum. There should be at-least 100-fold difference between the minimum and maximum binding affinities, to provide a sufficient range of values.

These steps ensured that a set of diverse proteins and unique ligands was generated.

2.1.4 Preparation of Files

Data for the complexes and their binding affinities for the PDBbind 2016 Refined Set and CASF 2016 set were downloaded from the PDBbind website (`http://www.pdbbind.org.cn/`). Each complex consisted of a processed protein structure stored in the PDB-file format and the processed ligand structure saved in a mol2 and sdf file format. Hydrogen atoms were already added to both protein and the ligand, and protonation of the protein structure was carried out under neutral pH assumptions.

2.2 Measures of Model Performance

There are four main metrics to evaluate the performance of a scoring function: scoring power, ranking power, docking power and screening power. [23]. Many of

these measures are correlated, but they represent distinct goals for a good scoring function. This project only tests the scoring power and docking power.

Scoring power

Scoring power is the ability of a scoring function to predict binding affinity in linear correlation with experimental binding data. It is quantified with the following descriptors, which are computed between *y_test* (the experimental binding affinities of the test data) and *y_predict* (the binding affinities of the test data as predicted by the models)

- **Pearson's correlation coefficient**: The correlation coefficient tests for linear correlation and hence is the most accurate descriptor of scoring power. A value close to +1 indicates that the model predicts binding affinities in linear correlation to the known binding affinities; a value close to 0 indicates no correlation and a negative value indicates that the predicted binding affinities trend in the opposite direction of the experimental/ true values. This was calculated with *pearsonr* from *scipy.stats* [24].

$$r = \frac{\mathrm{Cov}(Y, \hat{Y})}{\sigma_Y \sigma_{\hat{Y}}}$$

- **Root mean square error (RMSE)**: The RMSE is a positive value that provides an estimation of how well the regression model is able to predict the target value (accuracy).

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{N}}$$

  Where y_i is the experimental value and $\hat{y}_i$ is the predicted value. RMSE was calculated by taking the square root of the value returned by *mean_squared_error* from *sklearn.metrics* [25]. RMSE is low for a good regression model.

- **Coefficient of determination (R2)**: The R2 score of regression is a measure of how well the regression line approximates the experimental data. There are different mathematical definitions of R2, but the following is used in this project:

$$R2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \overline{y})^2}$$

  Where y_i is the experimental value, $\hat{y}_i$ is the predicted value and $\overline{y}$ is the

mean. R2 was calculated with *R2_score* from *sklearn.metrics*. R2 is close to 1 for a good model.

RMSE and R2 do not strictly test for scoring power, but provide information on the general performance of the regression model. It should be noted that a regression model may perform poorly on the basis of metrics like RMSE and R2, but still have high Pearson correlation coefficient i.e high ranking power.

Ranking power

Ranking power refers to the ability of a scoring function to correctly rank known ligands of a protein by their binding affinities when the binding poses of those ligands are given. Ranking power is only computed on ligands that bind (or are expected to bind) to the same target protein. It is quantified with the following descriptors:

- **Spearman's rank correlation coefficient**: Spearman's rank correlation coefficient is a non-parametric measure of the monotonicity of the relationship between *y_test* and *y_predict*. A value close to 1 indicates a strong monotonic relationship. It was computed with *spearmanr* from *scipy.stats* [24].
- **Kendall's rank correlation coefficient**: Kendall's rank correlation coefficient is another non-parametric measure of the ordinal association between *y_test* and *y_predict*. A value close 1 indicates a strong agreement between the two data sets. It was computed with *kendalltau* from *scipy.stats* [24].

Docking power

Docking power is the ability of a scoring function to discriminate between the native pose and conformation of the ligand and decoy/false poses of the ligand [23]. A good scoring function should assign the highest binding affinity to the native pose.

Screening power

Screening power refers to the ability of a scoring function to distinguish between true ligands and known non-binding ligands of the target. A good scoring function should assign a higher binding affinity to true ligands, when compared to false ligands.

2.3 Tests of Model Performance

Using the above performance metrics for scoring and ranking power, the ML scoring functions were put through the following tests:

2.3.1 Scoring power on a general set

In general, it is desirable to have a scoring function that has high scoring power for a wide variety of protein-ligand complexes. For this test, the following training and test sets were used:

- **Primary test set**: The test set is the CASF 2016 benchmark set (section 2.1.3). It is a set of diverse and representative complexes.
- **Primary training set**: The primary test set was generated by removing the complexes present in the CASF 2016 benchmark set from the PDBbind 2016 Refined Set (section 2.1.2).

$$\text{Primary test set } = \text{PDBbind 2016 Refined Set} - \text{CASF 2016 set}$$

$$(n = 3847) = (n = 4056) - (n = 209)$$

The summary statistics of these sets is given below:

Table 2.1: Summary statistics for primary training and test sets

	Primary training set	Primary test set
Size	3847	209
Mean ± SD	6.39 ± 2.00	6.57 ± 2.12
(Min,Max)	(2.00,11.85)	(2.07,11.82)

The primary training set and primary test set are disjoint i.e no protein-ligand complex $P_i - L_i$ is common between the sets. However, some proteins may be repeated between the training and test sets, albeit the proteins are complexed with different ligands. For example, the protein Hsp90-alpha is present is both the training and test sets, but the complex of Hsp90-alpha and 2-methyl-4-diethylamide-phenol is unique to the test set.

2.3.2 Scoring and ranking power on a novel protein

As mentioned previously, some proteins are repeated between the primary training and test set, and this can cause overestimation of model performance. In real drug

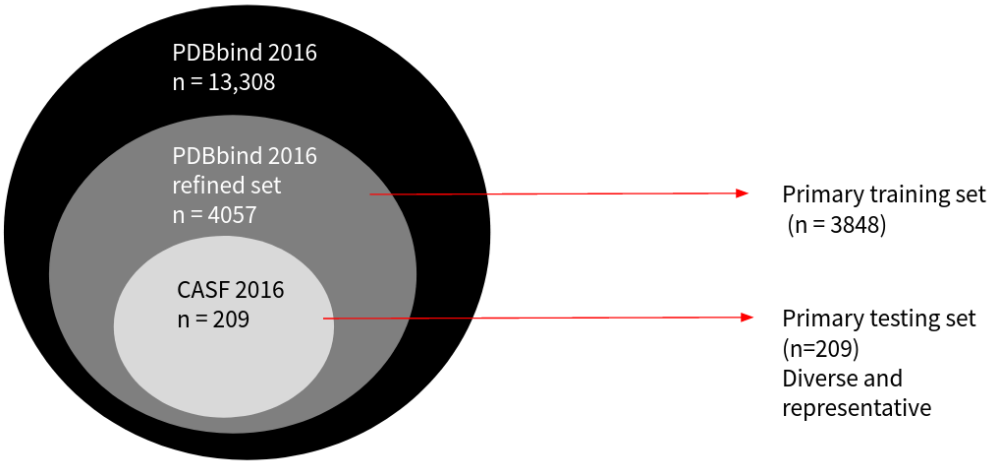


Figure 2.1: A schematic of the data sets used in the project

discovery scenarios, the target protein may not be present in the training set of the ML scoring functions. Therefore it is relevant to assess the performance of ML scoring functions on novel or unseen proteins. This is a more stringent test of the generalizability of the model. For this test, three proteins were selected: **HIV-1 protease, carbonic anhydrase and trypsin**. These were chosen on the basis of their high abundance in the PDBbind 2016 Refined Set. For each protein, a corresponding training and test set was constructed:

- **Test set**: The protein-specific test sets were constructed by extracting all complexes containing the protein from the PDBbind 2016 refined set. HIV-1 protease test set ($n = 280$), carbonic anhydrase test set ($n = 214$), trypsin test set ($n = 59$).
- **Training set**: The training set consisted of the complexes that were left after the protein-specific complexes were removed from the PDBbind 2016 Refined Set. HIV-1 protease training set ($n = 3776$), carbonic anhydrase training set ($n = 3842$), trypsin test set ($n = 3997$).

Index files corresponding to all of these test and training sets were generated by custom python scripts. Both can be found in the appendix.

2.4 Feature Engineering

Feature engineering or extraction the process of using domain knowledge to extract features from raw data. The structure data in a .pdf or .sdf file is not ideal for an ML model to use directly as an input. Instead, there are many different ways to characterise a chemical structure as a set of features which are better suited as inputs for ML models. For this project, the intermolecular interaction features described by Ballester et al. was used [10]. Each feature is an occurrence count x for an m, n atom pair, and is defined as

$$x_{m,n} = \sum_{k=1}^{k=k_m} \sum_{l=1}^{l=l_n} \theta(d_{cutoff} - d_{kl}) \tag{2.1}$$

k_m = number of atoms of atomic number m in the protein
l_n = number of atoms of atomic number n in the ligand
d_{kl} = Euclidean distance between the k^{th} of atom in the protein and l^{th} atom in the ligand
d_{cutoff} = cutoff distance for contact (12 Å)

$$\theta(\text{heavyside function}) = \begin{cases} 1 & d_{kl} \leq d_{cutoff} \\ 0 & d_{kl} > d_{cutoff} \end{cases}$$

For example, $x_{6,7}$ is the number of pairs made by carbon atoms in the protein with nitrogen atoms in the ligand within a 12 $\mathring{A}$ neighbourhood.

Given that only proteins and ligands with atoms {C,N,O,F,P,S,Cl,Br,I} are present in the complexes, a total of $9 \times 9 = 81$ pairs (equivalently features) are possible and are present for each complex. Since the atoms {P,F,Cl,Br,I} are absent in the 20 canonical proteinogenic amino acids, $9 \times 5 = 45$ features will always be zero, and hence have been discarded. Therefore, each complex is characterised by a feature vector $\mathbf{X} \in \mathbb{N}^{36}$.

The binding affinity of complexes (K_d or K_i) span many orders of magnitude, from $10mM$ to $1pM$. Hence, these values are log transformed.

$$\text{Binding affinity } y = -log_{10}K_d \text{ or } -log_{10}K_i \tag{2.2}$$

Therefore, the n^{th} training and testing example can be described by the pair $(\mathbf{X}^{(n)}, y^{(n)})$.

While this feature representation is simple and intuitive, it has several drawbacks. First, the feature vector generated is usually sparse. Second, all atoms of a particular element are treated the same, and hybridisation state or bonded neighbours are not considered. Third, hydrogen atoms are not included, although they are known to contribute significantly to non-covalent forces [10].

The features for a given set were extracted using C programs by Ballester et al. and can be found on http://chemistry.st-andrews.ac.uk/staff/jbom/group/RF-Score.html [10]. Briefly, the program takes an index file and .pdb and .sdf files for the complex as inputs, and returns a .csv file with the feature vectors for each complex. Index files and feature sets used in this project can be found at https://github.com/ira-zibbu/Minor-Project-2013-ML-Scoring-Functions.

2.5 Models

The problem of using the three-dimensional structural data of a complex to predict binding affinity is essentially a regression problem. In this project, five regression ML models were compared. A brief description of each model is provided below. Conventional scoring functions were not compared in this project for two reasons. First, the test sets used to develop conventional scoring functions are not publicly available, which makes it difficult to ensure that the test and training sets are disjoint. Second, scoring functions are usually not available as standalone programs, and are usually only distributed in combination with molecular docking programs.

2.5.1 Multivariate Linear Regression

Linear regression the one of the simplest regression models available. The model is built on the assumption that a linear function $f(\mathbf{X})$ maps the explanatory variable $\mathbf{X}$ to the response variable y.

$$y = f(\mathbf{x}) = \beta_0 + \sum_{i=1}^{P} \beta_i X_i \tag{2.3}$$

The parameters $\{\beta_0, \beta_i..\beta_P\}$ are learned from the training data. Linear models tend to suffer from high bias due to the strict assumption of the linear form of mapping function [8]. This model was chosen to act as a baseline for comparison with the remaining non-parametric and non-linear models. It was implemented using the *linear_model* module from *sklearn* [25].

2.5.2 Multivariate Adaptive Regression Splines (MARS)

MARS is a non-parametric supervised learning model that was created by Jerome Freidman in 1991 [26]. MARS models take following form

$$f(\mathbf{X}) = \sum_{i=1}^{k} c_i B_i(\mathbf{X}) \tag{2.4}$$

The model $f(\mathbf{X})$ is a sum of k basis functions $B_i(\mathbf{X})$, weighted by coefficients c_i. A basis function can take the following forms:

- A constant function p. There is only one constant basis function in the model, and it corresponds to the y intercept.
- A hinge function of the form $\max(0, x - a)$ or $\max(0, a - x)$, where a is a constant and is called the knot or pivot of the hinge function.
- A product of two or more hinge functions.

Therefore, a linear combination of hinge functions can create a piecewise graph, hence the inclusion of the term "splines". The MARS algorithm consists of two steps:

- Forward pass: MARS begins with the intercept p, and proceeds by adding basis functions to the model such that the added function reduces the sum of squares residual error. Possible pivots for the hinge functions are chosen from the values of the explanatory variable available in the training data set. This process of addition continues until the residual error shrinks below a threshold value or a maximum number of terms (passed as an argument to the model) is reached.
- Backward pass: The forward pass leads to a complex and overfit model. At each step of the backward, basis functions that do not make the model effective are pruned until the best submodel is identified.

MARS was implemented with the *Earth* module from the *pyearth* library [27]. MARS generally has low bias due to the innate flexibility of the model and low variance due to the backward pass step.

2.5.3 k Nearest Neighbour (kNN)

kNN is a popular supervised learning algorithm used for both regression and classification tasks. kNN works by computing the Minkowski distance between the

Minkowski distance = $(\sum_{i=1}^{k}(|a_i - b_i|)^q)^{1/q}$ The case $q = 2$ is the Euclidean distance

query example and other training examples, and identifies the k (an integer value given by the user) nearest neighbours. The mean (weighted or unweighted) of the k neighbours is assigned as the label to the query example. Very low values of k lead to overfitting, while very high values of k lead to underfitting. kNN regression was implemented with *KNeighborsRegressor* from *sklearn.neighbors* [25].

2.5.4 Support Vector Machine (SVM) Regression

SVMs were originally developed by Vapnik in 1995 as an algorithm to solve classification problems. Since then SVMs have also been extended to regression problems, and prove to be effective due to their non-linear and non-parametric nature. The goal of SVM classifiers is to find an optimal hyperplane that maximises the margin while minimising misclassification error.

Given that the target scoring function being learned is non-linear, a kernel function is used to map the input space to a higher dimension feature space. Here, the popular radial basis function was chosen. In SVM regression, the objective is to learn a hyperplane $f(\mathbf{X})$ in the feature space that is defined by a vector ω and bias b. Formally:

$$f(\mathbf{X}) =< \omega, K(\mathbf{X}) > +b \tag{2.5}$$

where $K(\mathbf{X})$ is the kernel function and $< \cdot, \cdot >$ is the dot product. Additionally, the model is built such that predictions should not deviate from true values by more than a threshold ϵ. This is achieved by minimising the ϵ insensitive loss function:

$$L_\epsilon = \max(0, |y - f(\mathbf{X})| - \epsilon) \tag{2.6}$$

A non-negative terms ζ_i and ${\zeta_i}^*$ (also known as slack variables) are introduced as a penalty. Therefore, the SVM regression problem can be stated as the following optimisation problem:

$$\min \ ||\omega||^2 + C(\sum_{i=1}^{N}(\zeta_i + {\zeta_i}^*) \tag{2.7}$$

$$\text{subject to } \begin{cases} y_i - < \omega, K(\mathbf{X}_i) > -b & \leq \epsilon + \zeta_i \\ < \omega, K(\mathbf{X}_i) > +b - y_i & \leq \epsilon + {\zeta_i}^* \\ \zeta_i, {\zeta_i}^* \geq 0 \end{cases} \tag{2.8}$$

For two training examples $\mathbf{X}_i, \mathbf{X}_j$, the rbf kernel $K(\mathbf{X}_i - \mathbf{X}_j) = \exp(-\gamma||\mathbf{X}_i, \mathbf{X}_j||)^2$, where γ is a hyperparameter that controls the variance.

Where C is a hyperparameter that controls the tradeoff between model complexity and flatness by scaling the penalty up or down. SVM regression was implemented with the *SVR* module from *skleanr.svm* [25].

2.5.5 Random Forest (RF)

Random forests are a common ensemble supervised ML technique. Ensemble models are composed of base models that are separately trained and then combined in a specific manner to achieve a model that is more robust than a single base model. In random forests, the base models are decision trees [28]. In a decision tree, each non-terminal node is associated with a binary question about a feature element x_i and a threshold t_j of the form $x_i \leq t_j$. If the answer is true, then the left child node is followed, else the right child node is followed. After multiple nodes, the leaf or terminal nodes are reached, each of which represents a disjoint subset $\mathbf{R}_l$ of the input space. Thus the input space is divided into l subsets.

In a regression model decision tree, the target function $f(\mathbf{X})$ is estimated as a piecewise constant function, where each subset of the input space R_l is associated with a constant value $y = c_l$. Formally:

$$y = f(\mathbf{X}) = \sum_l c_l \mathbf{I}(\mathbf{X} \in R_l) \tag{2.9}$$

where

$$\mathbf{I}(\mathbf{X} \in R_l) = \begin{cases} 1 & \mathbf{X} \in R_l \\ 0 & \text{otherwise} \end{cases}$$

The decision tree is built by minimising the sum of square loss function. Random forests combine multiple decision trees. In order to maximise the diversity of trees being combined, bootstrap aggregating or bagging is employed. For a training data set D, bagging creates M data sets of B samples each by uniformly sampling D with replacement, and using these M subsets to train M decision trees. To predict the output for a query sample, the average value of the M trees is taken. In addition to bagging, random forests may use column sampling and row sampling to increase the diversity of trees in the forest. The random forest regressor was implemented with the *RandomForestRegressor* module from *sklearn.ensemble* [25].

2.6 Hyperparameter Optimisation

Hyperparameters are parameters that control the learning process, but are not directly learned from the testing data. Instead, hyperparameters are passed as

arguments to the learning algorithms. Each model (except linear regression) has hyperparameters associated with it, but only a few hyperparameters are important to the outcomes of the given regression problem. The hyperparameters chosen for optimisation were decided on the basis of a previous study [8].

Optimal hyperparameter values can be determined by searching through the chosen parameter space. The parameters for the models of this project were tuned on the primary training set (section 2.3.1). A grid search of the parameter space with a 10-fold cross-validation scheme was carried out, with the R2 score. It was implemented with *GridSearchCV* from *sklearn.model_selection* [25]. Hyperparameters, the parameter space and the optimal values obtained are summarised the tables.

Parameter	Parameter space	Optimal values
max_degree	[1,2..,5]	1
penalty	[0,1,2..,15]	5

Table 2.2: MARS: Hyperparameter values

Parameter	Parameter space	Optimal values
n_neighbours	[1,2..,30]	13
p	[1,2,3,4]	1

Table 2.3: kNN: Hyperparameter values

Parameter	Parameter space	Optimal values
epsilon	[-5,-4,..2]	0.25
gamma	[-5,-4,..2]	0.125
C	[-10,-9,..10]	1

Table 2.4: SVM regression: Hyperparameter values. Parameter space indicated are the exponents for base 2.

2.6.1 Feature Normalisation

The values in the feature vector described in section 2.4 range from 0 a few thousand. Normalising the features to the same range ensures that no single feature

Parameter	Parameter space	Optimal values
mtry	[2,3..,max_features]	2

Table 2.5: Random forest: Hyperparameter values. Max_features is the maximum number of features in the feature vector. In this case it is 36.

can disproportionately affect the outcome of the regression problem. Feature normalisation was carried out with *MixMaxScaler* from *sklearn.preprocessing* [25].

$$X_{\text{normalised}} = \frac{X_i - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}} \tag{2.10}$$

Chapter 3

Results

3.1 Result 1: Scoring power on a general set

As described in section 2.3.1, the ML models were evaluated on a diverse and representative test set as a general test of their scoring power. The five models (section 2.5) were trained on the primary training set ($n = 3847$), and tested on the primary test set ($n = 208$) . Scoring power refers to the ability of a model to predict binding affinity values in a linearly correlated manner with the experimental binding data. This was visually inspected by plotting a scatter plot between *y_predict* and *y_test*. If the data is linearly correlated, we expect the data points to fall on a straight line. The degree of linear correlation was quantified with the Pearson correlation coefficient. Additionally, the root mean square error (RMSE) and coefficient of determination were computed as measures of performance of the regression task.

From Fig 3.1 and Table 3.1, we conclude that in the general test, the random forest has the highest scoring power. The model has the highest Pearson correlation coefficient and R2 score, and lowest RMSE. This confirms the results of a previous study [8]. The SVM regression, kNN and MARS model have an intermediate performance. As expected, the multivariate linear regression model performs the worst. The high bias due to the parametric nature of the model, and the inability to capture non-linear interactions causes poor performance. Although it has a Pearson correlation coefficient higher than the MARS model, it has a lower R2 score. A model may perform the regression task poorly, but still produce linearly correlated predictions for the test examples.

For a good model, we expect the residuals to be symmetrically distributed about

	Pearson correlation coefficient	RMSE	Coefficient of determination
Linear regression	0.544	2.055	0.06
MARS	0.533	1.937	0.166
kNN	0.567	1.77	0.166
SVM regression	0.604	1.754	0.316
Random forest	0.692	1.630	0.407

Table 3.1: Model performance on general test

the line $y = 0$. From Fig 3.2, we observe that this is true for all models except the linear regression model, where the residuals are biased towards the region $y < 0$. This adds further evidence that linear models are unsuitable for use as scoring functions.

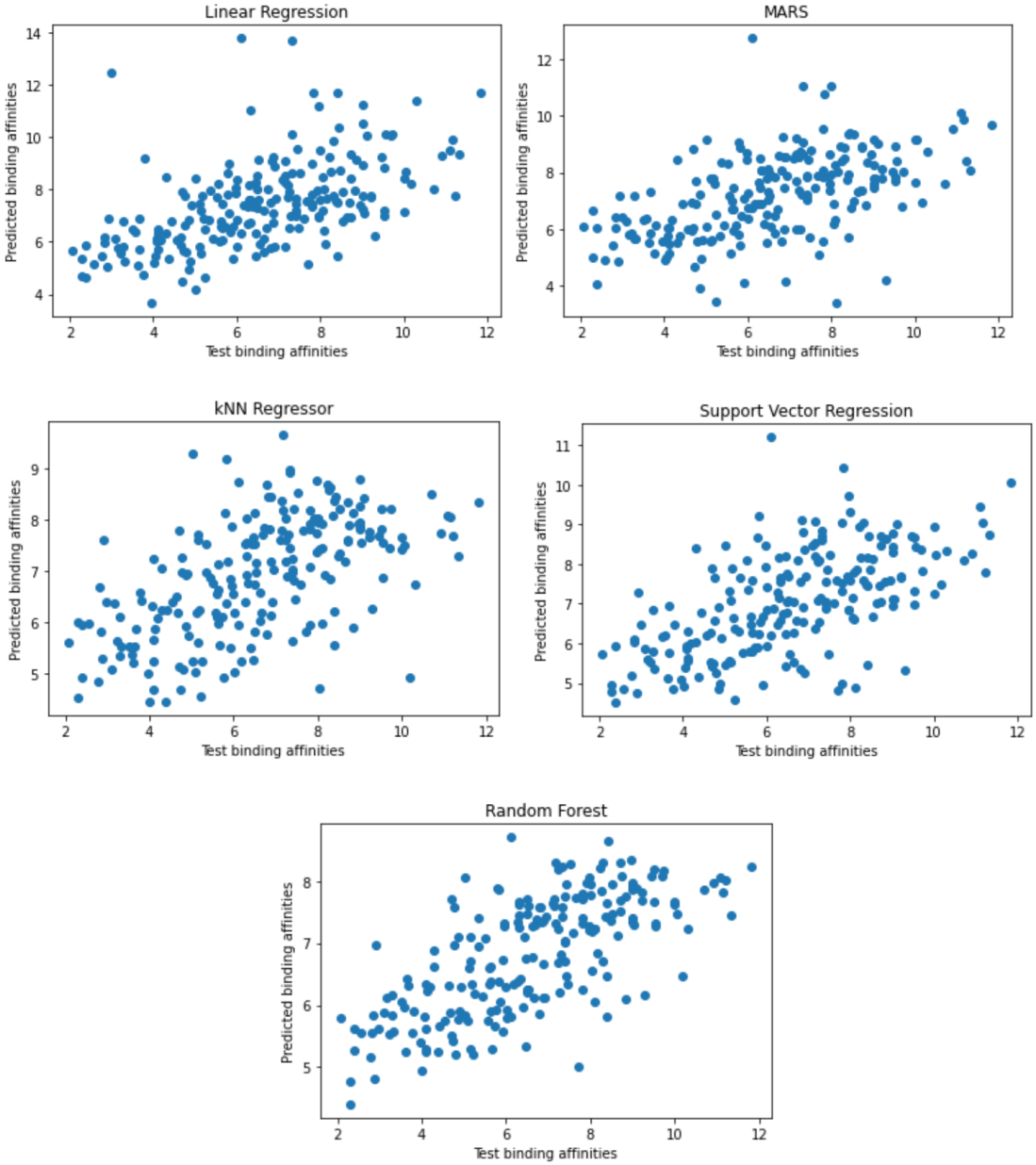


Figure 3.1: Scatter plot of predicted vs experimental binding affinities

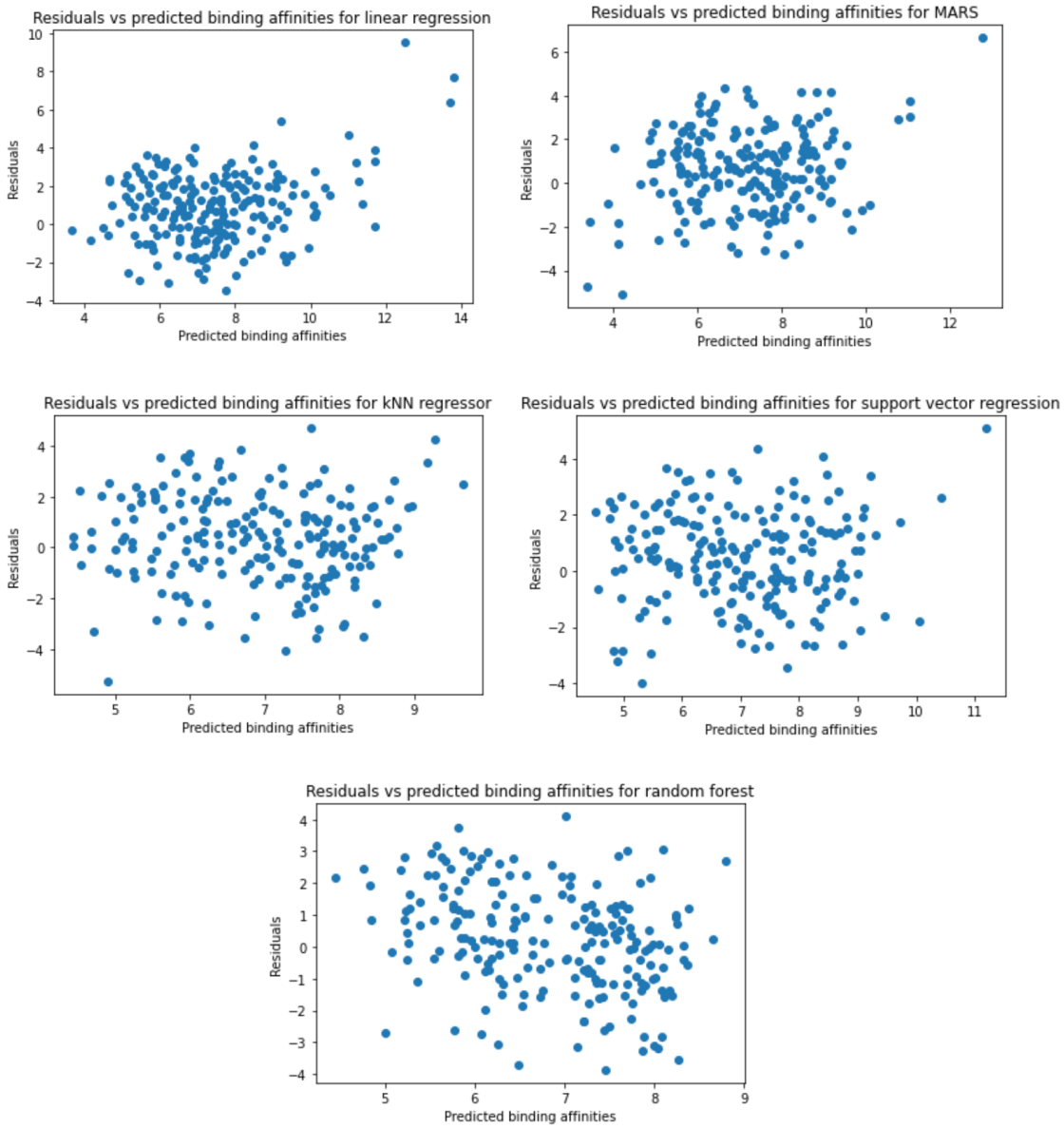


Figure 3.2: Scatter plot of residuals

3.2 Result 2: Scoring and ranking power on a novel protein

The performance of the five ML models on "unseen" proteins that were absent from the test set was investigated as described in section 2.3.2. The scoring power of the model was quantified by the Pearson correlation coefficient, RMSE and R2 score. The ranking power was quantified by the Spearman and Kendall rank correlation coefficients. These results are summarised in the tables 3.2, 3.3, 3.4. The scatter plots of the performance of the random forest model on each test set is also included as representative plots (Fig 3.3, 3.4, 3.5).

3.2.1 HIV protease

On the HIV protease specific test (Table 3.2), the scoring power of all models was very poor, with negative or non-significant Pearson correlation coefficients. Performance on the regression task was also very poor, with high RMSE values and negative R2 scores. The ranking power of the models on this set was very poor, with negative or non-significant rank correlation coefficients (Fig 3.3).

3.2.2 Carbonic Anhydrase

The performance of these models on the carbonic anhydrase test was better than their performance on the HIV protease set (Table 3.3, Fig 3.4). MARS had the highest scoring power with a Pearson correlation coefficient of 0.46, and linear regression had the lowest scoring power, with a Pearson correlation coefficient of 0.29. All of the models performed the regression task poorly with negative R2 scores. This again shows that a model does not need to perform the regression task well to have high scoring power. The rank correlation coefficients were highest for the random forest model, and lowest for the support vector regression.

3.2.3 Trypsin

On the trypsin test, all models had high scoring and ranking power, and the random forest model performs the best with the highest Pearson correlation coefficient and rank correlation coefficients (Table 3.4). The random forest also stands out as the only model with a positive R2 value. Surprisingly, linear regression has ranking and scoring power comparable to the random forest model (Fig 3.5).

	LR	MARS	kNN	SVR	RF
Pearson r_p	0.043 (ns)	-0.0411 (ns)	0.286	0.065 (ns)	-0.036 (ns)
RMSE	1.987	2.276	2.888	2.089	2.370
R2	-0.393	-0.892	-1.945	-0.540	-0.982
Spearman r_s	0.047 (ns)	0.053 (ns)	-0.43 (ns)	0.038 (ns)	-0.243
Kendall τ	0.030 (ns)	0.037 (ns)	-0.236	0.0276 (ns)	-0.153

Table 3.2: Model performance on the HIV protease test. Key: Linear regression (LR), support vector machine regression (SVR), random forest (RF). (ns) indicates non-significant values ($p \leq 0.05$).

	LR	MARS	kNN	SVR	RF
Pearson r_p	0.293	0.461	0.286	0.347	0.314
RMSE	1.808	1.801	1.569	1.729	1.430
R2	-0.469	-0.892	-0.107	-0.344	0.081
Spearman r_s	0.270	0.380	0.232	0.240	0.190
Kendall τ	0.187	0.260	0.160	0.156	0.129

Table 3.3: Model performance on the carbonic anhydrase test. Key: Linear regression (LR), support vector machine regression (SVR), random forest (RF).

	LR	MARS	kNN	SVR	RF
Pearson r_p	0.772	0.618	0.709	0.733	0.807
RMSE	2.340	2.482	1.544	1.722	1.226
R2	-1.883	-2.245	-0.255	-0.562	0.207
Spearman r_s	0.741	0.614	0.654	0.678	0.732
Kendall τ	0.556	0.404	0.475	0.485	0.523

Table 3.4: Model performance on the trypsin test. Key: Linear regression (LR), support vector machine regression (SVR), random forest (RF).

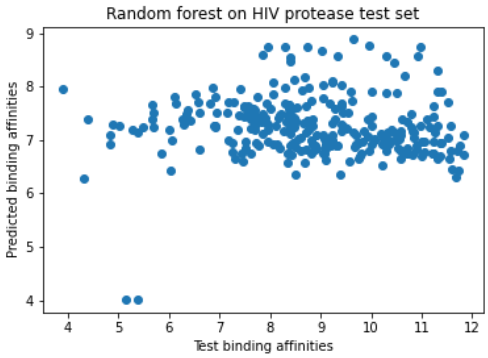


Figure 3.3: Scatter plot of predicted vs experimental binding affinities, as predicted by the random forest model on the HIV-1 protease test.

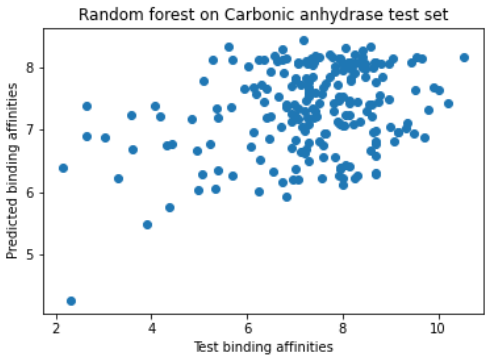


Figure 3.4: Scatter plot of predicted vs experimental binding affinities, as predicted by the random forest model on the carbonic anhydrase test.

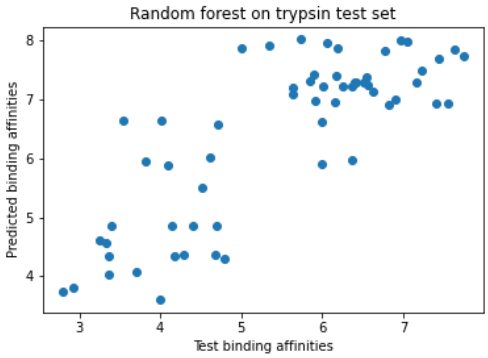


Figure 3.5: Scatter plot of predicted vs experimental binding affinities, as predicted by the random forest model on the trypsin test

3.3 Conclusion

This project was centred on applying ML models to the problem of scoring functions. ML scoring functions present many advantages over conventional scoring functions due to their non-linear and non-parametric nature. The scoring and ranking power of ML scoring functions was assessed on a general set and novel protein test sets. On the general set, the random forest ensemble model had the best ranking power with the highest Pearson correlation coefficient and R2 score, and the lowest RMSE. The linear regression model had the lowest ranking power. Conventional scoring functions were not evaluated in this project, but linear regression represents the general nature of conventional scoring functions. The non-parametric and non-linear models outperformed linear regression, which bolsters the case of using ML scoring functions.

From the protein-specific tests, there was no single model that performed well across all three proteins. No model had good ranking or scoring power on HIV protease test, indicating that there are some proteins that are difficult to predict binding affinities for, possibly due to their biochemistry. A model may have poor ranking power but good scoring power and vice versa. We also observed that model performance drops for all models when proteins similar to the query are absent from the test set. Overall, we see a strong case for the development and use of ML scoring function models in the future.

3.4 Future Directions

There are many limitations of this study, which can be addressed in future work. This project compared the performance of scoring functions on the basis of their scoring and ranking power, but investigating docking and screening power is also important. Docking power can be studied by generating decoy ligand binding poses for a given protein-ligand pair. Screening power can be studied by providing decoy or known non-binding ligands. For example, the DUD-E database is a dataset of property matched decoy ligands thhat can be used for scoring function benchmarking [29]. Only one kind of feature representation was considered in this study, which is associated with a separate set of limitations as discussed in section 2.4. Deep learning methods have been proposed that can automatically learn features from raw data, thus removing the need for explicit feature engineering [5]. For most scoring functions, solvent molecules are not considered and proteins are modelled as rigid bodies, which limits the applicability as protein structure is

dynamic and is affected by the solvent. This study only used 5 regression models, and did not explore deep learning models such as neural networks (NN). NNs have been shown to be effective scoring functions in the past [14] [5]. Due to the utility of molecular docking to both commercial drug discovery campaigns and academic research, scoring functions will improve with continued development and contribute to the ever-growing toolkit of computational techniques.

Appendices

LIST OF PYTHON SCRIPTS

The following python scripts were created to manipulate and generate index files, and build and test models. All code can be found in the accompanying github repository https://github.com/ira-zibbu/Minor-Project-2013-ML-Scoring-Functions

- kNN.py: This module uses the k nearest neighbours algorithm for regression to predict binding affinities for the training dataset
- lin_regress.py: This module uses linear regression to predict binding affinities for the training dataset
- MARS.py: This module uses MARS to predict binding affinities for the training dataset
- Random_Forest.py: This module uses random forest to predict binding affinities for the training dataset
- Support_vector_regress.py: This module uses SVM regression to predict binding affinities for the training set.
- novel_protein_tests: Trains and tests models on the novel protein specific tests.
- processdata.py: Loads the .csv feature sets, drops the zero-value features, normalizes the data
- carbonic_anhydrase_index_generate.py: Obtains the PDB IDs and BA of all carbonic anhydrase complexes from PDBbind 2016 refined set and prints them to a new index file
- conversion.py: Extracts the PDB IDs and the BA from the file PDBbind_2016_refinedset_v1.txt and writes it to another file PDBbind_2016_refinedset_v2.txt
- crosscheck: Verifies that two index files are disjoint and do not have any complexes in common
- get_HIV_protease_index.py: Obtains the PDB IDs and BA of all HIV-1 protease complexes from PDBbind 2016 refined set and prints them to an index file.
- subsetting.py: Creates a new index file by removing complexes of one index file that are present in another index file
- trypsin_index_generate.py: Obtains the PDB IDs and BA of all typsin complexes from PDBbind 2016 refined set and prints them to an index file.

Bibliography

1. Deore, A. B., Dhumane, J. R., Wagh, R. & Sonawane, R. The Stages of Drug Discovery and Development Process. *Asian Journal of Pharmaceutical Research and Development* **7,** 62–67. ISSN: 2320-4850. https://www.ajprd.com/index.php/journal/article/view/616 (Dec. 15, 2019).
2. *Computer-aided drug design* (ed Singh, D. B.) (Springer, Singapore, 2020). 306 pp. ISBN: 9789811568152 9789811568176.
3. Ashburn, T. T. & Thor, K. B. Drug repositioning: identifying and developing new uses for existing drugs. *Nature Reviews Drug Discovery* **3.** Number: 8 Publisher: Nature Publishing Group, 673–683. ISSN: 1474-1784. https://www.nature.com/articles/nrd1468 (Aug. 2004).
4. Hughes, J., Rees, S., Kalindjian, S. & Philpott, K. Principles of early drug discovery: Principles of early drug discovery. *British Journal of Pharmacology* **162,** 1239–1249. ISSN: 00071188. https://onlinelibrary.wiley.com/doi/10.1111/j.1476-5381.2010.01127.x (Mar. 2011).
5. Yang, X., Wang, Y., Byrne, R., Schneider, G. & Yang, S. Concepts of Artificial Intelligence for Computer-Assisted Drug Discovery. *Chemical Reviews* **119,** 10520–10594. ISSN: 0009-2665, 1520-6890. https://pubs.acs.org/doi/10.1021/acs.chemrev.8b00728 (Sept. 25, 2019).
6. Liao, C., Peach, M. L., Yao, R. & Nicklaus, M. C. in *In Silico Drug Discovery and Design* 6–20 (Future Science Ltd, Oct. 2013). https://www.futuremedicine.com/doi/full/10.4155/ebo.13.181.
7. Maia, E. H. B., Assis, L. C., de Oliveira, T. A., da Silva, A. M. & Taranto, A. G. Structure-Based Virtual Screening: From Classical to Artificial Intelligence. *Frontiers in Chemistry* **8,** 343. ISSN: 2296-2646. https://www.frontiersin.org/article/10.3389/fchem.2020.00343/full (Apr. 28, 2020).

8. Ashtawy, H. M. & Mahapatra, N. R. A Comparative Assessment of Predictive Accuracies of Conventional and Machine Learning Scoring Functions for Protein-Ligand Binding Affinity Prediction. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* **12,** 335–347. ISSN: 1545-5963. http://ieeexplore.ieee.org/document/6883187/ (Mar. 1, 2015).

9. Forli, S. *et al.* Computational protein–ligand docking and virtual drug screening with the AutoDock suite. *Nature Protocols* **11.** Number: 5 Publisher: Nature Publishing Group, 905–919. ISSN: 1750-2799. https://www.nature.com/articles/nprot.2016.051 (May 2016).

10. Ballester, P. J. & Mitchell, J. B. O. A machine learning approach to predicting protein–ligand binding affinity with applications to molecular docking. *Bioinformatics* **26,** 1169–1175. ISSN: 1367-4811, 1367-4803. https://academic.oup.com/bioinformatics/article/26/9/1169/199938 (May 1, 2010).

11. Li, H., Sze, K.-H., Lu, G. & Ballester, P. J. Machine-learning scoring functions for structure-based virtual screening. *WIREs Computational Molecular Science* **11.** ISSN: 1759-0876, 1759-0884. https://onlinelibrary.wiley.com/doi/10.1002/wcms.1478 (Jan. 2021).

12. Ain, Q. U., Aleksandrova, A., Roessler, F. D. & Ballester, P. J. Machine-learning SFs to improve structure-based binding affinity prediction and virtual screening. *Wiley Interdisciplinary Reviews: Computational Molecular Science* **5,** 405–424. ISSN: 17590876. https://onlinelibrary.wiley.com/doi/10.1002/wcms.1225 (Nov. 2015).

13. Zhang, S., Golbraikh, A. & Tropsha, A. Development of quantitative structure-binding affinity relationship models based on novel geometrical chemical descriptors of the protein-ligand interfaces. *Journal of Medicinal Chemistry* **49,** 2713–2724. ISSN: 0022-2623 (May 4, 2006).

14. Durrant, J. D. & McCammon, J. A. NNScore: A Neural-Network-Based Scoring Function for the Characterization of ProteinLigand Complexes. *Journal of Chemical Information and Modeling* **50.** Publisher: American Chemical Society, 1865–1871. ISSN: 1549-9596. https://doi.org/10.1021/ci100244v (Oct. 25, 2010).

15. Li, L., Wang, B. & Meroueh, S. O. Support Vector Regression Scoring of Receptor–Ligand Complexes for Rank-Ordering and Virtual Screening of Chemical Libraries. *Journal of Chemical Information and Modeling* **51.** Publisher: American Chemical Society, 2132–2138. ISSN: 1549-9596. https://doi.org/10.1021/ci200078f (Sept. 26, 2011).

16. Wójcikowski, M., Ballester, P. J. & Siedlecki, P. Performance of machine-learning scoring functions in structure-based virtual screening. *Scientific Reports* **7,** 46710. ISSN: 2045-2322. https://www.nature.com/articles/srep46710 (Apr. 25, 2017).

17. Protein Data Bank: the single global archive for 3D macromolecular structure data. *Nucleic Acids Research* **47,** D520–D528. ISSN: 0305-1048. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6324056/ (Database issue Jan. 8, 2019).

18. Wang, R., Fang, X., Lu, Y., Yang, C.-Y. & Wang, S. The PDBbind Database: Methodologies and Updates. *Journal of Medicinal Chemistry* **48.** Publisher: American Chemical Society, 4111–4119. ISSN: 0022-2623. https://doi.org/10.1021/jm048957q (June 1, 2005).

19. Chen, X., Ji, Z. L., Zhi, D. G. & Chen, Y. Z. CLiBE: a database of computed ligand binding energy for ligand–receptor complexes. *Computers & Chemistry* **26,** 661–666. ISSN: 0097-8485. https://www.sciencedirect.com/science/article/pii/S0097848502000505 (Nov. 1, 2002).

20. Liu, T., Lin, Y., Wen, X., Jorissen, R. N. & Gilson, M. K. BindingDB: a web-accessible database of experimentally determined protein-ligand binding affinities. *Nucleic Acids Research* **35,** D198–201. ISSN: 1362-4962 (Database issue Jan. 2007).

21. Carlson, H. A. *et al.* CSAR 2014: A Benchmark Exercise Using Unpublished Data from Pharma. *Journal of Chemical Information and Modeling* **56.** Publisher: American Chemical Society, 1063–1077. ISSN: 1549-9596. https://doi.org/10.1021/acs.jcim.5b00523 (June 27, 2016).

22. Parks, C. D. *et al.* D3R grand challenge 4: blind prediction of protein-ligand poses, affinity rankings, and relative binding free energies. *Journal of Computer-Aided Molecular Design* **34,** 99–119. ISSN: 1573-4951 (Feb. 2020).

23. Su, M. *et al.* Comparative Assessment of Scoring Functions: The CASF-2016 Update. *Journal of Chemical Information and Modeling* **59,** 895–913. ISSN: 1549-9596, 1549-960X. https://pubs.acs.org/doi/10.1021/acs.jcim.8b00545 (Feb. 25, 2019).

24. Virtanen, P. *et al.* SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* **17,** 261–272 (2020).

25. Pedregosa, F. *et al.* Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* **12,** 2825–2830 (2011).

26. Friedman, J. H. Multivariate Adaptive Regression Splines. *The Annals of Statistics* **19.** ISSN: 0090-5364. https://projecteuclid.org/journals/annals-of-statistics/volume-19/issue-1/Multivariate-Adaptive-Regression-Splines/10.1214/aos/1176347963.full (Mar. 1, 1991).

27. *Python implementation of Jerome Friedman's Multivariate Adaptive Regression Splines algorithm* https://github.com/scikit-learn-contrib/py-earth.

28. Jiang, H. *Machine learning fundamentals: a concise introduction* ISBN: 9781108837040 9781108940023 (Cambridge University Press, United Kingdom ; New York, NY, 2021).

29. Mysinger, M. M., Carchia, M., Irwin, J. J. & Shoichet, B. K. Directory of Useful Decoys, Enhanced (DUD-E): Better Ligands and Decoys for Better Benchmarking. *Journal of Medicinal Chemistry* **55.** Publisher: American Chemical Society, 6582–6594. ISSN: 0022-2623. https://doi.org/10.1021/jm300687e (July 26, 2012).