UNIVERSITAT DE BARCELONA

FUNDAMENTAL PRINCIPLES OF DATA SCIENCE MASTER'S THESIS

Applying AutoML techniques in drug discovery: systematic modelling of antimicrobial drug activity on a wide spectrum of pathogens

Author:
Marcos DE LA TORRE

Supervisors:
Dr. Miquel DURAN-FRIGOLA
Dr. Jordi VITRIÀ

A thesis submitted in partial fulfillment of the requirements for the degree of MSc in Fundamental Principles of Data Science

in the

Facultat de Matemàtiques i Informàtica

January 15, 2023

UNIVERSITAT DE BARCELONA

Abstract

Facultat de Matemàtiques i Informàtica

MSc

Applying AutoML techniques in drug discovery: systematic modelling of antimicrobial drug activity on a wide spectrum of pathogens

by Marcos DE LA TORRE

Predictive modelling of antimicrobial activity of molecules is a crucial step towards the discovery of anti-infective medicines. Unfortunately, there is a shortage of models covering endemic pathogens of the Global South, reflecting the existing bias in research towards diseases prevalent in wealthy countries.

This project has developed a pipeline to systematically build drug discovery models, in particular antimicrobial activity prediction models for small molecule compounds. The data of assay results on a selected pathogen is extracted from a publicly available database: ChEMBL. This data is then cleaned and processed in order to build predictive models with various Automated Machine Learning (AutoML) techniques using the ZairaChem tool from the Ersilia Open Data Initiative.

The pipeline has been applied on 6 pathogens of great relevance to global health known as ESKAPE, for which the data has been obtained and processed, and baseline models created. We have built the full set of final models for one of these pathogens, *Staphylococcus aureus*. The pipeline can be used on any other pathogen for which ChEMBL has sufficient data. This pipeline will be used to deploy models in the Ersilia Model Hub, a repository of pre-trained ML for drug discovery in global health. This will be an opportunity to compensate for the shortage of ML models adapted to the needs of the Global South.

Acknowledgements

First of all, I am extremely grateful to my supervisor in Ersilia, Dr Miquel Duran-Frigola. He is the manager everyone would like to have. It has been a great collaboration and an excellent learning experience for me.

Many thanks to my supervisor in the UB, Dr Jordi Vitrià, for the guidance, ideas, and the reassurance that we were moving in the right direction.

My sincere thanks as well to Dr Gemma Turon for her expert answers to my questions and for many relevant suggestions, comments and remarks.

Finally, I would like to thank my family, partner and friends for being there and for their kind support during this period.

Contents

Chapter 1

Introduction

1.1 Motivation

Predictive modelling of antimicrobial activity of molecules is a crucial step towards the discovery of anti-infective medicines. Unfortunately, there is a shortage of models covering endemic pathogens of the Global South, reflecting the existing bias in research towards diseases prevalent in wealthy countries.

In particular, there is a shortage of models for the six pathogens known collectively as *ESKAPE*. These are:

- *Enterococcus faecium*
- *Staphylococcus aureus*
- *Klebsiella pneumoniae*
- *Acinetobacter baumannii*
- *Pseudomonas aeruginosa*
- *Enterobacter* (various species)

The study (Ikuta et al., 2022) estimated the number of global deaths caused by 33 of the most deadly bacterial pathogens (figure 1.1), with a disproportionally high rate of the years of life lost occurring in Sub-Saharan Africa. The six ESCAPE pathogens, which have been selected as proof of concept for our model building pipeline, are among the top in this list.

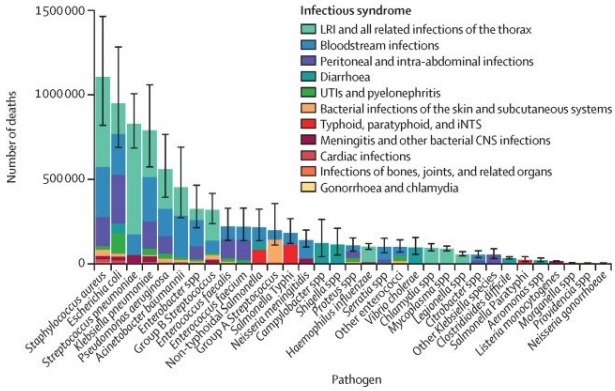


FIGURE 1.1: Global number of deaths by pathogen and infectious syndrome, 2019. Chart from (Ikuta et al., 2022)

1.2 Technical problem statement

Overall, the goal is to compile bioactivity data from public repositories and build machine learning models from it.

The information on the result of experiments on compounds activity against pathogens are scattered in scientific papers and other sources. Fortunately, the ChEMBL project has done an enormous amount of work to compile and harmonize this data. Still, before we can use it to train a predictive model, the data needs preparations and some decisions need to be made.

The idea is to be able to build models in a systematic way, so that the same pipeline can be applied to any pathogen (having sufficient data in the ChEMBL database) to gather the data, transform it, use it to train the models, and evaluate them. To build models without the need of manual intervention, AutoML techniques can be used, such as those provided by the ZairaChem tool from the Ersilia Open Data Initiative.

The pipeline will be applied to build models for the six ESKAPE pathogens.

1.3 Previous work

There is an acceptable amount of compound bioactivity prediction models for some well-studied pathogens, such as *Plasmodium falciparum* (malaria) and *Mycobacterium tuberculosis* (tuberculosis). Unfortunately, models for ESKAPE pathogens are scarce due to low interest by pharmaceutical companies.

This project is a practical application of previously developed methods and tools. In particular, we have used the ZairaChem tool to build the models, that in turn relies on a number of existing chemical descriptor models and AutoML frameworks. Zairachem is described in detail in section 2.4.

1.4 Ersilia Open Data Initiative

Ersilia Open Data Initiative (www.ersilia.io) is a tech non-profit organization supporting research against infectious and neglected diseases in low-income countries.

The Ersilia Model Hub is a repository of pre-trained Machine Learning models for infectious and neglected disease research. It aims at providing open source AI/ML tools to accelerate drug discovery in global health, with a particular focus on the needs of the Global South. The repository is populated with models from the scientific literature (with appropriate acknowledgement) as well as models developed by the Ersilia team or contributors.

Chapter 2

Background info and theory

2.1 Automated Machine Learning (AutoML)

Building a Machine Learning model usually requires to make a number of decisions, mainly in data preprocessing, the choice of the ML algorithm, and choice of hyperparameters. An Automated Machine Learning (AutoML) framework automates some or all of these decisions.

AutoML is currently a very popular trend, and the number of available frameworks is growing quickly. Here we mention the ones that Zairachem uses at the moment.

- **FLAML** (Wang et al., 2021) is used in the majority of the Zairachem submodels, fitting Random Forest and XGBoost models. Its main differentiating feature is high speed.
- **AutoGluon-tabular** (Erickson et al., 2020), which Zairachem applies on top of its 12-variable embedding, builds sophisticated ensemble models in multiple layers. It has ranked very high in benchmarks against other AutoML frameworks.
- **KerasTuner** (O'Malley et al., 2019) is a framework that finds optimal hyperparameters for a neural network. Zairachem uses it on top of the Grover model in Zairachem. This makes sense because Grover is a pre-trained neural network, and with KerasTuner we can add one or more extra layers that get trained on our specific data.
- In addition, a **Convolutional Neural Network** (CNN) is used in one of the submodels, to make use of the Molmap descriptors. Although a CNN is not properly AutoML, in this case it plays an analogous role, since it is used with the default configuration and has been shown to perform well without the need of hyperparameter tuning (Shen et al., 2021).

It is important to remark that Zairachem is modular and designed to incorporate any additional descriptors and AutoML frameworks in order to further improve performance.

2.2 Drug Discovery

The research and development of new drugs (Drug Discovery) is expensive and time-consuming. Over 90% of drug candidates fail during the clinical trial phases (Namba-Nzanguim et al., 2022). Because of this, methods that can successfully select the most promising candidates are very valuable. In particular, computer-based simulations or models (called *in silico* methods), are an effective tool.

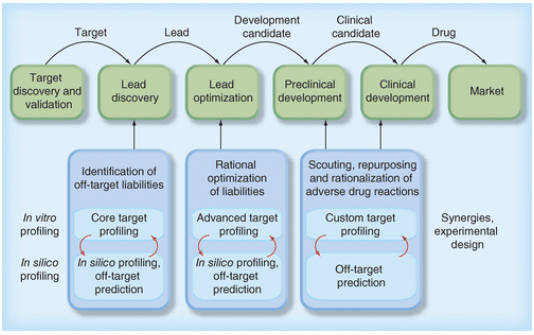


FIGURE 2.1: Example Drug Discovery pipeline - Image from Schmidt et al., 2014

The advances in Artificial Intelligence and Machine Learning in recent years is leading to important improvements in data-driven drug discovery. Machine Learning allows to build models with higher predictive power than traditional statistical models, and they often offer results also with limited amounts of data. In particular, Machine Learning models are being applied successfully to perform screening of drug candidates, reducing the cost of the process.

Figure 2.1 shows an overview of a Drug Discovery process that includes *in silico* profiling.

2.2.1 Assay types

In order to determine if a compound is effective against a pathogen, there are different assay strategies. Some of the most common ones are described here. For each of these four, our pipeline will build specific prediction models if sufficient data is available.

- **Minimum Inhibitory Concentration (MIC)**

  A MIC assay determines the lowest concentration of a compound that completely inhibits visible growth of a pathogen over a defined time period. It is usually determined by exposing the pathogen to an incremental series of the compound concentrations. Usually measured in $\mu g/mL$ or in nM.

- **Inhibition Zone (IZ)**

  An IZ assay consists on growing a culture of the pathogen on a plate, and then adding a small amount of the compound to the plate in the form of a disc. The area around the disc that is free of growth is called the inhibition zone. The larger the inhibition zone, the more effective the antimicrobial is at inhibiting the growth of the pathogen. Usually measured in mm.

- **Inhibition**

  An Inhibition assay is performed by adding the compound to a solution containing the pathogen and measuring the remaining activity of the pathogen after a certain amount of time. The resulting percentage of inhibition is the comparison between the activity before and after adding the compound. Measured in %.

- **Half-maximal Inhibitory Concentration (IC_{50})**

An IC_{50} assay tries to determine the concentration of the compound required to inhibit 50% of the growth of the pathogen. This is done by trying different concentration values, observing the response, and then fitting a curve to find the 50% point. Usually measured in $\mu g/mL$ or in nM.

2.2.2 Machine Learning in Drug Discovery

The majority of Machine Learning tasks in Drug Discovery are:

- Supervised (classification or regression), as in our case
- Unsupervised: Typically clustering
- Generative: Used to invent new candidate molecules

Our project will deal with supervised models to predict molecular bioactivity. This kind of models are a type of quantitative structure-activity relationship (QSAR) modeling.

2.3 ChEMBL Database

2.3.1 What is ChEMBL

ChEMBL ((Mendez et al., 2018)) is *a large, open-access bioactivity database*. It compiles information about small molecules and their biological activity. The information is obtained from published journal articles, data on approved and candidate drugs, from other databases such as PubChem, BioAssay and BindingDB, as well as deposited data from other institutions. The data is maintained manually by curators.

In this project we use ChEMBL release 31, from July 2022. It contains information from over 85000 documents, totaling more than 19 million measurements for over 2.3 million compounds (source: ChEMBL31 release notes).

2.3.2 Available data on important pathogens

We have queried the ChEMBL database to quantify the number of assay results (*activities*) that are available for the 20 pathogens that cause the most deaths, according to the chart shown in 1.1. We can see that Chembl contains a rich amount of data for most of these pathogens.

Pathogen	Data points in ChEMBL
Staphylococcus aureus	132,317
Escherichia coli	187,614
Streptococcus pneumoniae	20,104
Klebsiella pneumoniae	31,464
Pseudomonas aeruginosa	99,448
Acinetobacter baumannii	32,064
Enterobacter spp	8,257
group b *Streptococcus*	~1,200
Enterococcus faecalis	20,580
Enterococcus faecium	8,288
non-typhoidal *Salmonella*	~12,000
group a *Streptococcus*	~20,000
Salmonella typhi	~4,700
Neisseria meningitidis	531
Campylobacter spp	1,450
Shigella spp	3,799
Proteus spp	9,079
Haemophilus influenzae	6,701
Serratia spp	3,103

The central entity in the Chembl database is the assay. The main properties of an assay that are relevant to this project are the following: compound, assay type, measurement units, value...

2.3.3 Target Organism vs Protein

In ChEMBL, all assays are associated to a target. The target can be a complete organism, a specific protein, or others (such as tissues or cell lines).

For our models, we will only consider organism and protein targets. Some models will be built to predict activity against organisms (the entire pathogen), other models on activity against specific proteins.

In any case, we are primarily interested on assays targeting organisms. Protein targets are interesting as a complement, but we have data for only a few proteins out of the thousands of proteins in an organism.

2.3.4 SMILES data

The Simplified Molecular-Input Line-Entry System (SMILES) is a format to describe chemical formulas of compounds using short ASCII strings. It is widely used in computational Chemistry. Many Chemical databases (such as ChEMBL) and Machine Learning models utilize SMILES.

In this project, the models receive their input in the form of SMILES strings.

2.4 ZairaChem tool

ZairaChem is a machine learning (ML) tool to build small molecule activity prediction models. It is fully automated, not requiring manual choices of algorithms or hyperparameters. It is meant to work with limited computing resources.

See Turon et al., 2022 for a more detailed description of the tool and a success story using ZairaChem in a real setting in South Africa.

ZairaChem takes as training data a set of molecules in SMILES format together with an indicator of activity (1-0 binary variable).

In order to build models that predict the activity, we need to obtain numerical properties (*descriptors*) based on the SMILES text string. ZairaChem uses various existing methods for this purpose. The idea is that combining descriptors of very different nature, the models can adapt to a variety of tasks. The following descriptors are currently used by ZairaChem:

- Classic descriptors: Contain basic physico-chemical properties
- Classic fingerprint: 2048 binary variables encoding atomic connectivity (atoms and bonds between them)
- Chemical Checker (Duran-Frigola et al., 2020): Known and inferred bioactivity. A vector of 3200 components.
- ECFP fingerprint: 2D structural fingerprints (*morgan counts*). A vector of 2048 components. Similar concept to the Classic fingerprint but counts instead of binary
- Mordred (Moriwaki et al., 2018): >1600 physicochemical parameters
- Grover embedding: graph-based embeddings. 5000 dimensions
- Molmap (Shen et al., 2021): Two 2D images built with descriptors, where pixels of the image that are close to each other correspond to correlated descriptors

Once the descriptors are generated, the submodels will be trained, each using one of the previous descriptors and an appropriate Machine Learning algorithm. By default ZairaChem trains 9 submodels. The following table lists the submodels, what descriptors they are based on, and the Machine Learning Algorithm that builds them (typically an AutoML framework - see section 2.1).

Submodel	**Descriptor**	**ML Algorithm**
classic	Classic descriptors	FLAML
fingerprint	Classic fingerprint	FLAML
cc-signaturizer	Chemical Checker	FLAML
morgan-counts	ECFP fingerprint	FLAML
mordred	Mordred	FLAML
grover-embedding	Grover embedding	FLAML
manifolds	Grover embedding	Manifold + AutoGluon
reference-embedding	Grover embedding	Keras Tuner
molmap	Molmap	CNN

At prediction time, each of these submodels will give a binary classification score, and the 9 results are aggregated into a single score.

Chapter 4 provides more detail about the ZairaChem pipeline.

Chapter 3

Methodology and data

3.1 Data extraction from ChEMBL Database

We have developed two python modules to carry out the data extraction from the ChEMBL Database into the datasets required for the models: the function `chembl_activity_target` and the class `ChemblMoleculeSampler`.

3.1.1 Function to extract required fields from ChEMBL

A python function (`chembl_activity_target`) has been created to extract from the ChEMBL database a list of assay results (called "activities" in the database) for a given pathogen. The extraction includes a number of useful variables from different tables, such as:

- Activity results: molecule used, value, units, etc.
- Assay information: type, publication, etc.
- Target information: organism, protein

This information is deduplicated and standardized into a clean data set.

The criterion to select a pathogen is based on searching for a given text in the organism's name. If a more sophisticated selection is required, a practical approach is to use this function with a more generic value, and then filter the resulting dataset as desired.

3.1.2 Class to obtain negative cases

It may happen that a data extraction of assay results for a given pathogen has a majority of positive results. This is mostly due to the existing positive reporting bias in biomedicine, where positive (active) results are more likely to be published than negative results.

In this case, completing the data with a sample of negative cases may help to build a better model. A possible approach for this is to select randomly some molecules used in other assays in ChEMBL, making sure that we exclude those that are known to be positive. While these molecules are not 100% guaranteed to be negative, the probability that a random molecule is active against our pathogen just by coincidence is extremely low (certainly below 1%, much less for ESKAPE pathogens). This approach makes sense because ChEMBL is a good representative of the chemistry space used in Medicine.

This procedure is implemented in the class `ChemblMoleculeSampler`. It will generate a random sample of molecules of the required size, making sure that they do not belong to a given set of known active molecules.

3.2 Regression or classification?

The results of an experiment as provided in a publication (and therefore in the ChEMBL database) is usually a continuous value (examples: $1000nM$, $0.5\mu g/ml$, 80%) Because of this, we would be tempted to use a regression model for it. However, there are some reasons to prefer a classification model:

- Some of the input data is very skewed.
- Often the input data capped to a certain value (e.g., $\geq 1000nM$).
- Most important of all: the drug development processes work as a pipeline with several steps. Each drug candidate may pass a step or not pass it. This means that the user is not interested in the exact measurement value, rather in the likelihood that the molecule is considered active.

The continuous values will be converted into a binary result ("active" or "not active"), based on a given threshold. The threshold will be different depending on the assay type and the measurement units.

In a brainstorming session with subject matter experts from Ersilia, we also decided to build two versions of every model, with two different thresholds. One of the thresholds (high confidence) will be more demanding, and the other (low confidence) more relaxed. The scientist who uses the models will choose one or the other based on the required task, or they may use some kind of combination of the results of the two models.

3.3 Configuring activity threshold values

The threshold values to define what results are active or not are provided in the configuration table, which the pipeline will use when building the datasets for training. This configuration table defines, for each combination of **assay type** and **measurement units**:

- active_direction: +1 or -1
- threshold low confidence
- threshold high confidence

An active direction value of +1 indicates that a value higher than the threshold will be active. If the active direction is -1, a value lower than the threshold will be active. This depends on the type of experiment and the units used, and it cannot be deduced from the database values.

We have provided a configuration table that covers the most common combinations of assay type and measurement units used in the ChEMBL data corresponding to the six ESKAPE pathogens. If the process is applied to other pathogens, it is possible that the table needs to be expanded to cover new assay types and/or units. The assay results data that does not match the configuration file will not be included in the datasets for training the models. If this is the case for a large amount of data, a warning will be issued.

Our current version of the configuration table is the result of an iterative process working together with subject matter experts from Ersilia. Based on the statistics of

TABLE 3.1: Configuration table (extract)

standard type	standard units	active direction	low cut	high cut
MIC	ug.mL-1	-1	2.5	0.5
IZ	mm	1	15	20
MIC	nM	-1	5000	1000
Inhibition	%	1	50	80
IC50	nM	-1	5000	250

the data and the knowledge of the usual acceptance criteria in each kind of experiment, after a few iterations, the current version is considered useful for modeling. It is likely we will want to refine some of the thresholds after we have more experience using this modeling process. Besides, our code is open source and any other user may want to customize this configuration.

In some assays, the exact measurement result is not provided and instead a qualitative explanation is given in the text. The ChEMBL database, in this case, may store the text "Active" or "Not Active" in the activity comment field (a free text field), and the numeric value will be missing. Our process also includes these cases and records them active or not as indicated by ChEMBL.

Some additional data cleaning is done to standardize some values and remove possible invalid values. In the end, we obtain a clean dataset with the molecule representation (smiles) and the activity indicator (0 or 1).

For illustration, table 3.1 contains the 5 most frequent combinations of assay type (`standard_type`) and measurement units (`standard_units`), with the selected thresholds for low confidence (`low_cut`) and high confidence (`high_cut`).

- MIC ($\mu g/mL$), figure 3.1
- IZ (mm), figure 3.2,
- MIC (nM), figure 3.3
- Inhibition, figure (%) 3.4
- IC50 (nM), figure 3.5

3.4 Prediction tasks

We will build several models for each pathogen, so they can be useful for different *tasks*. Initially we want to build a model for a pathogen of interest that predicts if a given molecule is active against the pathogen. This may mean to kill it, but also to inhibit its growth or make it less harmful. We call this initial generic task *organism-anytype*. The model for this task will use all the assays that are marked as targeting the organism (pathogen).

It may also be useful to define tasks for the most common types of assay. For example, the task *organism-mic* aims to predict the activity of a molecule against the pathogen when used in an assay of type MIC. If a researcher wants to perform a MIC experiment and wishes to obtain promising candidates for it, a model based on this task may be more helpful than the generic model.

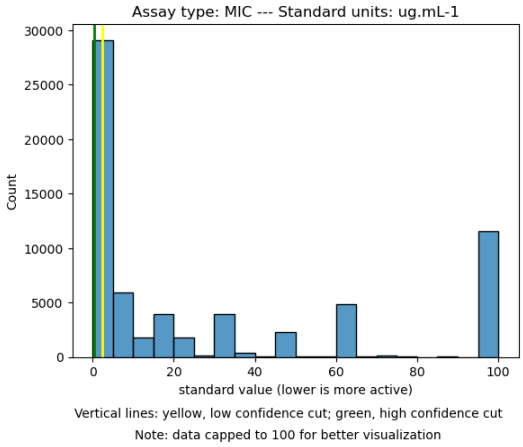


FIGURE 3.1: Value distribution for MIC ($\mu g/mL$)

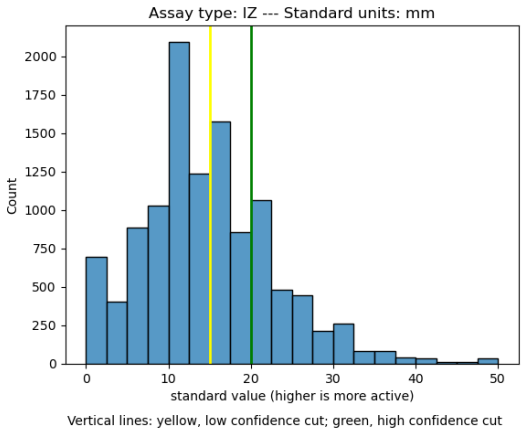


FIGURE 3.2: Value distribution for IZ (mm)

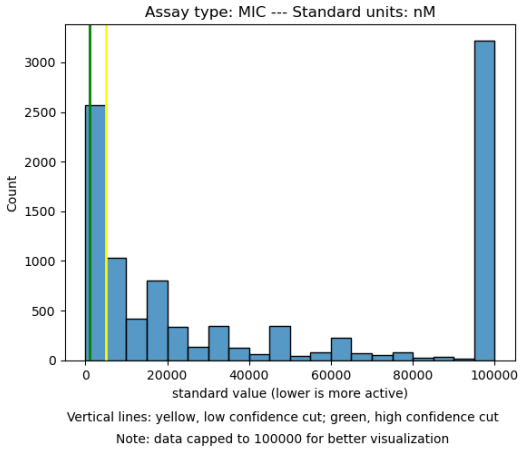


FIGURE 3.3: Value distribution for MIC (nM)

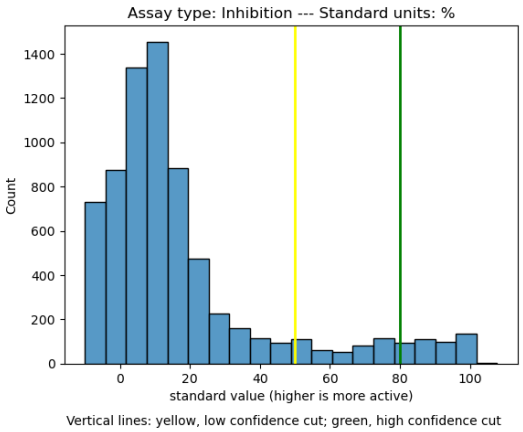


FIGURE 3.4: Value distribution for IZ (mm)

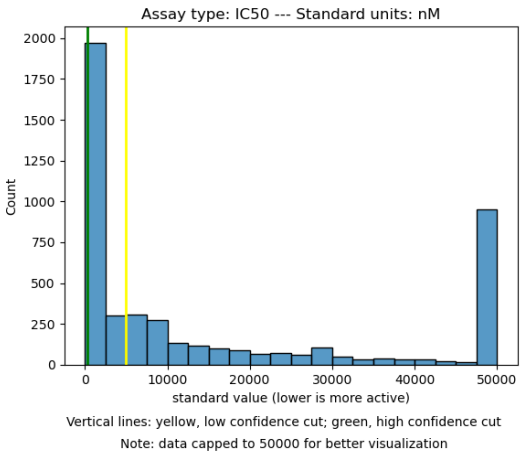


FIGURE 3.5: Value distribution for IC50 (nM)

Furthermore, sometimes there are very large-scale assays that try thousands of molecules. It is possible to build a model just on the data from that assay, which may be very precise if a researcher decides to perform a very similar experiment with other molecules. Because of this, we also define a task for each of the largest assays on the selected pathogen.

In addition to assays targeting an organism, ChEMBL also classifies other assays as targeting a specific protein of the pathogen. These are much more focused assays, and they aim at obtaining very particular effects. We will define a task for each of the target proteins for which the database contains a minimum amount of data.

For each organism and task, two models will be generated (high and low confidence) if the amount of data is considered sufficient. For this there is a configurable parameter with the required minimum number of positive (active) cases in the dataset. Currently the limit is set at 30 positive cases, which seems to be a bare minimum based on the experience implementing Zairachem in a real setting (Turon et al., 2022).

In the end, for a given organism we may have between 4 and 28 models depending on how much data ChEMBL has on that organism.

Table 3.2 lists all the tasks for which models may be built.

Given a pathogen, the program `create_datasets` automatically creates a dataset for each of the models that will be trained for that pathogen. The program takes care of the following:

1. Obtain from the ChEMBL database all the assay results for the required pathogen
2. Merge with the configuration table to obtain active direction and low/high confidence activity thresholds
3. Calculate activity value (target variable for the models)
4. For each of the tasks indicated in 3.2:
   (a) Filter data as required for the task (by target type, assay type, individual assay, individual protein)
   (b) Fill up with sampled negative cases if less than 50%
   (c) Store the resulting datasets if minimum size requirements are met

In the end, a total of 72 datasets were generated for the 6 ESKAPE pathogens. See the full list in appendix B.

3.5 Split into training and test data

Each dataset will be split into a training and a test set. The test set (about 20%) will be used to obtain model performance statistics that indicate how good the model is.

An alternative strategy would have been k-fold cross validation. This would offer a more robust measurement of the model performance. However, the training of these models is costly (see table 4.1), and running for instance a 5-fold cross validation would take 5 times longer and become intractable within the timeframe of this project.

The most usual approach to the train-test split is to perform a random sampling of the data. In our case, however, this does not offer a realistic assessment of the model performance.

TABLE 3.2: Tasks

Task	Description
organism_anytype	model trained using assays that target the pathogen
organism_mic	model trained using assays of type MIC that target the pathogen
organism_iz	model trained using assays of type IZ that target the pathogen
organism_inhibition	model trained using assays of type Inhibition that target the pathogen
organism_ic50	model trained using assays of type IC50 that target the pathogen
organism_assay_top1	model trained with one specific assay (top 1 amount of data)
organism_assay_top2	model trained with one specific assay (top 2 amount of data)
organism_assay_top3	model trained with one specific assay (top 3 amount of data)
protein_any	model trained using assays that target any protein
protein_top1	model trained with one specific protein (top 1 amount of data)
protein_top2	model trained with one specific protein (top 2 amount of data)
protein_top3	model trained with one specific protein (top 3 amount of data)
protein_top4	model trained with one specific protein (top 4 amount of data)
protein_top5	model trained with one specific protein (top 5 amount of data)

The historical data that we have of the assays performed usually cover a small subset of all the chemical space of possible molecules. Drug design is an iterative process based on making modifications to promising new compounds. Therefore, a random split does not necessarily simulate a realistic prospective scenario.

When we try to predict the behavior of a molecule that is similar to existing molecules in the training dataset, the model will typically perform well, as this task is easy (comparable to interpolation). On the other hand, predicting a molecule that is very different to the molecules used for training the model is difficult (comparable to an extrapolation).

We want the test dataset to emulate the situation that we will find when the model is used on real cases that may be quite different from the training cases. ZairaChem helps with this by offering a command to generate train-test splits with different criteria. We will try three criteria:

- Random
- Scaffold
- Similarity

The random split criterion just selects randomly 20% of the data for the test set. This will evaluate the performance when predicting molecules that are similar to the ones we already know.

The scaffold split criterion classifies each of the molecules into groups based on their *scaffold*, which can be thought of as the core of the molecule. Molecules with the same scaffold tend to have properties in common. This split method ensures that the test set contains different scaffolds than the training set. This emulates the situation of trying to predict molecules that are genuinely novel. The performance metrics obtained with this split are expected to be worse.

The similarity split criterion relies on a measure of similarity between molecules. Based on this measure, the dataset is divided into 5 clusters using LSH clustering. This results in similar molecules being in the same cluster. One of these clusters will

be chosen as test set. Typically, the difficulty of this task is a balance between the random split (easy) and the scaffold split (hard), offering perhaps a fairer measure of performance.

We will see the comparison of the model performance with different split criteria in section 5.2.3.

Chapter 4

Modeling

4.1 ZairaChem pipeline

The modeling is performed with the ZairaChem tool. In section 2.4, we explained what descriptors and Machine Learning algorithm ZairaChem uses. Here we give an overview of the ZairaChem modeling pipeline.

- Data pre-processing

  Check that the input file contains a column with the SMILES strings (input variable) and the activity value (target variable, binary in our case).

  Apply the MELLODDY-Tuner protocol (Heyndrickx et al., 2022) to validate and standardize the SMILES strings.

- Obtain descriptors

  Various existing descriptor models from the Ersilia Model Hub are obtained for the molecules in the dataset. More details in section 2.4. This results in various vectors of several thousand variables.

  The descriptor variables are normalized, and missing data is imputed using nearest-neighbor.

  From the GROVER descriptor, an embedding is generated

- AutoML: On each set of descriptors and embedding, an AutoML method is run to fit a model. More details in section 2.4. This results in 9 submodels fitted.

- Pooling: The 9 submodels are aggregated to form an ensemble model.

- Reports and output: Performance reports are provided.

4.1.1 Pooling of submodel results

At the end of the pipeline, the results from the 9 submodels are aggregated into a single score using a weighted average. Before the averaging, the scores are scaled with quantiles. The weights are given by feature importance according to an independently trained random forest classifier.

4.1.2 ZairaChem vs *LazyQSAR*

As explained, ZairaChem builds an ensemble of various submodels, some of which are quite heavy to train and run.

In order to facilitate quick testing and prototyping, ZairaChem allows a faster way to fit a model using the LazyQSAR package, also developed by Ersilia. LazyQSAR

TABLE 4.1: Model run times

Pathogen	Task code	Total cases	Similarity split run time (h:mm)	Scaffold split run time (h:mm)
saureus	organism_anytype	61742	10:24	11:08
saureus	organism_anytype_hc	61742	10:23	8:37
saureus	organism_mic	48026	7:54	8:25
saureus	organism_mic_hc	48026	15:15	7:17
saureus	organism_iz	8178	1:55	1:43
saureus	organism_iz_hc	8178	1:35	1:33
saureus	organism_inhibition	7008	1:36	1:12
saureus	organism_inhibition_hc	7008	1:47	1:16
saureus	organism_ic50	1576	0:30	0:29
saureus	organism_ic50_hc	1576	0:29	0:28
saureus	protein_any	5008	1:17	0:59
saureus	protein_any_hc	4517	1:00	0:55
saureus	organism_assay_top1	4518	0:56	0:53
saureus	organism_assay_top1_hc	4518	0:54	0:50
saureus	protein_top1	616	0:19	0:15
saureus	protein_top1_hc	606	0:15	0:16
saureus	protein_top2	828	0:27	0:20
saureus	protein_top2_hc	683	0:17	0:18
saureus	protein_top4	696	0:23	0:18
saureus	protein_top4_hc	562	0:18	0:16
saureus	protein_top5	310	0:11	0:12
			Total: 58 hours	Total: 48 hours

uses only the two most basic submodels of ZairaChem: *classic* and *fingerprint*. These two submodels are considerably lighter to calculate but still offer fair predictive power. This way the LazyQSAR model can be used as a baseline model or as a preliminary trial.

4.2 Model training - Run times

As we have seen, our pipeline can be used to build models for any pathogen that has sufficient available data in the ChEMBL database. We are focusing in particular on the six ESKAPE pathogens. We have built the models for the six pathogens with LazyQSAR. We have chosen the pathogen with the largest amount of data, *Staphylococcus aureus*, to build full ZairaChem models as an illustration of the process.

Table 4.1 shows the run times of each model, for the two different train-test splits performed (*similarity* and *scaffold*). The models were trained on a PC with a 2.80 GHz Intel i7 processor and 16 GB of RAM.

The models for any other pathogen may be obtained by following the same process.

In chapter 5 we assess and discuss the results of the models.

Chapter 5

Model results

5.1 ZairaChem automatic report charts

ZairaChem produces by default a number of charts for every model created. These charts offer a useful overview of the data and model behavior. We show here some examples for illustration, from the output of model *saureus_organism_inhibition*.

In figure 5.1, on the left, we visualize the scores of the real positives (*Active*) versus those of the the real negatives (*Inactive*). This gives a good idea of the trade-off between the number of true positives we want to capture and the number of false positives we can accept. The higher we set the threshold to classify a value as positive, the higher the precision — at the cost of losing true positives.

In the same figure, on the right, we have the confusion matrix based on a threshold value set by ZairaChem.

Figure 5.2 shows two especially interesting charts, also created by ZairaChem for every model. These are projections (PCA and UMAP) of one of the descriptor vectors (grover) into two dimensions . The interpretation is that points that are close to each other in the projections have probably similar values of the descriptor, and may have properties in common.

These projections may for example help identify clusters of active components, or problematic situations such as zones where there is a lack of training data.

5.2 Analysis of model results

We have trained the following models:

- All tasks for all 6 ESKAPE pathogens, trained with LazyQSAR (total 70 models)
- All tasks for the pathogen *Staphylococcus aureus*, trained with full ZairaChem (total 21 models)

Each model has been trained twice: using the similarity and the scaffold train-test splits. A detailed list of the models is shown in appendix B.

The performance metrics produced by Zairachem have been compiled for analysis. To compare the performance of different models we use the AUROC metric (Area under the Receiver Operating Characteristic). This can be used to answer interesting analysis questions.

5.2.1 Question: Are some pathogens easier to model?

We compare the performance of the models trained on each of the 6 pathogens. We use the *LazyQSAR* models, because for the moment the full ZairaChem models have

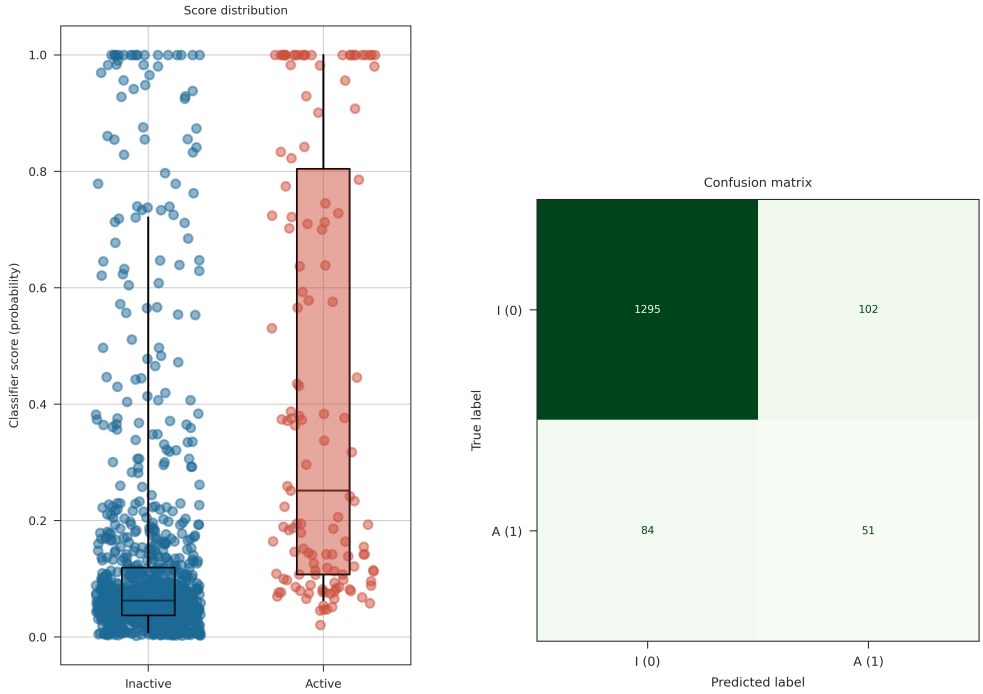


FIGURE 5.1: Sample Zairachem model assessment charts

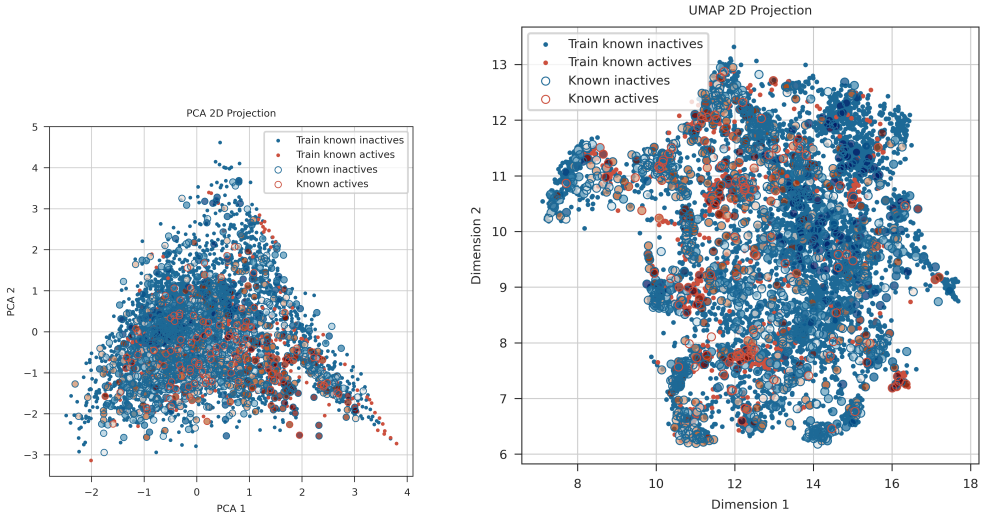


FIGURE 5.2: Example of projections in Zairachem output

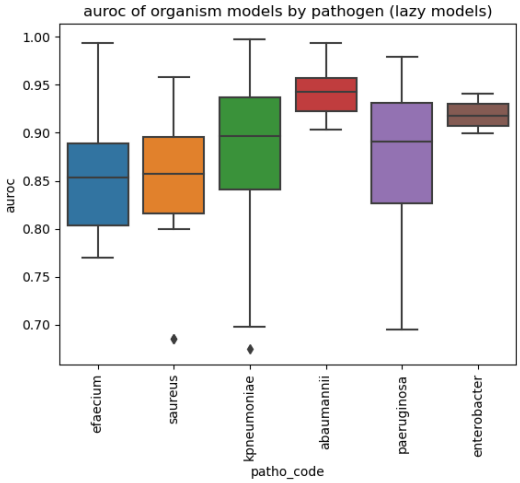


FIGURE 5.3: Compare between pathogens

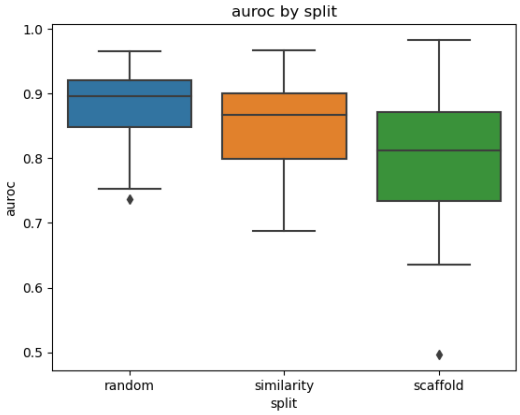


FIGURE 5.4: Compare between split criteria

only been trained for one pathogen. We focus only on organism models, not protein models. The reason is that protein models exist only in some pathogens and may behave differently, confusing the analysis. Besides, the specific proteins used are very different between pathogens.

The results include both the similarity split and the scaffold split.

The answer to the question is **yes**, as we see in figure 5.3. Predicting the activity against pathogens *s. aureus* and *e. faecium* seems to be the most difficult, obtaining a median performance slightly above 0.85. On the other hand, it seems easier for *a. baumannii* and *enterobacter*, obtaining a median AUROC well above 0.90.

5.2.2 Question: Are the similarity and scaffold splits harder to predict than a simple random split?

We expect a random split to be relatively easy to predict, and the similarity and scaffold splits to be harder.

Indeed we see in figure 5.4 that the performance of the models is highest with the random split, a bit less with similarity, and the lowest with the scaffold split.

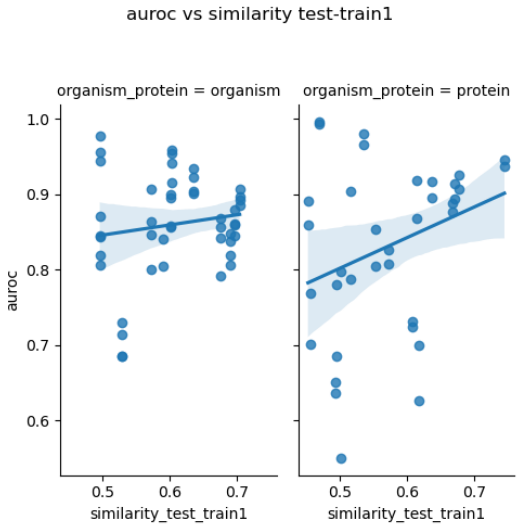


FIGURE 5.5: Compare between split criteria

5.2.3 Question: Is there a relationship between how similar the train and test sets are, with the model performance?

This is in a way analyzing the same issue as in the previous question, but with a different strategy: the Zairachem output provides for each predicted molecule, a measure of similarity between that molecule and those used to train the model. From there we can calculate the average similarity of the whole test dataset with the training dataset.

The result (figure 5.5) shows a slight increase of the AUROC for models with a higher train-test similarity, as expected. That is, the prediction seems more difficult for molecules very different from the training set. However, the amount of data does not allow a solid conclusion, as we see by the very wide tolerances of the regression lines.

5.2.4 Question: How much better is ZairaChem than LazyQSAR?

The full ZairaChem model is an ensemble of 9 submodels, while the LazyQSAR approach uses only 2 of those submodels. We expect therefore that the performance of the full ZairaChem is better (at the cost of longer training time).

For the pathogen *s. aureus* we have run both the LazyQSAR and the full ZairaChem models. We can therefore compare the performance of both approaches.

As we see in figure 5.6, the improvement of ZairaChem vs LazyQSAR is highest when we use the scaffold split. This is the most demanding one, that simulates trying to predict new molecules that are very different from those used to train the model. On the other hand, if we use the random split, the advantage of ZairaChem is almost zero, meaning that a simpler model like LazyQSAR is good enough for this easier job. The similarity split is a bit of an intermediate situation.

This result is OK, given that LazyQSAR is already a fairly good modeling tool. We suspect that the ZairaChem method for pooling the 9 submodels has margin for improvement, as we see in the following analysis. Besides, ZairaChem is flexible to easily add more descriptors from the Ersilia Model Hub.

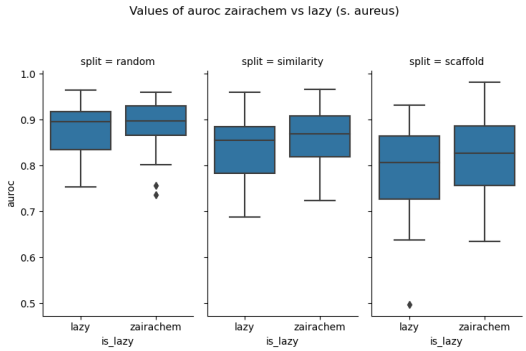


FIGURE 5.6: Compare between ZairaChem and LazyQSAR, by split

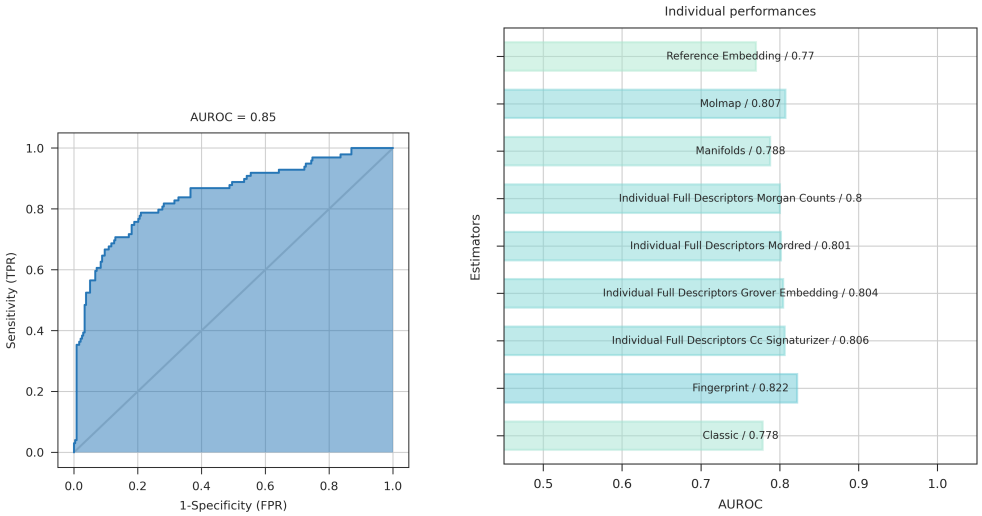
5.3 Analysis of the model pooling

We have seen in section 5.2.4 that the ZairaChem models are performing just marginally better than the LazyQSAR. Since ZairaChem performs an ensemble of 9 submodels and LazyQSAR uses just 2 of those submodels, we may wonder if the ZairaChem ensemble is successful at getting the most information possible from the submodels.

5.3.1 Initial exploration of the submodels

To get an initial idea of the submodels behavior, we pick two models and have a look at the output chart that ZairaChem generates comparing the performance of the submodels. On the left we see the ROC curve of the ensemble, on the right the AUROC of each individual submodel.

The following charts show the output of the model *saureus_organism_inhibition*:



We see that the ensemble model is better than any of the individual models. This would be the ideal case. The next charts show the output of the model *saureus_organism_ic50*:

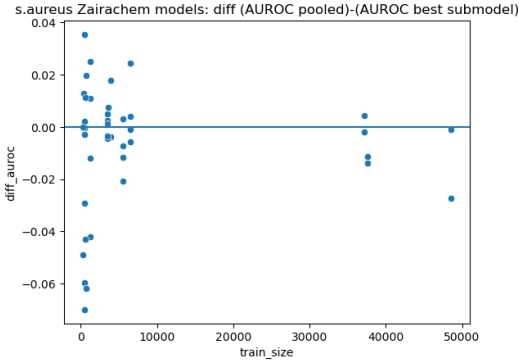
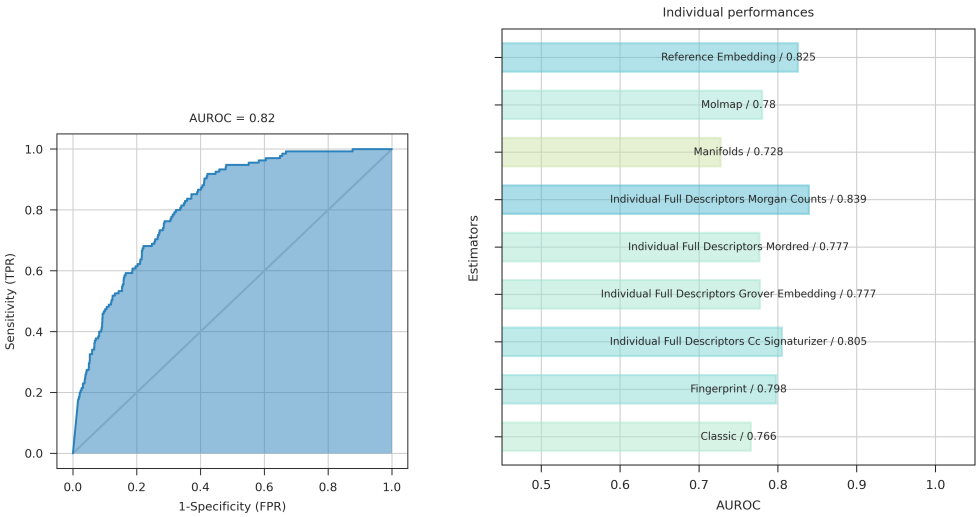


FIGURE 5.7: Within each model, compare pooled performance with best submodel



In this case we see that the ensemble model is actually performing worse than one of the submodels.

5.3.2 Compare ensemble with best submodel

Now we want to check this in general. One initial analysis is to compare the AUROC of the ensemble model versus the best submodel.

Ideally, we expect that the pooled model is often as good or even better than the best submodel. We check this in figure 5.7. The pooled model is sometimes better, sometimes worse, and often very similar to the best submodel. This is not bad but makes us suspect that there is margin for improvement in the pooling system.

We wonder which are the submodels that sometimes outperform the pooled model, in case we find a clear pattern. Figure 5.8 shows this. Apparently the classic fingerprint model is often better than the pooled. The number of models is not enough for a strong conclusion, but it does give a hint for a future improvement of the pooling system.

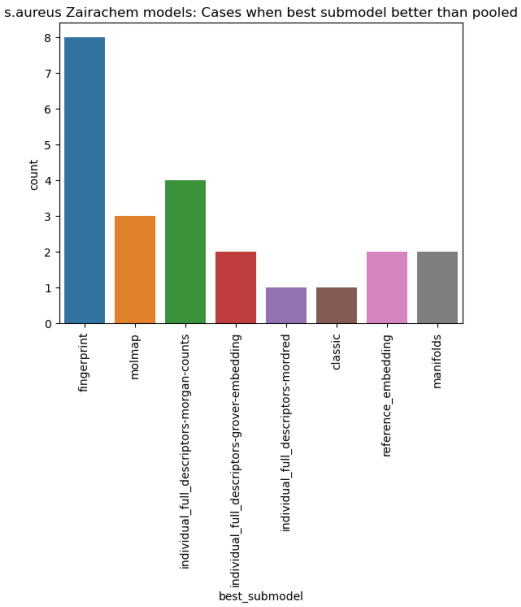


FIGURE 5.8: Count of cases of submodels outperforming the pooled model

5.3.3 Compare cases where predicted values differ

One final question to check: in an ensemble model, we would hope that there is not too much overlap in the information provided by the different submodels, so that the ensemble can in a way take the best from each.

One way to check this is to look at individual predictions and see if the pooled model and the submodels are right or wrong in the same cases.

For one of the models, *saureus_organism_ic50*, we have generated and compared the predicted values of the pooled model and of each of the submodels.

We take all the cases where the pooled prediction is wrong. Then we check, for each of the submodels, how often the submodel prediction would be correct:

correct cc_signaturizer	35%
correct morgan_counts	24%
correct mordred	33%
correct grover	16%
correct manifolds	22%
correct classic	22%
correct reference_embedding	22%
correct fingerprint	24%
correct molmap	30%

This seems indicative that the ensemble method is not making the most of the available information from the submodels.

Chapter 6

Conclusion and future directions

6.1 Conclusion

We have built a pipeline that can be used to obtain models that predict the effectiveness of a molecule against a given pathogen. This can be applied to any pathogen for which there is enough data in ChEMBL. In principle the pipeline does not require to make any decisions on the models, other than monitoring the results.

The full pipeline has been successfully applied to generate models for the pathogen *s. aureus*. This is a good illustrative example because it has enough amount of data to generate models for all the defined tasks. It is also a good test case to verify that the system works with relatively large data.

A partial version of the pipeline (the LazyQSAR models) has been run for the 6 ESKAPE pathogens. This was useful as a feasibility study, and also to get an idea of the predictive performances we can expect from this system.

This project has been the first case of large-scale, systematic use of ZairaChem. This has been a benefit in itself, as it has helped find and solve some bugs, and to propose improvements in functionality and usability of the ZairaChem modeling tool.

The analysis of the model results indicates that there is margin to improve the performance of the ZairaChem ensemble model (the pooling of the submodels).

6.2 Future directions

The models for the pathogen *s. aureus* have been trained. Now the models will be further validated by domain experts, and then a distilled version will be created. The final models will be published for open access in the Ersilia Model Hub.

The other 5 ESKAPE pathogens will follow the same process using the established pipeline until they are published.

After this, the pipeline is available to automatically build models on any other pathogen for which there is sufficient data in ChEMBL. As we saw in section 2.3.2, ChEMBL has abundant data on the majority of the pathogens that cause the most deaths in the world. We hope this tool will contribute to the drug discovery process in low-resource environments. The Ersilia Open Data Initiative currently has active projects in Cameroon and South Africa dedicated to bioactivity prediction against some of these pathogens.

As for improvements in the ZairaChem tool, there is the intention to investigate a better way to combine the results of the submodels to improve the predictive power. In addition, the tool will continue incorporating new descriptors and AutoML techniques.

Appendix A

Source code

The main repository of this project is: https://github.com/ersilia-os/antimicrobial-ml-tasks

The functions to obtain the input data and the sampled negatives from the ChEMBL database, also developed by me, are in a different repository. The reason is that these functions are more generic and prepared to be useful for other projects using ChEMBL. The repository is: https://github.com/ersilia-os/chembl_ml_tools

The ZairaChem tool has been developed by Ersilia Open Data Initiative. Here I have made a small contribution and provided some feedback for fixes and improvements. ZairaChem repository: https://github.com/ersilia-os/zaira-chem

The project relies on the Ersilia Model Hub, also developed by Ersilia Open Data Initiative: https://github.com/ersilia-os/ersilia

Appendix B

List of datasets

TABLE B.1: List of datasets for the ESKAPE pathogens

Pathogen code	Task code	Total cases	Positive cases	Total cases
efaecium	organism_anytype	5475	1856	34%
efaecium	organism_anytype_hc	5475	910	17%
efaecium	organism_mic	5025	1747	35%
efaecium	organism_mic_hc	5025	841	17%
efaecium	organism_inhibition	110	40	36%
efaecium	organism_ic50	152	76	50%
efaecium	organism_ic50_hc	132	65	49%
saureus	organism_anytype	61742	18507	30%
saureus	organism_anytype_hc	61742	9559	15%
saureus	organism_mic	48026	14143	29%
saureus	organism_mic_hc	48026	7577	16%
saureus	organism_iz	8178	3722	46%
saureus	organism_iz_hc	8178	1898	23%
saureus	organism_inhibition	7008	676	10%
saureus	organism_inhibition_hc	7008	365	5%
saureus	organism_ic50	1576	562	36%
saureus	organism_ic50_hc	1576	143	9%
saureus	protein_any	5008	2504	50%
saureus	protein_any_hc	4517	1416	31%
saureus	organism_assay_top1	4518	130	3%
saureus	organism_assay_top1_hc	4518	57	1%
saureus	protein_top1	616	308	50%
saureus	protein_top1_hc	606	138	23%
saureus	protein_top2	828	414	50%
saureus	protein_top2_hc	683	240	35%
saureus	protein_top4	696	348	50%
saureus	protein_top4_hc	562	281	50%
saureus	protein_top5	310	58	19%
kpneumoniae	organism_anytype	20147	4257	21%
kpneumoniae	organism_anytype_hc	20147	2567	13%
kpneumoniae	organism_mic	12894	2899	22%
kpneumoniae	organism_mic_hc	12894	1903	15%
kpneumoniae	organism_iz	2782	1310	47%
kpneumoniae	organism_iz_hc	2782	673	24%
kpneumoniae	organism_inhibition	5538	39	1%
kpneumoniae	organism_ic50	163	65	40%

Continued on next page

Table B.1 – continued from previous page

Pathogen code	Task code	Total cases	Positive cases	Total cases
kpneumoniae	organism_ic50_hc	163	31	19%
kpneumoniae	protein_any	448	224	50%
kpneumoniae	protein_any_hc	363	150	41%
abaumannii	organism_anytype	28111	678	2%
abaumannii	organism_anytype_hc	28111	287	1%
abaumannii	organism_mic	4530	604	13%
abaumannii	organism_mic_hc	4530	244	5%
abaumannii	organism_inhibition	23396	33	0%
paeruginosa	organism_anytype	51533	5405	10%
paeruginosa	organism_anytype_hc	51533	2380	5%
paeruginosa	organism_mic	23467	3214	14%
paeruginosa	organism_mic_hc	23467	1397	6%
paeruginosa	organism_iz	4987	1998	40%
paeruginosa	organism_iz_hc	4987	947	19%
paeruginosa	organism_inhibition	24244	306	1%
paeruginosa	organism_inhibition_hc	24244	162	1%
paeruginosa	organism_ic50	740	116	16%
paeruginosa	protein_any	2911	1230	42%
paeruginosa	protein_any_hc	2911	750	26%
paeruginosa	organism_assay_top1	24074	47	0%
paeruginosa	protein_top1	658	152	23%
paeruginosa	protein_top2	571	209	37%
paeruginosa	protein_top2_hc	571	110	19%
paeruginosa	protein_top3	916	458	50%
paeruginosa	protein_top3_hc	808	404	50%
paeruginosa	protein_top4	197	74	38%
paeruginosa	protein_top4_hc	197	34	17%
paeruginosa	protein_top5	116	44	38%
enterobacter	organism_anytype	3798	1682	44%
enterobacter	organism_anytype_hc	3798	1330	35%
enterobacter	organism_mic	3525	1617	46%
enterobacter	organism_mic_hc	3525	1292	37%
enterobacter	protein_any	330	162	49%
enterobacter	protein_any_hc	330	94	28%
enterobacter	protein_top1	256	127	50%
enterobacter	protein_top1_hc	256	76	30%

Bibliography

Duran-Frigola, Miquel et al. (2020). "Extending the small-molecule similarity principle to all levels of biology with the Chemical Checker". In: *Nature Biotechnology* 38.9, pp. 1087–1096.

Erickson, Nick et al. (2020). "Autogluon-tabular: Robust and accurate automl for structured data". In: *arXiv preprint arXiv:2003.06505*.

Heyndrickx, Wouter et al. (2022). "Conformal efficiency as a metric for comparative model assessment befitting federated learning". In.

Ikuta, Kevin S et al. (2022). "Global mortality associated with 33 bacterial pathogens in 2019: a systematic analysis for the Global Burden of Disease Study 2019". In: *The Lancet* 400.10369, pp. 2221–2248.

Mendez, David et al. (Nov. 2018). "ChEMBL: towards direct deposition of bioassay data". In: *Nucleic Acids Research* 47.D1, pp. D930–D940. ISSN: 0305-1048. DOI: 10.1093/nar/gky1075. eprint: https://academic.oup.com/nar/article-pdf/47/D1/D930/27437436/gky1075.pdf. URL: https://doi.org/10.1093/nar/gky1075.

Moriwaki, Hirotomo et al. (2018). "Mordred: a molecular descriptor calculator". In: *Journal of cheminformatics* 10.1, pp. 1–14.

Namba-Nzanguim, Cyril T. et al. (2022). "Artificial intelligence for antiviral drug discovery in low resourced settings: A perspective". In: *Frontiers in Drug Discovery* 2. ISSN: 2674-0338. DOI: 10.3389/fddsv.2022.1013285. URL: https://www.frontiersin.org/articles/10.3389/fddsv.2022.1013285.

O'Malley, Tom et al. (2019). *KerasTuner*. https://github.com/keras-team/keras-tuner.

Schmidt, Friedemann et al. (2014). "Predictive in silico off-target profiling in drug discovery". In: *Future Medicinal Chemistry* 6.3, pp. 295–317.

Shen, Wan Xiang et al. (2021). "Out-of-the-box deep learning prediction of pharmaceutical properties by broadly learned knowledge-based molecular representations". In: *Nature Machine Intelligence* 3.4, pp. 334–343.

Turon, Gemma et al. (2022). "First fully-automated AI/ML virtual screening cascade implemented at a drug discovery centre in Africa". In: *bioRxiv*.

Wang, Chi et al. (2021). "FLAML: A fast and lightweight automl library". In: *Proceedings of Machine Learning and Systems* 3, pp. 434–447.