Gemma Bel Bordes

Computational prediction of the mechanisms of action of drugs using machine learning algorithms

Degree Final Project

Conducted by Dr Marta Sales Pardo
and Dr Roger Guimerà Manrique

Bachelor's degree of Biomedical Engineering

UNIVERSITAT ROVIRA i VIRGILI

Tarragona
2021

Abstract

Over time, drug design costs have risen, although it has not impacted the number of successful drugs. Machine learning tools could potentially change this situation by introducing new methods to assess the properties of drugs, such as their mechanism of action. We have looked for the best model to predict these mechanisms, given gene expression and cell viability data of the cells treated with the drug, employing well-known algorithms such as support vector machines and a stochastic block model. The former resulted in the best predictive ability, while the latter remains a good model for interpreting the data interactions.

Keywords: machine learning; mechanism of action; gene expression; cell viability.

Acknowledgements

I would like to express my most sincere gratitude to Marta and Roger for their guidance and for giving me this opportunity, which has been very gratifying.

Also to the other members or former member of SEES:lab research group. Especially to Toñi, who did not hesitate to help me with the optimization of the code.

Last but not least, to my family, including Pol, and all my friends. It has been a tough year, and they were always there to comfort me.

Contents

1 Introduction 1

1.1 Aims of the project 3

2 Data 4

2.1 Data Origin 4
2.2 Data structure 5
2.3 Data characterization 6
2.4 Data pre-processing 7
2.5 Data splitting 10

3 Machine Learning algorithms to predict Mechanisms of Action 11

3.1 General concepts of Machine Learning 11
3.2 Algorithms particularities 12
3.2.1 Off-The-Shelf models 12
3.2.1.1 Random Forest 12
3.2.1.2 Support Vector Machine 13
3.2.1.3 Logistic Regression 14
3.2.1.4 Naïve Bayes 14
3.2.2 Stochastic Block Model 15
3.3 Metric 19

4 Implementation of the algorithms and results 20

4.1 Off-The-Shelf models 20
4.1.1 Random Forest 20
4.1.2 Support Vector Machine 22
4.1.3 Logistic Regression 23
4.1.4 Naïve Bayes 24
4.1.5 Metamodels 24
4.1.6 Summary 25
4.2 Stochastic Block Model 27
4.2.1 Only gene expression 28
4.2.1.1 Number of iterations choice 28
4.2.1.2 Weight of gene expression choice 28
4.2.1.3 Number of groups choice 30
4.2.2 Both gene expression and cell viability 30
4.2.3 Summary and interpretation of the SBM 31

5 Discussion 38

6 Conclusions 41

7 Bibliography 42

Appendix 1. Programming code 47

Appendix 2. Additional figures 47

1 Introduction

Drug design and drug discovery are challenging processes in biomedical research due to the complexity of the effects those new potential drugs could have when tested in living conditions. A drug introduces a perturbation within the cell leading to molecular changes that will produce, in turn, its pharmacological effect. These changes are coined in the literature as Mechanisms of Action (MoA), the importance of which has not always been considered by pharmaceutical researchers; as a matter of fact, it was the influence of molecular biology on drug discovery that triggered the pursuit of understanding the biochemical MoAs of new chemical compounds [1]. Even though some authors do not agree on the optimal time when the MoAs of a drug should be elucidated [2], mechanistic understanding is paramount since it can help infer the value of drug therapies, leading to enhanced safety and efficacy [3].

Although drug discovery has introduced many technological breakthroughs and takes advantage of significant R&D investment, it has not reached the community expectations, and the benefits remain to be materialized in terms of productivity [4], [5]. This failure can be attributed to different factors such as the continuously decreasing acceptance of risk tolerance [6] or the faulty approaches to discover novel drugs [3], [4]. Many drugs are withdrawn after a long research period because they could not foresee the toxicity and the off-target effects of the proposed compounds [5]–[7]. Thus, the number of drugs that finally meet with the approval of the expert agencies such as the Food and Drug Administration is still low compared with what could be hoped given the advances on disease understanding [4]. By contrast, some recent publications are trying to give a positive view of the pace the pharmaceutical industry follows by working side-by-side with academia [8]–[11], suggesting that further innovations will be well received.

This low attrition rate can not be explained in one single way [4]–[7], [12], [13]. However, a better understanding of the compounds under study's inherent properties will help tackle efficacy and safety problems [5], which happen to be the main reasons for clinical failure [14]. Consequently, new ways which could contribute to the anticipation of such problems before entering the clinical phase should be developed by researchers. It is now when the concept of MoA could play an important role; as mentioned earlier, MoA understanding has been demonstrated to directly impact whether a drug will be safe or efficient enough since it associates molecular interactions with their response [3], [15].

Identification of the MoAs of drugs is not an easy task, especially in Phenotypic Drug Discovery (PDD), one of the most used approaches in drug design. PDD is antagonistic to the Target-based Drug Discovery (TDD) in terms of the processes order they follow. While in TDD, the very first step is to identify the target which the drug should attack, PDD is target (and MoA) agnostic at the beginning so that a large number of compounds will be tested to identify phenotypical changes of interest. On the one hand, PDD is closer to an empirical approach, which makes it more attractive for first-in-class drug discovery (i.e. drugs using a new MoA for that specific condition) [16]. On the other hand, this latter methodology will require subsequent efforts to discover the effects of that compound in the cell, which can become a significant drawback of PDD [17], [18]. Nevertheless, MoA elucidation should be essential in both approaches [2].

Different methods aim to detect the mechanisms of action and the targets of drugs (novel or already on the market compounds). Broadly speaking, those can be either direct

biochemical methods or indirect methods [18]. Direct biochemical approaches focus on the affinity between the target and the drug, and it often involves prior chemical modifications [18] (e.g., fluorescence labels) and thorough experiment preparation [17]. Alternatively, indirect methods comprise not only approaches related to omics data, from genomics to metabolomics, but also computational techniques [17]. Both of them are really close to one another since databases are essential for bioinformatics tools.

Computational tools offer new opportunities in the drug discovery workflow, since they provide the opportunity to explore existing information about drugs and MoAs that does not require expert consultation. Apart from long periods used for research, during this last decade, the cost of R&D in the pharmaceutical area has been dramatically rising, as shown in Fig. 1, thus creating an expensive and time-consuming context [19]. Taken together, it seems plausible to identify some of the steps of the drug discovery chain as potential tasks to be enhanced by Machine Learning (ML) algorithms [20]–[22] in order to reduce the expenses and enhance the results. ML is expected to lead to a resurgence of the pharmaceutical field [21], [22] due to its potential for identifying hidden patterns in the data, which could be challenging to be detected by traditional analysis.

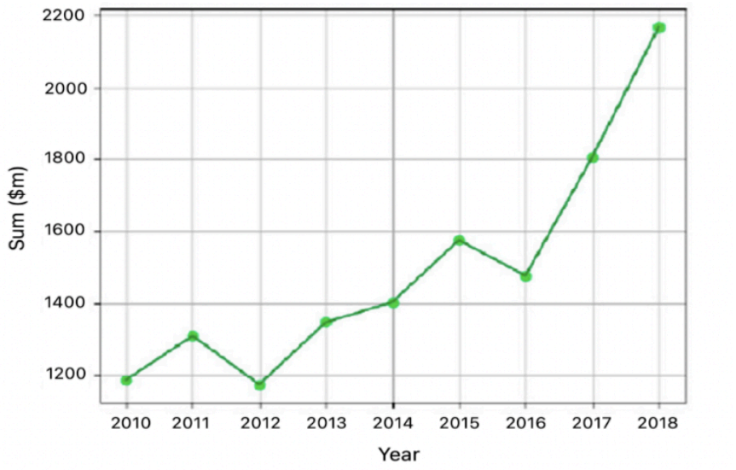


Figure 1. Evolution of average drug development cost in millions of dollars for a cohort of 12 major pharmaceutical labs. Adapted from [23].

Regarding recent studies that some research groups have been working on in order to predict the mechanisms of action of drugs through ML, it can be appropriated to divide them into different methodologies depending on the data used as the input of their approaches. Up to now, two general methods can be identified [17]: profiling methods and structure-based methods (or ligand-based methods). This last group might be more restrictive because of the need for previous knowledge of the compound structure; notwithstanding, some authors suggest that structural biology is paramount for interpreting drug-target mechanisms and interactions [24]. The profiling methods group contrastingly seek models that use data from experiments which focus on phenotypic profiles induced by the drugs, and employ a pattern recognition [17]. In a nutshell, cells treated with a specific compound will suffer changes resulting in a particular phenotypic profile, which will give insight into how the drug works and which can be captured either with imaging tools [25]–[28] or gene expression approaches

[29]–[33]. Data collections such as the Connectivity Map (CMap) [34], which aims to connect diseases, treatments and genes, have quickly become popular since their launching to guide gene expression-based models [35].

Many studies have shown that introducing gene expression is critical for predicting MoAs. Several tools have been developed for this purpose like MANTRA [29] or DEMAND [31]. For MANTRA, they created a drug network based on similarities among gene signatures after treating different cell lines at different dosages and times, with which they looked for communities that could be identified with a specific MoA [29]. Alternatively, DEMAND also uses a molecular interaction network besides the data from gene expression profiles. The main objective was to assess profile changes to obtain a ranked vector of genes affected by the drug and, after that, to verify it with experimental data (turning out to be a hybrid method) [31]. Lately, there have been attempts to integrate other relevant data such as cell viability [36] or results coming from CRISPR-Cas9 screens (a technology with which they can obtain gene expression profiles for gene knockout experiments) [33]. There is still room for improvement in terms of the algorithms performance and data largeness since most of these models only use a small sample of drugs, cell lines and other conditions related to the treatment of those cells like the dosage and the time. This gap could be filled thanks to: (1) improvements of the Connectivity Map leading to a scaled-up platform termed L1000 [37], which has been able to represent up to a million gene profiles; and (2) the contribution of diverse institutions to offer novel and complete datasets including, cell viability results of the same experiments of drug treatment on top of gene signatures [38].

1.1 Aims of the project

The main objective of this project is to find the best ML model to predict the MoA (or MoAs) of any drug based on the effects it has at a cellular level when tested in a pool of several cell lines, from which gene expression and cell viability data is obtained.

In order to find the best model, we have set other specific aims, which are:

- Assess the importance of gene expression and cell viability data for predicting the MoAs of a drug.
- Implement several Off-The-Shelf ML algorithms aiming to predict the MoAs and compare their performance.
- Implement a Stochastic Block Model, a more sophisticated algorithm, to predict the MoAs and to understand the interactions within the data.
- Interpret the results of the built algorithms and comprehend how they get to the MoA assignation, if possible.

2 Data

2.1 Data Origin

All the data used in this project comes from the *Kaggle* competition "Mechanisms of Action Prediction" [38], which agrees on exploiting it for academic research. *Kaggle* is the data science platform of reference, and they often host competitions like this one so as to engage worldwide people to take part in its community while they stimulate unique solutions as a response to new challenges. In this case, three distinguished institutions and projects related to the data science field (Connectivity Map [39], the Laboratory for Innovation Science at Harvard [40] and the NIH Common Funds Library of Integrated Network-Based Cellular Signatures [41]) proposed the MoA competition. They aligned their forces to provide a unique collection of gene expression and cell viability data, gathered with mechanisms of action information.

There is not much information about the new technology that could simultaneously measure cell viability and gene expression of different cell lines within the same sample. However, we do know the underlying methodology for both of the analysis.

- Gene expression was measured through an assay based on L1000.

  Even though the laboratory protocol they followed is out of the project's scope, the general concept is interesting. Given that gene expression is highly correlated [42], Subramanian et al. [43] hypothesized that any cellular state could be captured at a lower cost if only directly analysing essential genes. Such reduced representation should recover, in turn, most of the information of the full transcriptome, meaning that they would only consider the most informative genes, which they referred to as *landmarks*. Once they concluded that 1.000 genes should be sufficient to explain the whole transcriptome, they chose the landmark genes through statistical tools such as a principal component analysis and a k-Means clustering. Briefly, in the end, the L1000 assay platform can provide information on the expression of a thousand genes by using a ligation-mediated amplification approach and further processing involving coloured stains and fluorescence (for more technical details see [37], [44]).

- Cell viability was measured through an assay based on PRISM.

  Cell viability assessment does not involve the degree of complexity of gene expression. PRISM approach [45], which stands for Profiling Relative Inhibition Simultaneously in Mixture, consists basically in labelling each cell line with a unique DNA bar code. After that, cells are treated with the drug for a few days, and then they analyse the quantity of each bar code remaining, mirroring a count of still available cells (for more technical details see [45]).

The bottom line at this point is that this *Kaggle* assay consisted of measuring how 100 different human cell lines gathered in a single sample responded to drug treatment of more than 5000 drugs over periods from 1 to 3 days at two different dosages. Attached to this was the actual sample's mechanisms of action. Moreover, as in any experiment, some of the cell combinations were exempt from drug testing so as to leave them as control samples.

Due to its nature, the competition provided both training and test datasets; the former set to create the model and the latter to check its accuracy. However, the test set lacked information on the MoAs, so, for us, there was no point in using it since we would have nothing to compare our predictions with. Hence, the datasets to work with were the ones containing the measurements and the treatment information each sample had undergone (training features) as well as the MoAs of each mixture (training targets). Important aspects that could have been biologically helpful, particularly the name of the drugs, cells and genes, were hidden from the data initially, albeit they revealed them after the contest deadline.

2.2 Data structure

Both features and targets files contain the same number of rows, each accounting for a specific sample, which may have undergone different experimental processes. Since all these samples were initially alike and what makes them different is how they were treated afterwards, we will refer to these rows as conditions. There is a total of 23.814 rows in these datasets, including treated and non treated (i.e., control) conditions, which have a specific identification code (**sig_id**).

On the one hand, the file containing the features (i.e., the properties or measurements that the model will use to predict the response and characterize each condition) has 875 columns which hold the following meanings:

- **Column 1: cp_type** indicates whether the condition has been treated with a drug compound or not. It is a binary variable whose categories are *trt_cp* (treated) and *ctl_vehicle* (non treated).
- **Column 2: cp_time** stands for how long a condition, if applicable, has been treated. There are three options: *24*, *48* or *72* hours.
- **Column 3: cp_dose** represents the dosage level chosen for the treatment. It is also binary since it can be low (*D1*) or high (*D2*).
- **Columns 4-775: g-** columns comprises the continuous measurements of gene expression levels of the condition. The hosts of the competition z-scored and normalized the raw values of the gene expression through quantile normalization [38], which aims to homogenize their distributions to get rid of technical-based differences.

  Once normalized, gene expression takes values from -10 to 10.
- **Columns 775-875: c-** columns contain the normalized values of cell viability levels measured for each cell line.

  Cell viability values should be between -10 to 10 as well. However, since we are not working with all the data points from the experiment (just with the *Kaggle* training sets), it turns out that the actual maximum value represented in the dataset is 6,4 instead of 10.

As a summary of the aforementioned structure, Table 1 shows two conditions with their respective features. The first one is an example of a treated condition, whereas the latter

stands for a non treated sample. Information about treatment time and dosage is annotated in both, albeit it should not be useful for the second condition.

Table 1. Features dataset example. The rows account for the conditions or samples of the experiment and the columns for the different features.

sig_id	cp_type	cp_time	cp_dose	g-0	...	g-771	c-0	...	c-99
id_000644bb2	trt_cp	24	D1	1,0620	...	-0,0224	-0,060	...	0,4176
id_fffc1c3f4	ctl_vehicle	48	D2	0,3942	...	0,3603	1,065	...	0,3808

On the other hand, there is also the targets dataset, depicted in Table 2. In this case, the targets (i.e., the variables the model will have to predict) are the mechanisms of action the conditions can have, integrated in the dataset as a collection of 206 columns. The values these columns will contain will be either 0s or 1s, depending on the absence or the presence of that MoA in that condition. Such mechanisms could imply, for example, the blockage of molecular channels or the inhibition of a specific protein, which happens to be the most frequent MoA in the data (inhibitor of the nuclear factor kappa-light-chain-enhancer of activated B cells - *NF-kB inhibitor*).

Table 2. Targets dataset example. The rows account for the conditions and the columns for each one of the targets or MoAs.

sig_id	5-alpha_reductase_inhibitor	...	gsk_inhibitor
id_000644bb2	0	...	1
id_fffc1c3f4	0	...	0

2.3 Data characterization

As a first step of understanding the data and its interactions, we can extract some analytical information. In this case, conditions become the unifying thread of the whole problem, so we are interested in assessing the relationships between conditions and the other variables. These variables, as a reminder, were essentially treatment information, gene expression, cell viability and mechanisms of action.

Firstly, according to the information about the treatment these conditions might have undergone, we can confirm the expected details. The proportion of control samples is low with respect to the treated ones. Only 7,8% of conditions have been non-drug tested. From the resting 92,2%, while half of them were treated with a high and low dose respectively, if we analyse the proportion of conditions treated for 24, 48 and 72 hours, we can verify that each third of the conditions correspond to these treatment time periods.

Then, for gene expression variables, a heat map could be valuable to detect patterns in this data since this data is nothing but an extensive matrix of values belonging to a specific condition and a given gene. Heat maps will depict the data as a grid with coloured boxes describing a specific measurement value. These graphs are typical in research, primarily when representing gene-based datasets. As a consequence of the large number of conditions we are working with, Fig. 2 shows the gene expression heatmap drawn for a reduced representation

of these conditions (1022 out of the original 23814). Furthermore, this graph has been hierarchically clustered through dendrograms to make it easier to detect possible hidden patterns or groups within the data. Knowing that each vertical line of Fig. 2 represents a condition, whereas the horizontal lines stand for the different genes, we can notice that some conditions have extreme gene expression values, either positive or negative, placed at both sides of the graph. From those most coloured parts, we can conclude that some groups of genes behave similarly. Therefore, we can determine around ten groups of genes overall by analysing the most left side of the heat map.

The importance of the last variables is such that additional statistical analysis may be worthy. The next aspect to examine is the difference between the control and the treated conditions over their gene expression and cell viability outcomes. We computed the z-scores of all these values against the controls (i.e., using the mean and the standard deviation of non treated conditions). By doing so, we mathematically calculate how many standard deviations a specific value is away from the control mean. Fig. 3 and Fig. 4 show the distributions of the z-scores of a representative gene and cell, respectively. As expected, control samples generally have a fairly narrower distribution than those tested with a drug.

Finally, as a consequence of the datasets including multiple columns to associate each condition with their MoA, we can hypothesize that not only can a condition have a single mechanism of action, but it also can be related to a bunch of mechanisms at the same time. However, we have found out that most of them have only one MoA annotated, as depicted in Fig. 5. In addition, some conditions have no MoA annotated. Unsurprisingly, all the control conditions fit this latter case because they should not induce any significant biological change. Still, a minority of drug-treated conditions do not have any MoA, which triggers the possibility that either those drugs did not have an effect or did have an effect, but the mechanism is not one of the 206 represented in this data.

2.4 Data pre-processing

In order to prepare the data for the machine learning algorithms implementation, it must undergo some pre-processing techniques with the aim to avoid futher problems and to create new combinations of these datasets.

In the first place, as seen in 2.2., there are two columns of the features dataset whose values are not numerical. Those variables are *cp_type* and *cp_dose* and must be converted to numbers to avoid further problems. For *cp_type*, we will substitute *trt_cp* with a 0 (treated condition) and *ctl_vehicle* with a 1 (control condition). For *cp_dose*, a 0 will mean a low dose (originally *D1*) and a 1 will represent a high dose (originally *D2*).

A variety of new datasets can emerge from the partition of the data if setting some restraints. In this case, we have thought that it might be helpful to have it divided into different datasets according to the treatment time and dose of the conditions. In the end, there will be six sub-datasets representing the six possible combinations of treatment duration (24, 48 and 72 h) and dosage (low or high). By doing so, we will be able to create multiple models. Thus, if such combinations of these two variables happen to be significant for the behaviour of the conditions, the performance of the predictive algorithms could increase.

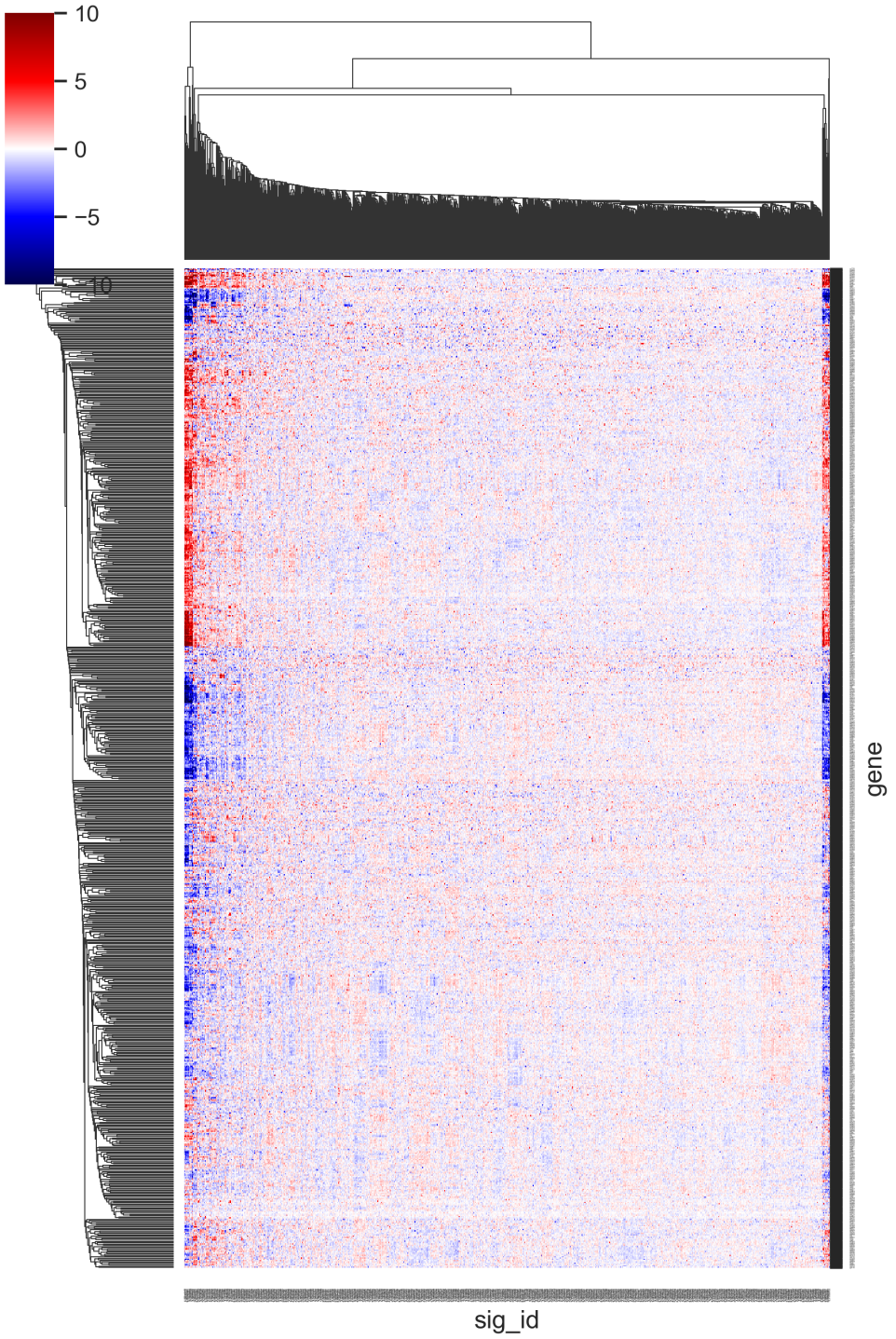


Figure 2. Gene expression heat map with a reduced representation of the conditions (1022 conditions out of the 23814). Dendrograms hierarchically clustered both rows, which correspond to genes, and columns, corresponding to conditions. Each element has a colour assigned based on its gene expression value.

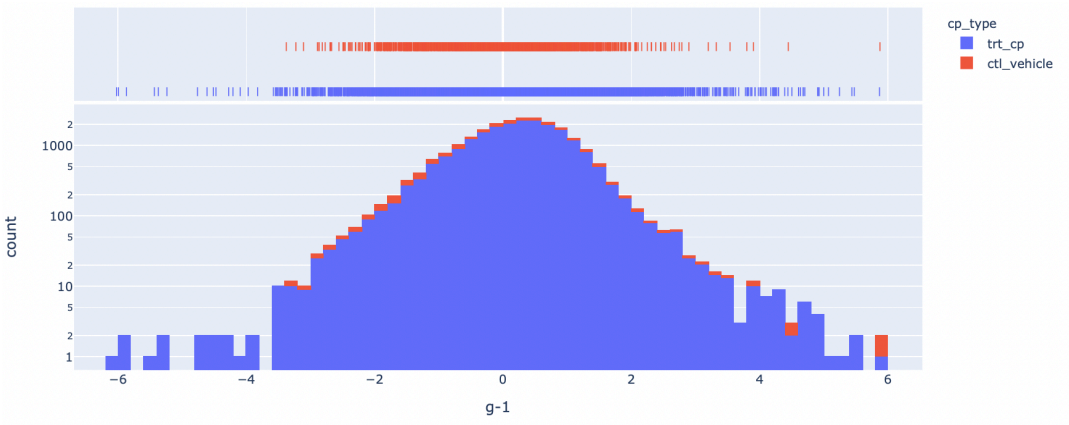


Figure 3. Z-scores distribution of gene expression for *gene 1*. The colours account for the condition type, treated (trt_cp) or control (ctl_vehicle). The y axis is log-scaled, and it represents the number of treated conditions (in blue) that falls into each gene expression interval. The control conditions have been scaled up to resemble the treated conditions distribution, but the counts should be lower for them.

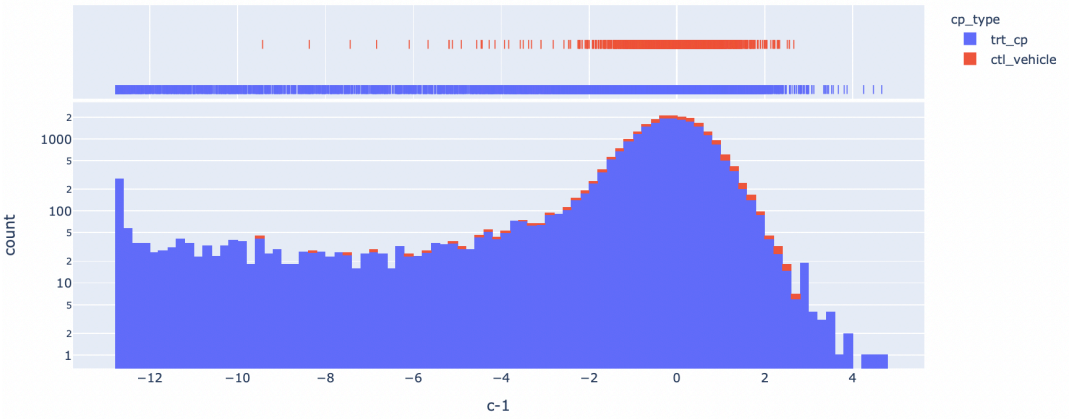


Figure 4. Z-scores distribution of cell viability for *cell 1*. The colours account for the condition type, treated (trt_cp) or control (ctl_vehicle). The y axis is log-scaled, and it represents the number of treated conditions (in blue) that falls into each cell viability interval. The control conditions have been scaled up to resemble the treated conditions distribution, but the counts should be lower for them.

Some machine learning algorithms to be used later need discrete data as the input. In our case, the originally continuous variables are those including genes and cells measurements, and we should discretise them. To do so, we need to determine threshold values and decide how many counterparts these variables should finally have. On the one hand, we want to know which genes are over-expressed, normally-expressed or under-expressed, forcing the number of counterparts to be three. On the other hand, we want to differentiate the cell lines that have been less viable from those that have better maintained their population, so we will just need two labels to represent them.

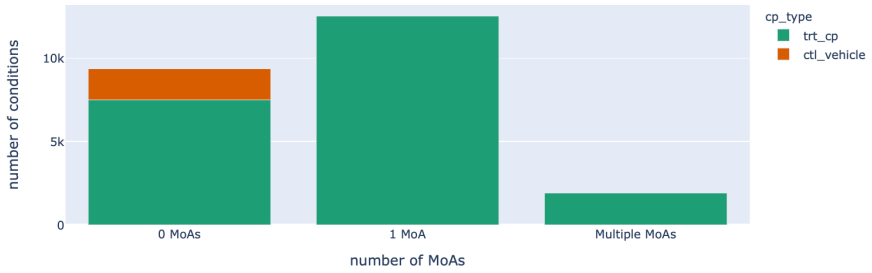


Figure 5. Comparison between the number of mechanisms of action each condition has annotated. The colours account for the condition type, treated (trt_cp) or control (ctl_vehicle).

The initial idea might be that the drug treatment is the main responsible for those states of over- or under-expression of a gene and the decrease of cell viability. For that reason, we want to compare the distributions of the values belonging to the treated conditions versus the controls, which we consider as ordinary. We will take advantage of the z-scores distributions obtained from the last section in order to choose threshold values that can separate typical values, including most of the control conditions, from those out of the ordinary, in which the minimum number of control points should fall in:

- For gene expression, we observe in Fig. 3 that a z-score of $\pm$ 5 can reach the goal of separating well enough all these values in three new categories: -1 for under-expressed (until -5), 0 for normally-expressed (from -5 to 5) and finally 1 for over-expressed (for z-scores higher than 5).
- For cell viability, focusing on Fig. 4 and considering that only two categories are needed for the discretisation, a reasonable threshold could be a z-score of -6. So, in the end, there will be two discrete labels to express those values that account for a drop of cell viability (for z-scores lower than -6), represented with a 0, and those that have held a stable population (for values higher than -6), annotated with a 1.

2.5 Data splitting

We must be divided all the data into training and test datasets before implementing any algorithm. The models will use the training datasets ”to learn”, while test sets will be useful to validate these models. Given the significant number of conditions of the data, we will divide it randomly so as to have 95% of them for training (i.e., 22623 conditions) and the other 5% as the test (i.e., 1191 conditions). All the algorithms that we will implement will consider the same data splitting.

3 Machine Learning algorithms to predict Mechanisms of Action

Given the complexity of the data and the project's aims, we need tools to model this data to understand its inherent properties and predict the MoAs of drugs. Machine Learning (ML) or Statistical Learning (SL) offers plenty of methods to fulfil these objectives, although some of them will be better for a specific goal.

In this section, we will theoretically define the algorithms to model the data. In the first experimental part of the project, we will work with algorithms known for being "Off-The-Shelf", meaning that some programming libraries (in this case, in the *scikit learn* package of *Python*) have already coded. This selection includes the following algorithms: Random Forest, Support Vector Machine, Logistic Regression and Naïve Bayes. In contrast, the second experimental part focuses on a more sophisticated method, the Stochastic Block Model, which requires additional endeavour to programme its functionality. All these models will be finally validated and compared to check which one performs better in terms of predictiveness.

The section includes: firstly, a theoretical framework including general concepts of ML followed by explaining the particularities of each algorithm, and, finally, the definition of the metric that will check the performance of these algorithms.

3.1 General concepts of Machine Learning

ML or SL offers tools that learn from the given data with the pursuit of understanding it. Particularly, this learning can be supervised, which involves the algorithm to link inputs with known outputs, or unsupervised when there are no annotated outputs. This project clearly requires the former option since for each observation belonging to the inputs, which are the variables related to the experiments and the measurements, there is an associated response: having an MoA or not.

Before delving into the particularities of each algorithm, it is essential to have some concepts clear in terms of notation. In general, the technicalities of the following explanations related to ML and these methods come from [46], unless noted otherwise.

In section 3, we have already use the terms features and targets to refer to the different datasets we have. On the one hand, the features correspond to the data retrieved from the experiments, including both measurements (i.e., gene expression and cell viability) and treatment indications (i.e., time, dosage and treatment type). These features are the input variables for these algorithms and can be denoted with an X.

On the other hand, the targets stand for the annotated MoA of a condition. Since the mechanisms are those variables we want to predict, we will refer to them as the output variables, denoted with a Y. ML focuses on modelling the relationship between X and *Y* throughout a function f, which can be estimated with the different proposed algorithms. Besides, a random error term ϵ plays a role as well in this association. Therefore, the relationship between X and Y can be understood as

$$Y = f(\boldsymbol{X}) + \epsilon. \tag{1}$$

Suffice it to note that there are 206 different mechanisms of action that the model must consider in our case. As a consequence of having a multi-output problem with 206 Ys, 206

functions f have to be estimated, meaning that we will use the same algorithms several times with the same inputs, although the output will differ. If the error term, whose mean is zero, is obviated, we would have

$$Y_n \approx f_n(\boldsymbol{X}) \tag{2}$$

to be repeated 206 times (where n goes from 1 to 206).

The estimation of the function f is, in fact, what ML algorithms do. This function may be estimated both for inference, which accounts for understanding how changes in the inputs X affect the outputs Y, and for prediction, in which the purpose is to predict new outputs accurately. The latter usually implies that the function associating X and Y is not always well understood, causing it to be treated sometimes as a "black box", which can lead to reluctance when proposing the algorithms to the scientific community or industry. However, such algorithms can work well with a great variety of problems. In addition, the function can be estimated by means of parametric or non-parametric approaches. The distinction here is that parametric methods require a previous step of assuming the shape of the function (e.g., linear), causing, in turn, to estimate a set of parameters instead of a somewhat abstract function, whereas non-parametric approaches do not assume its functional form and, for that reason, can be more flexible. Nonetheless, this flexibility is often a synonym for less interpretability of the model, making the box mentioned before to be even more opaque.

To conclude this general introduction to ML and to understand our choice of algorithms to be used, it must be clear that there are two types of problems in ML overall relating to the properties of the output of the model. Regression problems are those in which the targets are quantitative variables (i.e., numerical values), while classification problems involve qualitative or categorical output variables, which happens to be the case of the project. Even though our targets take values of 1 or 0, these are different categories accounting for *yes* or *no* when asking if a condition has a specific MoA.

In a nutshell, the project aims to tackle a multi-output classification problem through different types of machine learning algorithms. The data will be divided as mentioned in 2.5. into a training and a test set. The former will estimate the functions f that associate the features or inputs with the MoAs, and, in the end, the same functions will work with the test inputs to predict the MoAs of these new entries.

3.2 Algorithms particularities

3.2.1 Off-The-Shelf models

3.2.1.1 Random Forest

Random Forest (RF) models combine several decision trees. These trees aim to segment the features space into several simple regions, known as splitting, which focuses on rules this data follows. The main idea is to divide the data recursively and binarily into different nodes depending on the value of a specific variable to classify it. The divisions aim to finally leave each node as pure as possible (i.e., containing predominantly observations from a single class).

To exemplify how decision trees work, imagine we only take into account the features

of the condition type (*cp_type*), the dosage of the treatment (*cp_dose*), gene expression of (*g-1*) and cell viability of (*c-1*) to understand the classification of a mechanism of action (the labels will account for having that MoA or not). One of the RF individual trees could have a similar shape as in Fig. 6. It is important to bear in mind that Fig. 6 is made-up; it does not represent the actual pattern of our data. Given that a single RF model includes many trees, the final prediction of the MoA would be given by the most voted class each tree predicted.

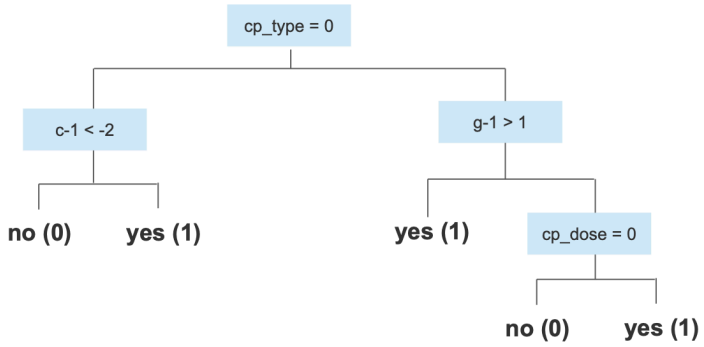


Figure 6. Made-up example of how a decision tree works for classifying the data. The tree finally classifies the condition into having or not having a mechanism of action, based on binary questions considering the values of its features.

Even though RF are ready for use in *Python*, there are some parameters that users can control of this model. For example, we can specify the number of trees, the minimum samples a node must have to split it into two internal nodes, among other aspects.

3.2.1.2 Support Vector Machine

Support Vector Machine (SVM) aims to represent the classification problem through an n-dimensional feature space divided by hyperplanes, flat affine subspaces of $n-1$ dimensions. In this context, if only having two X variables, the hyperplane would be a simple line dividing the plane, and the classification would depend on the side of that plane the observation fell in. In addition, this line (or any subspace needed to represent the feature dimensions) should maximize the distance between the observations that belong to each Y category, creating a gap between them. However, in SVM, there is room for a few training observations to be along this gap and be on the wrong side of the space to make the model more robust. Fig. 7 depicts another made-up example of how a two-dimension feature space could be modelled by the SVM, considering two gene expression variables to classify the points (which account for the conditions) into the category of having a given MoA or not.

Moreover, the boundary that separates the observations can take on different shapes apart from the linear hyperplane mentioned above. This shape can be modified when working with SVMs by means of what is known as kernels, which can be, for example, linear, radial or polynomial.

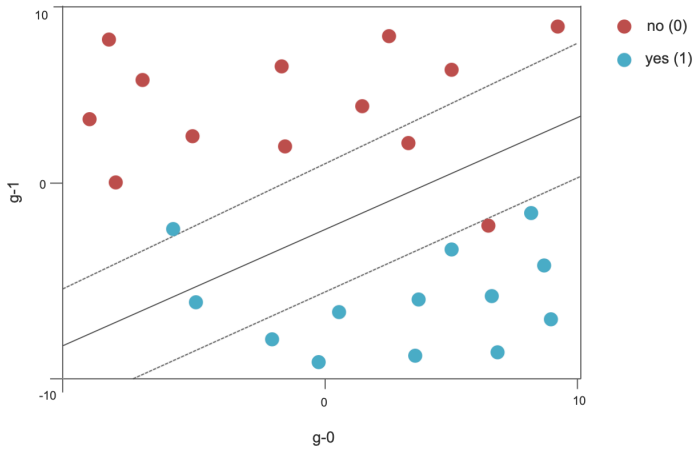


Figure 7. Made-up example of how a support vector machine for a two-dimensional feature space work for data classification. Depending on the localization of a condition in this plane, the support vector machine will classify it into having or not having a mechanism of action. The lines account for the hyperplane and the margins that separate both classifications.

3.2.1.3 Logistic Regression

Logistic Regression (LR) is a classification algorithm, although its name can be confusing. It models the probability that the response belongs to a particular category in problems that have binary responses. This probability is calculated like

$$p(Y = 1|\boldsymbol{X}) = \frac{e^{\beta_0+\beta_1\boldsymbol{X}}}{1 + e^{\beta_0+\beta_1\boldsymbol{X}}} \quad (3)$$

where β_0 and β_1 are coefficients that will be estimated from the data employing the maximum likelihood method, which aims to find those coefficients that better reproduce the original response for each variable of the feature space.

To illustrate the bottom line of this method, we will suppose that a gene expression variable (e.g., *g-0*) can classify the conditions based on the presence or the absence of a particular MoA. The estimated probabilities of having the MoA could be as in Fig. 8, where low gene expression values would have a low probability of having the MoA while the opposite case happens for high values. Again, this representation exemplifies how the model could work using the actual terminology used with our data, although it is made-up.

3.2.1.4 Naïve Bayes

Naïve Bayes (NB) models rely on Baye's theorem and the naïve assumption that each pair of features is independent. The objective is to assign the probability for a condition to have a particular MoA, considering its features. This probability can be calculated with Baye's

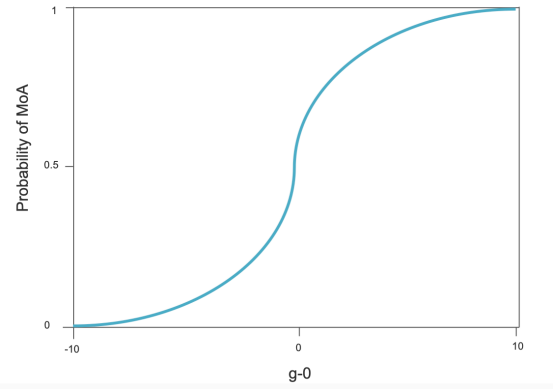


Figure 8. Made-up example of the probability of having a specific mechanism of action by the logistic regression model when only one features variable (i.e., gene expression for *g-0* is taken into account). If the condition had a low value of gene expression, the probability of having that mechanism would be very low, while it would happen the opposite for a high value of gene expression.

theorem as

$$p(y|\boldsymbol{X}) = \frac{p(\boldsymbol{X}|y) \cdot p(y)}{p(\boldsymbol{X})}, \tag{4}$$

where X are the features and y one of the two options related to the MoA (y could be the fact of either having or not having that MoA). Since X is composed of different variables ($X = x_1, ..., x_n$), but these are independent, in the end the posterior (i.e., the probability calculated above) is given by

$$p(y|\boldsymbol{X}) = \frac{\prod_{i=1}^{n} p(x_i|y) \cdot p(y)}{\prod_{i=1}^{n} p(x_i)}. \tag{5}$$

There are different ways to compute the likelihood term, we will use the Gaussian approach, which allows the features to be continuous (which is the original case for gene expression and cell viability variables) assuming they follow a normal distribution with the respective mean (μ) and variance (σ^2). Each individual likelihood will be calculated as

$$p(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}e^{(\frac{-(x_i-\mu_y)^2}{2\sigma_y^2})}}. \tag{6}$$

In general, NB models are good classifiers but poor estimators, meaning that the final classification are more reliable than the actual probabilities assigned to each class [47].

3.2.2 Stochastic Block Model

The second part of the experimental part focuses on the increasingly popular Stochastic Block Model (SBM), which does not work like the other algorithms for which we will have to

build several individual models concerning each MoA. This generative approach shapes the whole data through a graph or network from which different communities can emerge. An SBM will create a non-directed graph with nodes related to others by edges, and where the main idea is that some nodes can be part of a group or community, which, in turn, will be highly connected to another community of nodes.

However, instead of a basic SBM that only considers the possibility for a node to be part of a single group, we will be using a Mixed Membership Stochastic Block Model (MMSBM). By using so, a node does not have to belong to a specific group; instead, it can belong to a mixture of those.

Considering this is still quite an original algorithm for ML, we will thoroughly explain it, emphasizing the mathematical idea underneath. We have specially defined the equations for this project (following the concepts and terminology used in [48]).

All the values that associate conditions with features and targets must be discrete. Therefore, we will work with discrete gene expression and cell viability datasets (as described in 2.4). It is important to note that this model will not consider the features related to the information about the experiment (i.e., neither type of condition, treatment dose, nor treatment time) to reduce the system's complexity.

As in the other cases, the objective is to predict the MoA, m, associated to each one of the C conditions the data includes, c. For each condition we have information of the expression of G genes and the cell viability of V cells. For conditions in the training set we also have the list of MoAs associated to them out of M possible MoAs.

We will assume that there are underlying groups of conditions and MoAs, so that conditions in the same group are associated with the same groups of MoAs. We assume that there are K and L underlying groups of conditions and MoAs, respectively. Since we are using MMSBM, we then allow each condition c to belong to any condition group α with probability $\theta_{c\alpha}$; we also allow each MoA m to belong to any β MoA group with probability $\theta_{m\beta}$. θ_c and η_m are called membership vectors and because they represent the distribution of the memberships over groups they must satisfy the normalization condition:

$$\sum_{\alpha=1}^{K} \theta_{c\alpha} = 1 \ , \qquad \sum_{\beta=1}^{L} \eta_{m\beta} = 1 \ . \tag{7}$$

MMSBMs further assume that the probability that a condition in group α is associated with a MoA in group β is $p_{\alpha\beta}$ and not associated with probability $1 - p_{\alpha\beta}$. To ease with the notation, we define a variable r_{cm} which is equal to 1 if condition c is associated to MoA m and to 0 otherwise so that then we can write:

$$p_{\alpha\beta}(r) = \begin{cases} p_{\alpha\beta} & r = 0 \\ 1 - p_{\alpha\beta} & r = 1 \end{cases} \qquad \sum_{r} p_{\alpha\beta}(r) = 1, \tag{8}$$

which defines a probability matrix **p**.

Given the membership vectors and the association probability matrix, then the probability that condition c has an association r_{cm} with MoA m is:

$$p(r_{cm}|\boldsymbol{\theta}, \boldsymbol{\eta}, \mathbf{p}) = \sum_{\alpha,\beta} \theta_{c\alpha} p_{\alpha\beta}(r_{cm}) \eta_{m\beta} \quad . \tag{9}$$

Because we assume that MoAs are related to gene expression and cell viability data, we will assume that we can model the relationship between genes and conditions and between cell viability data and conditions also using an MMSBM. Furthermore we will assume that the group membership vectors for conditions (i.e., $\boldsymbol{\theta}$) are the same in the three cases.

The discretisation of gene expression and cell viability allowed us to set three different relationship values e between genes and conditions e, e_{cg}=0 if gene g has a normal expression level in condition c, e_{cg}=1 if gene g is over-expressed in condition c, and e_{cg}=-1 if gene g is under-expressed in condition c. From here, we assume that there are T underlying groups of genes, and define membership vectors for genes $\boldsymbol{\phi}_g$, so that $\phi_{g\gamma}$ is the probability that gene g belongs to gene group γ. We also define the probability matrix **q** so that $q_{\alpha\gamma}(e)$ is the probability that a condition in group α had an association of type e with a gene in group γ. As before $\boldsymbol{\phi}$ and **q** are subject to the normalization conditions:

$$\sum_{\gamma=1}^{T} \phi_{g\gamma} = 1 \ , \qquad \sum_{e} q_{\alpha\gamma}(e) = 1 \ . \tag{10}$$

Given the membership vectors for conditions and genes and the probability matrix **q** the probability that in condition c gene g has expression level e is

$$p(e_{cg}|\boldsymbol{\theta}, \boldsymbol{\phi}, \mathbf{q}) = \sum_{\alpha,\gamma} \theta_{c\alpha} q_{\alpha\gamma}(e_{cg}) \phi_{g\gamma} \quad . \tag{11}$$

We do the same for association between conditions an cell viability. In this case cell viability of condition c and cell type v is also a binary variable t_{cv}=0, 1. We assume that there are F underlying groups of cells and we define a membership vectors for each cell v, $\boldsymbol{\xi}_v$ so that $\xi_{v\delta}$ is the probability that cell v belongs to cell group δ. We also define an association probability matrix **s** so that $s_{\alpha\delta}(t)$ is the probability that cell v under condition c has cell viability type t.

Given the membership vectors and the association probability matrix, the probability that in condition c, cell v has cell viability of type t_{cv} is:

$$p(t_{cv}|\boldsymbol{\theta}, \boldsymbol{\xi}, \mathbf{s}) = \sum_{\alpha,\delta} \theta_{c\alpha} s_{\alpha\delta}(t_{cv}) \xi_{v\delta} \quad . \tag{12}$$

We are interested in predicting unobserved associations between conditions and MoAs, given the observed associations between conditions, MoAs, gene expression levels and cell viability, $\mathsf{D} = \{\{r_{cm}\}, \{e_{cg}\}, \{t_{cv}\}\}$. As a reminder, we have a set of conditions C^o for which we have a complete set of training observations (including MoA annotations) and a set of test conditions C^t of which we only know gene expression levels and cell viabilities.

In order to control for the effect of gene expression levels and cell viability, we define a log Posterior ($\log P(\text{model}|\mathsf{D})$) that gives different weights to the different terms, accounting for MoA, gene and cell information:

$$\begin{aligned} \log P(\text{model}|\mathsf{D}) \ &= \ \sum_{c\in C^o, m} \log p(r_{cm}|\boldsymbol{\theta}, \boldsymbol{\eta}, \mathbf{p}) + \lambda_g \sum_{c\in C^o+C^t, g} \log p(e_{cg}|\boldsymbol{\theta}, \boldsymbol{\phi}, \mathbf{q}) \\ &+ \ \lambda_v \sum_{c\in C^o+C^t, v} \log p(t_{cv}|\boldsymbol{\theta}, \boldsymbol{\xi}, \mathbf{s}) \end{aligned} \tag{13}$$

where λ_g and λ_v are the weights given to gene expression and cell viability information, respectively.

Our goal is to find the model parameters that maximize the posterior, which are $\{\boldsymbol{\theta}^\star, \boldsymbol{\eta}^\star, \boldsymbol{\phi}^\star, \mathbf{p}^\star, \mathbf{q}^\star, \mathbf{s}^\star\}$. With these parameters we can compute the probability of r_{cm}=1 for unobserved associations between conditions and MoAs using the equation

$$p(r_{cm} = 1|\boldsymbol{\theta}^\star, \boldsymbol{\eta}^\star, \mathbf{p}^\star) = \sum_{\alpha\beta} \theta^\star_{c\alpha}\, p^\star_{\alpha\beta}(1)\, \eta^\star_{m\beta}\ . \tag{14}$$

To obtain $\{\boldsymbol{\theta}^\star, \boldsymbol{\eta}^\star, \boldsymbol{\phi}^\star, \mathbf{p}^\star, \mathbf{q}^\star, \mathbf{s}^\star\}$ we will use an Expectation Maximization algorithm with the following iterative equations for modelling such parameters:

- Expectation step, with which we will compute these auxiliary functions

$$\Omega_{\alpha\beta}(c, m) = \frac{\theta_{c\alpha} p_{\alpha\beta}(r_{cm}) \eta_{m\beta}}{\sum_{\alpha'\beta'} \theta_{c\alpha'} p_{\alpha'\beta'}(r_{cm}) \eta_{m\beta'}} \tag{15}$$

$$\psi_{\alpha\gamma}(c, g) = \frac{\theta_{c\alpha} q_{\alpha\gamma}(e_{cg}) \phi_{g\gamma}}{\sum_{\alpha'\gamma'} \theta_{c\alpha'} q_{\alpha'\gamma'}(e_{cg}) \phi_{g\gamma'}} \tag{16}$$

$$\Pi_{\alpha\delta}(c, v) = \frac{\theta_{c\alpha} s_{\alpha\delta}(t_{cg}) \xi_{v\delta}}{\sum_{\alpha'\delta'} \theta_{c\alpha'} q_{\alpha'\delta'}(t_{cg}) \phi_{g\delta'}} \tag{17}$$

- Maximization step, with which we will use the above auxiliary functions to obtain new values for the parameters of the model.

$$\theta_{c\alpha} = \frac{\sum_{m\in M_c,\beta} \Omega_{\alpha\beta}(c, m) + \lambda_g \sum_{g\in G_c,\gamma} \psi_{\alpha\gamma}(cg) + \lambda_v \sum_{v\in V_c,\delta} \Pi_{\alpha\delta}(cv)}{|M_c| + \lambda_g |G_c| + \lambda_v |V_c|} \tag{18}$$

$$\eta_{m\beta} = \frac{\sum_{c\in C_m,\alpha} \Omega_{\alpha\beta}(c, m)}{|C_m|} \tag{19}$$

$$\phi_{g\gamma} = \frac{\sum_{c\in C_g,\alpha} \psi_{\alpha\gamma}(c, g)}{|C_g|} \tag{20}$$

$$\xi_{v\delta} = \frac{\sum_{c\in C_v,\alpha} \Pi_{\alpha\delta}(c, v)}{|C_v|} \tag{21}$$

$$p_{\alpha\beta}(r) = \frac{\sum_{c\in C^o, m/r_{cm}=r} \Omega_{\alpha\beta}(c, m)}{\sum_{c'\in C^o, m'} \Omega_{\alpha\beta}(c', m')} \tag{22}$$

$$q_{\alpha\gamma}(e) = \frac{\sum_{c\in C^o+C^t, g/e_{cg}=e} \psi_{\alpha\gamma}(c, g)}{\sum_{c'\in C^o+C^t, g'} \psi_{\alpha\gamma}(c', g')} \tag{23}$$

$$s_{\alpha\delta}(t) = \frac{\sum_{c\in C^o+C^t, v/t_{cv}=t} \Pi_{\alpha\delta}(c, v)}{\sum_{c'\in C^o+C^t, v'} \Pi_{\alpha\delta}(c', v')} \tag{24}$$

where M_c is the set of MoAs with known association with condition c; G_c is the set of genes associated with condition c; V_c is the set of cells with known viability in condition c; C_m is the set of conditions with known association with MoA m; C_g is the set of conditions for which we know the expression level of gene g; C_v is the set of conditions with known cell viability of cell v; finally, $|X|$ is the cardinality of set X.

The iterative algorithm to obtain the model parameters that maximize the log posterior will be as follows:

1. We will generate values at random for $\boldsymbol{\theta}$, $\boldsymbol{\eta}$, $\boldsymbol{\phi}$, $\boldsymbol{\xi}$, **p**, **q** and **s** and then apply the normalization conditions.
2. We will compute the auxiliary functions $\Omega_{\alpha\beta}(c,m)$, $\psi_{\alpha\gamma}(c,g)$ and $\Pi_{\alpha\delta}(c,v)$ using the Expectation Step equations.
3. We will finally obtain new values for the parameters from the equations in the Maximization step.
4. We should repeat steps 2 and 3 until the algorithm converges (i.e. the differences in parameter values are below some threshold or the log Posterior stabilizes).

3.3 Metric

Once having the models, these will predict the MoAs of new conditions that has not been seen yet, which belong to the test set. These predictions will undergo a validation process in order to check the performance of each approach. There are different metrics to do so, accuracy and precision among them, which considers the label's prediction (1 or 0 in this case). Nevertheless, guided by the *Kaggle* competition rules [38], we will be using another score that is also typically suggested for classification problems like this one, which is the logistic loss score (log-loss).

Log-loss uses the probabilities of the predictions instead of the final assignation of the classes. It indicates how close the prediction probability is to the original label of the response. The log-loss is

$$score = -\frac{1}{M}\sum_{m=1}^{M}\frac{1}{N}\sum_{i=1}^{N}[y_{i,m}log(\hat{y_{i,m}}) + (1-y_{i,m})log(1-\hat{y_{i,m}})] \tag{25}$$

where N and M are the number of conditions and MoAs in the test set, respectively, $\hat{y_{i,m}}$ is the predicted probability of a positive MoA response for a condition and $y_{i,m}$ is the actual label. In addition, log corresponds to the natural logarithm.

The higher the log-loss, the worse the model is in terms of performance since it means that the probabilities diverge more from the original classification.

4 Implementation of the algorithms and results

In this section, we will combine the implementation of the algorithms mentioned above with the results collected along the way. We will follow the order in the previous section, so the first part focuses on Off-The-Shelf methods and the second part on the Stochastic Block Model. We have trained all of these algorithms with the whole training datasets and, posteriorly, validated them using the same models with the test set. In the end, we will combine all the results related to the performance of the models given their log-loss score to compare their predictiveness level.

We have been working with *Python* to programme all these algorithms (see Appendix 1).

4.1 Off-The-Shelf models

The common steps we must follow with each one of the algorithms of this first part (RF, SVM, LR and NB) will be:

1. Apply the algorithm with the training data to obtain the models. Since there are 206 targets columns (i.e., MoAs) to predict, there will be 206 individual models.
2. Use the obtained models with the test data to predict the probabilities of conditions to have a positive response for each MoA (i.e., probability of having a 1 associating a condition to an MoA).
3. Since all the control conditions should clearly have no MoA assigned, we will force such probabilities to be 0.0, no matter how they were predicted.
4. Compute the log-loss score to obtain the result accounting for the performance of the model.

Nonetheless, to conclude this part, we will implement two different methodologies based on the results given by these four Off-The-Shelf ML algorithms, known as metamodels. The steps to follow will be different, and we will specify them later.

4.1.1 Random Forest

We have implemented several RFs in different ways to analyse the effect of various aspects related to the algorithm functioning or the data that it considers.

Firstly, we designed a simple test to detect how modifications to the algorithm's parameters affect the log-loss score. There are several parameters to control, and it would be highly time-consuming to cover them all. That is why we focused on two of these parameters that turned out to be of great importance. By default, each RF combines 100 estimators or trees and determines that the minimum samples a node must have to continue splitting the tree is 2. The experiment, then, consisted in comparing the log-loss given by the models with different combinations of these two parameters. The number of estimators chosen for this first screening was 100 and 200, combined with 2 and 4 minimum samples per leaf. Table 3

summarizes the log-loss scored by the RF using each parameters combination. We can observe that increasing both the number of estimators and the minimum samples per leaf causes the model to work better.

Table 3. Summary of the log-loss values scored by the random forest with different values for the number of estimators and the minimum samples per leaf.

		Number of estimators	
		100	200
Minimum samples per leaf	2	0,0544	0,0364
	4	0,0376	0,0306

To conclude with this analysis of the parameter's effects, given that the minimum samples per leaf is a parameter that will not raise the time needed to create the model, we decided to continue exponentially increasing its value until the log-loss stabilises. All these further experiments worked with 200 trees for each RF. The log-loss had clearly stabilised when we reached the 64 minimum samples, as shown in Fig. 9. Finally, the best score for an RF employing the original datasets (with 200 trees and 64 minimum samples per leaf) is **0,0174**.

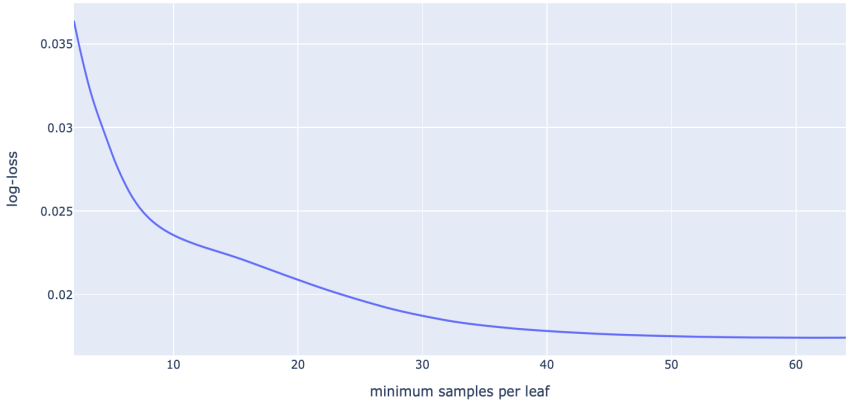


Figure 9. Evolution of the log-loss score based on the changes on the minimum samples per leaf of the random forest model.

Next, once we know the best selection of parameters (i.e., 200 trees and 64 minimum samples per leaf), we created various RF models for each group of conditions with a given combination of treatment dosage and time (as mentioned in 2.4), so we finally had six general models. We created a final dataset that gathered all the probabilities given by these to calculate the log-loss. Despite creating different models for this data partitioning, the log-loss did not improve, obtaining a score of **0,0187**.

Finally, we can retrieve deeper information that might be useful for the interpretability of the RFs, such as feature importance. Trees of an RF are split according to such variables that can better reduce the impurity of a node (i.e., to have more similar observations on a single node). So, variables or features will be more critical if they can significantly reduce this impurity and even more if they can lower it in nodes that contain many observations, which

are the nodes that are on the top of the tree. This way of assessing feature importance is known as Mean Decrease of Impurity (MDI), and we have measured it for all the features considered in each submodel used to predict each MoA, for the case of 200 estimators and 64 minimum samples per leaf. We could detect that, in general, there are just a few features that stand out, whereas the importance of the rest decreases slowly (as shown in Fig. 10 for the case of the *acetylcholine receptor antagonist* MoA). Among the most important features, we could never observe those accounting for treatment type, nor time, nor dose; most of these important features are gene expression variables, apart from a few cell viability variables.

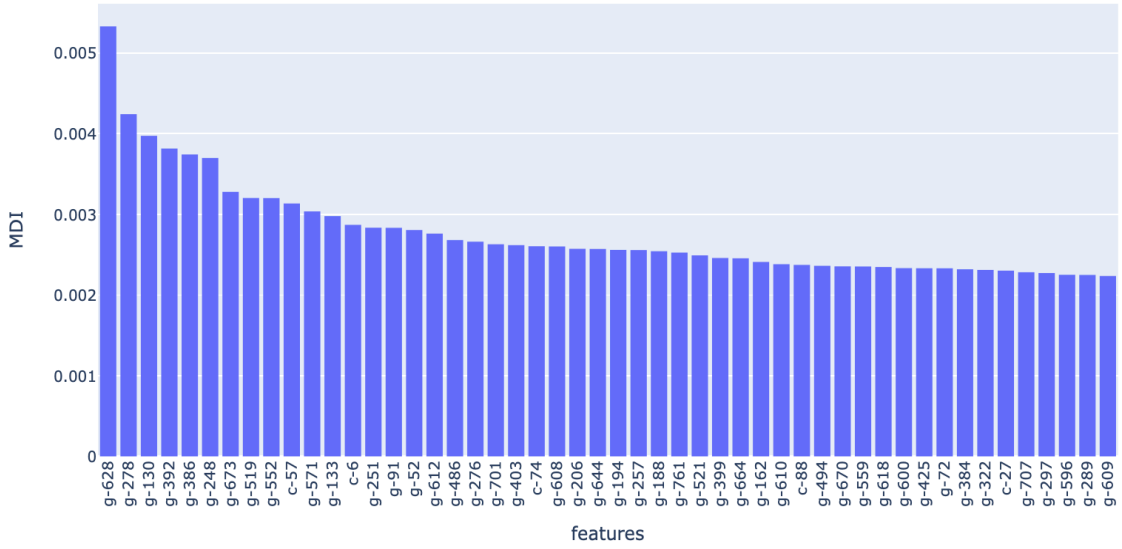


Figure 10. Feature importance for the prediction of the mechanism of action *acetylcholine receptor antagonist*. It represents the 55 most important features and their respective MDI (Mean Decrease of Impurity).

4.1.2 Support Vector Machine

Since SVMs are not as malleable as RFs, we performed fewer experiments with this algorithm. Firstly, we wanted to compare how the kernel used to model the data affected its performance. For that reason, we used a linear and a Radial Basis Function (RBF) kernel. The results of the log-loss score were:

- Linear kernel: **0,0159**.
- RBF kernel: **0,0149**.

Both SVMs significantly outperformed the best RF we had previously obtained. However, trying to model the complexity of our data with a linear kernel is not sufficient, or at least, the log-loss score is not as low as the log-loss of the RBF model, which is more flexible in terms of boundaries shapes that aim to divide the data.

Finally, we also followed the process of creating models for each dosage and time combinations, considering those six data partitions. In this case, the SVM employed an RBF kernel since it was the best option, as mentioned above. Nevertheless, it also turned out to have no positive effect. The log-loss score obtained in this case was **0,0166**. Therefore, dividing the data into such combinations is even more detrimental than using a simple linear kernel.

4.1.3 Logistic Regression

LR needed fewer features to build the models properly; otherwise, *Python* could not handle such an amount of data. We decided to address this problem by taking advantage of the feature importance rankings obtained from the best RF model. The bottom line was to select those most important features to predict each MoA. However, the number of features to be selected became an essential aspect of assessing.

First of all, we built an LR with the data accounting only for the 15 most important features for each MoA. Next, to pick these features more mathematically, we computed the z-scores of the MDI values of each individual model (used to predict each MoA). From these new values, we set a threshold to 2,5 to select only those features with a higher z-score (i.e., we will select all the features that are further than 2,5 standard deviations from the mean of MDIs) (see Appendix 2). With this latter selection, the number of features to be considered when building each submodel will differ. In Fig. 11, we can see the number of selected features for the first 25 MoAs, employing the z-score picking, which on average selects 22 features.

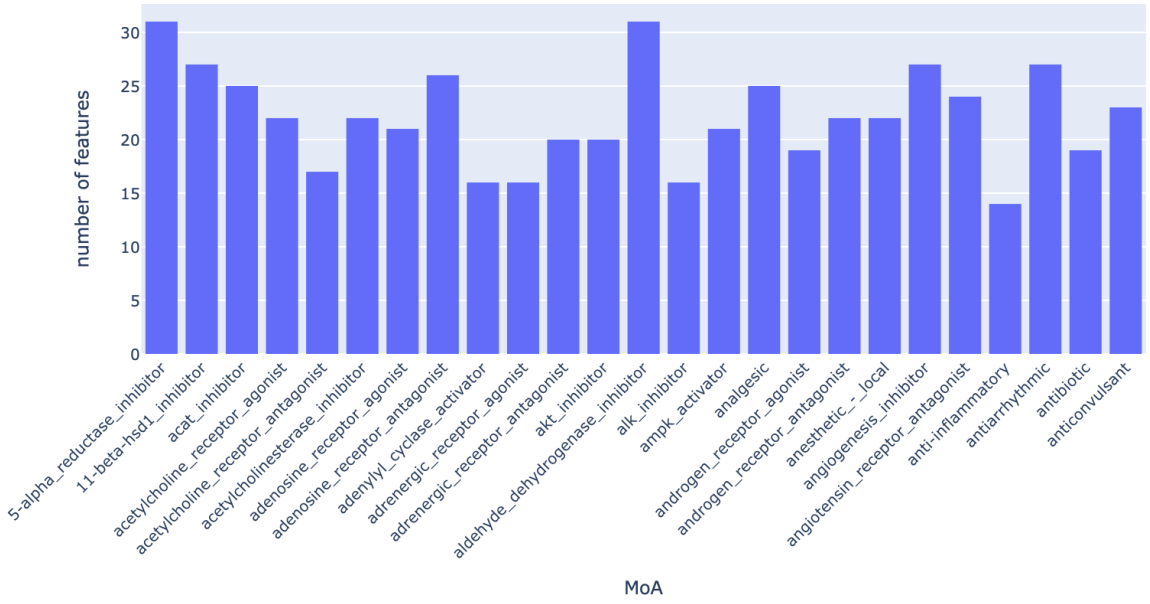


Figure 11. Number of features to be considered for the first 25 models (first 25 MoAs). The selection is based on the z-score values of the MDI (Mean Decrease of Impurity), picking those features with a z-score higher than 2,5.

The results obtained from implementing the LR for these new datasets were:

- 15 most important features selection: **0,0166**.
- Z-score-based most important features selection: **0,0167**

Hence, the LR works better when simply choosing a particular number of features instead of the z-scored-based selection. Despite significantly reducing the number of features to train and test the model, the LRs models are close to SVM and RF in terms of performance since the log-loss scores are similar.

Even though data partitioning based on time and dosage combinations did not improve the predictiveness of the previous algorithms, now that the model will not take such variables into account by any chance since the selection of features does not include them, we will try to do it again with the LR. However, it is still worthless since it scored a log-loss of **0,0199** (when using the top 15 features choice).

4.1.4 Naïve Bayes

For the NB algorithm, we followed the same steps as for the LR. As mentioned in 3.2.1.4., the approach to calculate this Bayesian model's likelihood is the Gaussian method, which requires continuous features. For that reason, and taking advantage of the selection of features we had done for the LR models, which did not consider treatment features as important ones, we will use the same methodology as above. We have implemented an NB with the 15 most important features data, the z-score-based selection of features data and, finally, the data partitioned into dosage and time combinations. The results are:

- 15 most important features selection: **0,200**.
- Z-score-based important features selection: **0,4094**.
- Dosage and time partitioned data: **0,1978**.

Such results are far from the rest; the scores are not in the same magnitude order, reasserting NB being a poor estimator. However, a surprising aspect is that it has been the only case for which a model did improve, although slightly, the performance when dividing the data into different combinations of dosage and time.

4.1.5 Metamodels

A metamodel is as a model of models. In other words, it is a methodology used to take advantage of the model's results, trained with the data of interest, to create a superior model that could outperform the individual models (i.e., the RF, SVM, LR and NB).

We have designed two strategies to follow this concept based on the results obtained for each algorithm implemented above. Firstly, one of such strategies, which we will refer to as *best-of-all*, aims to detect the model that has worked best for a specific MoA. Next, the

second one will build this aforementioned superior model taking all the final probabilities (i.e., outputs) of the previously used algorithms as its inputs. Both methodologies consider those models that obtained the lowest log-loss when implementing each ML algorithm, which are:

- RF model with 200 estimators and 64 minimum samples per leaf without data partition.
- SVM model with RBF kernel without data partition.
- LR model with the 15 most important features without data partition.
- NB model with the 15 most important features without data partition.

On the one hand, the best-of-all strategy consisted of using the four models to predict the probabilities for the same datasets that we used to train the algorithms, as the first step. Then, once we have all these probabilities, we compute the log-loss scores individually for each MoA (i.e., each column of the targets dataset) instead of the general log-loss we had analysed before. In the end, we know the metric scores that each one of these four models obtained for each MoA prediction. The next step is to record which algorithm works best for each MoA (i.e., which model obtained the lowest log-loss) and, finally, we will use those recorded algorithms to predict the final probabilities using the test datasets. To summarise, we will predict each MoA with one of these four models we had; so, for the first mechanism, we may use the SVM, while for the second one, the RF could be the best option to define its predicted probabilities.

The *best-of-all* strategy scored **0,0154** - the log-loss turns out to be the second-best score after the SVM with the RBF kernel.

On the other hand, the second strategy, rigorously a model of models, will create new datasets with the final probabilities (for both training and test set) that each one of the models had assigned for each MoA, although we decided to eliminate the NB probabilities due to its poor performance. There will be as many probabilities datasets as mechanisms (i.e., 206), and they will act as the metamodel features. The targets, in this case, will be the column added to these datasets containing the association between the condition and the MoA (i.e., 0 or 1). Once all these datasets are ready, we apply an RF using the probabilities and targets for the training set. Later, the same RF will use the data from the test set to predict its final probabilities. The log-loss obtained with this approach has been **0,0794**, which is significantly high compared with the other results.

4.1.6 Summary

To conclude this first experimental part, Table 4 summarize the log-loss that the most representative algorithms scored, including the particularities of each implementation.

Since the SVM with the RBF kernel has scored the best log-loss, it is the model that stands out from the Off-The-Shelf algorithms in terms of performance. Taking advantage of having the individual log-loss for each MoA, we have analysed how the frequency of a MoA (i.e., the number of conditions that actually have that MoA associated to) affects this score. To do so, we have plotted the log-loss against the number of conditions associated to an MoA in Fig. 12. We can observe that, in general, the lowest log-loss scores correspond to those MoAs that are not frequent (i.e., MoAs that have fewer conditions associated), while higher scores correspond

Table 4. Summary of the log-loss values obtained by the Off-The-Shelf ML algorithms.

Algorithm	**Particularities**	**Log-loss**
Random Forest	Default parameters	0,0544
	64 minimum samples per leaf and 200 trees	0,0174
	Dose-time data partition	0,0187
Support Vector Machine	Linear kernel	0,0159
	Radial basis function kernel	0,0149
	Dose-time data partition	0,0166
Logistic Regression	Top 15 features	0,0166
	Top z-scored features	0,0167
	Dose-time data partition	0,0199
Naïve Bayes	Top 15 features	0,2000
	Top z-scored features	0,4094
	Dose-time data partition	0,1978
Metamodel	*Best-of-all* strategy	0,0154
	Probabilities as features	0,0794

to those MoAs that are very common among the conditions. However, some MoAs obtained a relatively low log-loss although having many conditions associated. The most extreme case is what happens with the MoA *proteasome inhibitor*, which scored an exceptionally low log-loss although being a common MoA among the conditions.

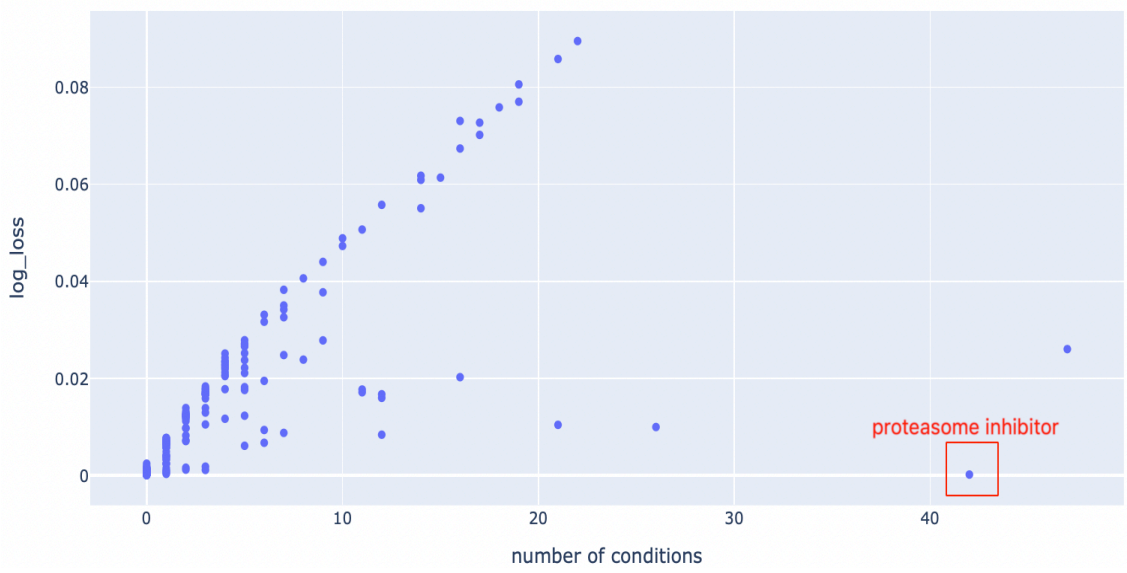


Figure 12. Log-loss scored by each mechanism of action with the support vector machine algorithm against the actual number of conditions each mechanism is associated with. The model used a radial basis function. The red box highlights the case of the mechanism *proteasome inhibitor*.

Additionally, given that the following algorithm we want to work with is the stochastic block model and we determined that we would only consider discrete gene expression and cell viability data, we wanted to assess the fact of only having this data and what would happen

with the discretisations that we discussed in 2.4. So, we have used the SVM for different data combinations: only considering gene expression, only cell viability, both cell and gene data and, finally, the latter, plus the treatment information (i.e., condition type, dose and treatment time). Moreover, we built SVMs for each combination twice to account for continuous and discrete genes expression and cell viability values, if considered for the model. Table 5 contains all the log-loss results for these cases. Since, in general, the discretisation did not damage the performance of the SVM considerably, except for some cases that we will discuss in the discussion section, we can continue with our purpose of creating an SBM.

Table 5. Summary of the log-loss values obtained by the SVM with different features selections, all of them assessed with both continuous and discrete values.

Particularities of the SVM	**Log-loss for continuous values (for genes and cells data)**	**Log-loss for discrete values (for genes and cells)**
Gene expression	0,0152	0,0182
Cell viability	0,0159	0,0186
Gene expression and cell viability	0,0149	0,114
Gene expression, cell viability and treatment information	0,0149	0,0175

4.2 Stochastic Block Model

The SBM will not work like the rest of the algorithms which have as many individual models as MoAs. Instead, the SBM will shape the data into a general graph of associations between each condition and their corresponding discrete values of genes, cells and MoAs (as seen in 4.2.2), without considering those features which stand for the treatment information. We must also forget about the idea of creating a model fitted to the training data to use it, finally, with the test dataset because from now on, the model will use both training and test datasets from the very first step.

We need to set several values for the SBM: the number of groups of conditions, MoAs, genes and cells (K, L, T, F, respectively) and the weight or importance we want to give to the gene expression and cell viability data (λ_g and λ_v, respectively) compared to the importance of the MoA annotations, among them. Since it is an iterative algorithm, it will also be crucial to decide when these iterations should stop. Next, we will discuss how we choose such values, based mainly on the usage of gene expression data, instead of both gene and cell data, to ease such picking. Once optimised, we will assess the effect of adding this cell viability data into the model.

In general, the SBM will consist of several runs; that is, we will run the code several times to get different results of the model's parameters (i.e., the membership vectors and the probability matrices) since the first assignation to the membership vectors is at random and can produce significant changes on their final values. Then, we will average the probabilities obtained by each run in order to get a final dataset of probabilities, from which we will calculate the log-loss. We must consider the time to be consumed for each run as a crucial aspect, besides the overall model's performance, since we need several runs to create the general SBM. Hence, we have to thoroughly analyse the number of groups and iterations, which will

also play an essential role in this computing time.

4.2.1 Only gene expression

4.2.1.1 Number of iterations choice

First of all, we want to decide how the iterative algorithm will know when to stop. Each iteration will involve the parameters of the models (i.e., the membership vectors and the probability matrices) to be better defined for our data, causing the log posterior to be higher (i.e., the probability of having that model given the data). So, the more iterations, the better the performance of the model. However, this algorithm is highly time-consuming, meaning that we should find a way to set the number of genuinely needed iterations for the model to work properly. We can assess this correct behaviour by analysing the evolution of the log posterior, which should stabilise at some point, or by checking the convergence of the parameters. Nonetheless, checking the log posterior difference or the parameter's convergence between successive iterations will not be optimal in terms of time.

For that reason, we decided to set a specific number of iterations for all the models to be created. To do so, we have examined such parameter's convergence and log posterior evolution for the cases of a single run with K=L=T=2, 3, 4 and 5. We have used a high number of iterations, 1500, to ensure the model's stabilisation. Fig. 13 depicts the evolution of the log posterior for all four values of K, L and T. The log posterior is constantly increasing, and we can confirm that 1500 iterations are far too many. Since all the steeper slopes happen before reaching the 200th iteration, we have decided that 500 iterations should be enough in general, considering that some cases (e.g., increasing the number of groups) could require more iterations to reach the model's stabilisation. Despite checking the average relative error of the parameters, see Fig. 14 for such evolution in the case of having K=L=T=3 groups, we can observe that the convergence does not allow us to set any rule that can be helpful to determine the number of iterations.

4.2.1.2 Weight of gene expression choice

Secondly, working with gene expression requires setting a weight (λ) to that data. The most important part of the SBM is the interactions between the conditions and the MoAs. However, the data standing for the gene expression (or/and cell viability) acts as the metadata that will help to shape all these interactions. The weight to assign to the metadata should be inversely proportional to its size compared to the primary data size (i.e., number of MoAs) so as to equally contribute to the log posterior [49]; so, if there are approximately 200 MoAs and 800 genes, the weight of the genes should be 1/4. In order to check this value to be working correctly with our SBM, we tested other values following the exact implementation: an SBM was built with ten runs, 500 iterations and K=L=T=4, only changing the λ_g values. These values of λ_g were 0,025, 0,25 and 2,5.

The log-loss obtained for each case of λ_g is:

- If λ_g is 0,025: **0,0210**.

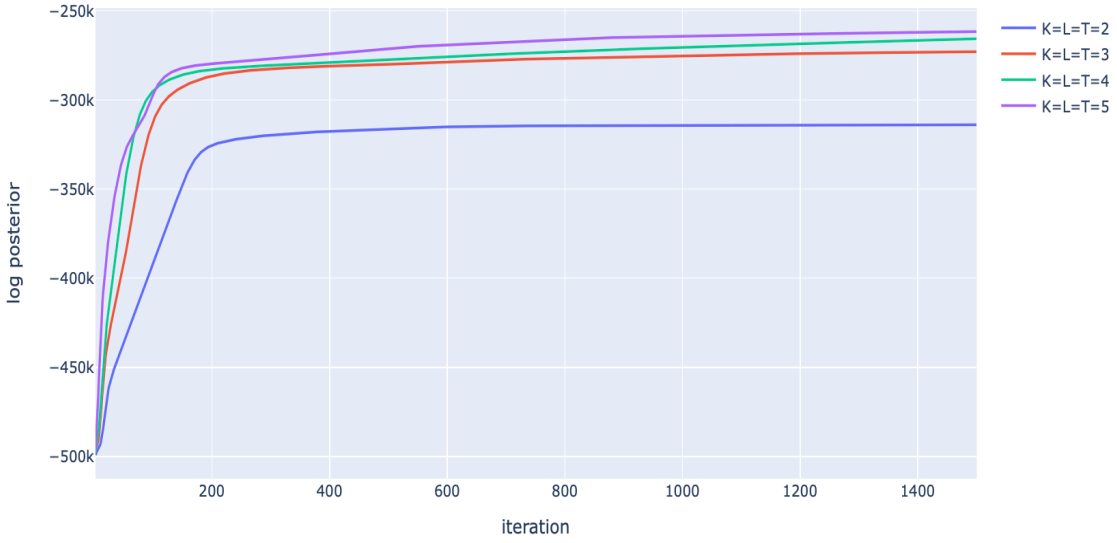


Figure 13. Log posterior evolution of the stochastic block models when setting K=L=T=2, 3, 4 and 5, respectively. Each line represents one of these cases.

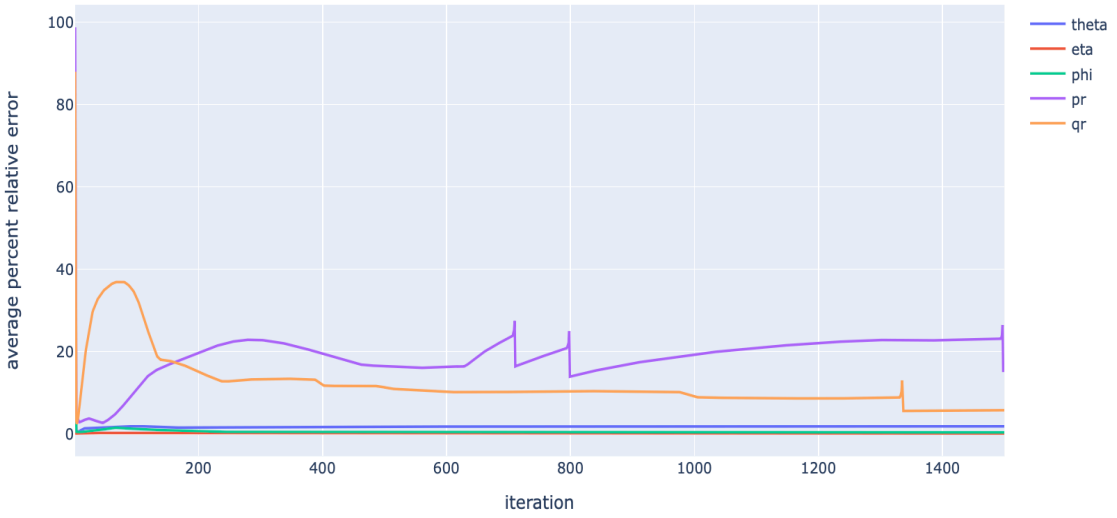


Figure 14. Convergence of the parameters evolution for the case of $K = L = T = 3$. The y axis represents the average relative error between two successive iterations as a percentage. Each line stands for a specific parameter of the model (i.e., including the membership vectors and the probability matrices).

- If λ_g is 0,25: **0,0204**.
- If λ_g is 2,5: **0,0205**.

Despite the closeness of these results, we still can corroborate the usage of a gene expression

weight of 0,25 since it scored the lowest log-loss.

4.2.1.3 Number of groups choice

Last but not least, we must set the number of groups of conditions, MoAs and genes (i.e., K, L and T). To determine the number of genes groups or T, we concluded from the gene expression heatmap (Fig. 2) that there were around ten sets of similar genes, as mentioned in 3.3. Since the algorithm we use is a mixed membership SBM and, then, one gene will not be associated with a single group, but it can be part of all these groups with a different probability of involvement (described in the membership vector), the model does not require these ten groups. Instead, we will set T=5 since we could easily separate ten items with five dimensions, which would account for each dimension of the membership vectors of the genes.

Up to now, we had been working with K=L=T. If T=5, then the number of groups of conditions and MoAs (i.e., K and L, respectively) should be equal to 5 as well. However, the model's performance would be very similar to the case in 4.2.1.2 (with K=L=T=4), which was higher than the log-loss obtained by many Off-The-Shelf algorithms. Therefore, we decided to raise K and L to 10 and 20 for both. Finally, we built two models with such combinations, considering λ_g=0,25, with 500 iterations and 100 runs for each model. The log-loss scores computed for both cases are the following:

- K=L=10 and T=5: **0,0198**.
- K=L=20 and T=5: **0,0196**.

The performance of the SBM has reached its maximum with K=L=20 and T=5. Nevertheless, it has not exceeded many of the first part algorithms.

4.2.2 Both gene expression and cell viability

Adding cell viability data will cause a notorious increment of time used to build the SBM. Since this section aims to analyse whether adding this data will have a positive or negative effect, comparing the performance of the models with and without it, we can decrease the values of K, L, T and F, which corresponds to the number of groups of cells. As long as both models work with the same values, the values assigned do not matter. The weight of the cell viability data (λ_v) should be 2 since there are 100 cells and approximately 200 MoAs. So, we will build two SBMs with K=L=T=F=2, λ_g=0,25, λ_v=2, 800 iterations so as to give the parameters more time to converge, and 10 runs for each model. Finally, the log-loss scores obtained with each model are:

- Only gene expression: **0,0206**.
- Both gene expression and cell viability: **0,0209**.

Since adding cell viability data did not cause the log-loss to decrease, we will not build any other SBM with both cells and genes data.

4.2.3 Summary and interpretation of the SBM

To sum up, Table 6 gathers all the log-loss results of the SBMs implemented above for different cases, including only gene expression data or both gene expression and cell viability data. All the SBMs depicted in Table 6 used λ_g=0,25 and λ_v=2, in case of including the latter data.

Table 6. Summary of the log-loss values obtained by the SBMs. In all cases, the weight of the parameters has been fixed for λ_g=0,25 and λ_v=2 (in case of considering cell viability data).

Particularities of the SBM	Log-loss
K=L=T=4 (genes)	0,0204
K=L=10 and T=5 (genes)	0,0198
K=L=20 and T=5 (genes)	0,0196
K=L=T=2 (genes)	0,0206
K=L=T=F=2 (genes and cells)	0,0209

The best performance is for the SBM with K=L=20 and T=5, without including cell viability data. As in 4.1.6, we want to compare the individual log-loss obtained with the SBM for each MoA against the number of conditions that MoA is actually associated with. In Fig. 15, we can observe that the log-loss scores also follows the tendency to be higher when the number of conditions is greater, although there are two MoAs, *proteasome inhibitor* among them, that break this rule since they are frequent mechanisms and the log-loss scores are lower than expected. Besides, we have noticed that, overall, the probabilities that the SBM assigns are significantly low. For instance, the maximum probability for the *proteasome inhibitor* is 0,5.

Even though the SBM did not outperform most of the algorithms used in the first experimental part (i.e., Off-The-Shelf algorithms, including SVM, RF and LR), SBMs are highly interpretable, and besides the results of the log-loss, we can analyse it in other ways. The SBM aimed to create communities that could explain the association between conditions treated with a specific drug with the MoA they finally had. Moreover, we can depict the model's parameters, including membership vectors and probability matrices, to understand how the model works. Following, we will analyse the final parameters of the SBM without cell viability data and with K=L=10 and T=5, since its log-loss (see Table 6) was similar to the case of K=L=20 and T=5, and the graphical representation of its parameters will be more straightforward.

On the one hand, the MoA each combination has annotated in the original dataset could influence how the SBM finally groups these conditions through their membership vectors (θ). To validate this idea, we used a principal component analysis (PCA) with all these vectors to detect clusters of conditions based on these original MoAs. The top principal components had an explained variance of 28%, 23% and 20% (for PC1, PC2 and PC3, respectively).

The first impression is that if a MoA is easy to predict, if it scores a low log-loss, the PCA should cluster the conditions clearly based on the mechanisms they actually have, whereas if the MoA was difficult to predict, it should be more challenging to detect such clusters. To verify it, Fig. 16 and 17 depicts the PCA analysis of the conditions membership

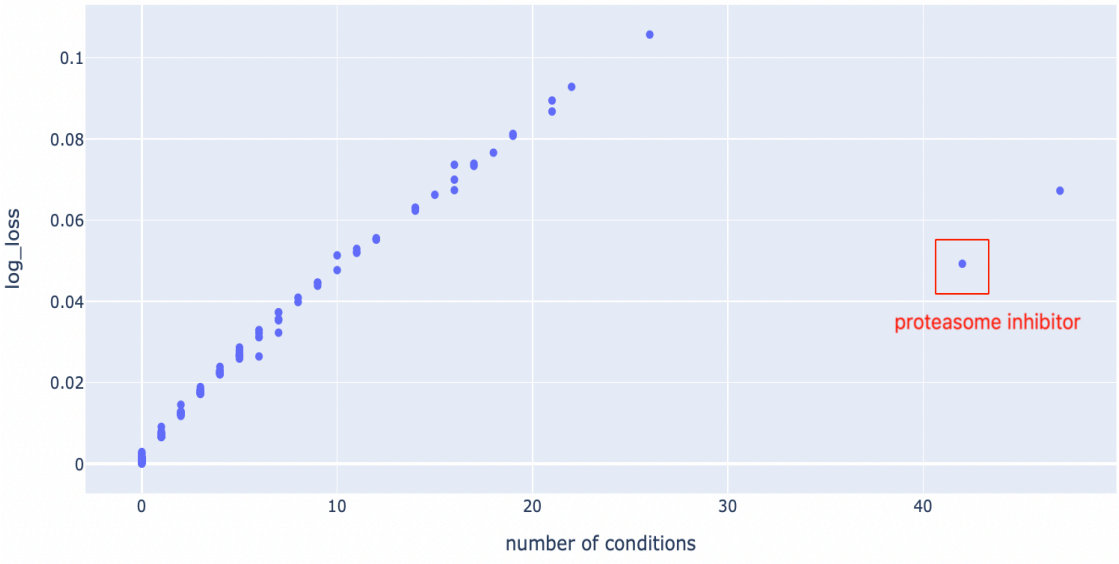


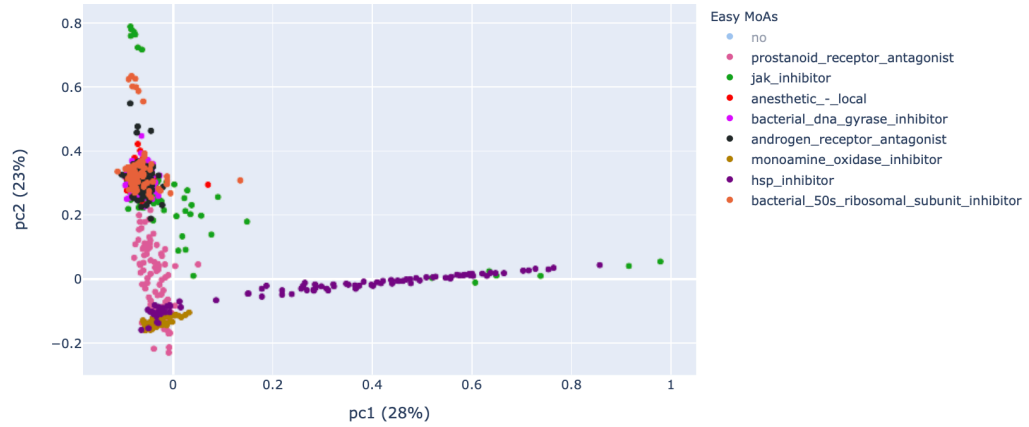
Figure 15. Log-loss scored by each mechanism of action with the stochastic block model against the actual number of conditions the mechanism are associated with. The model has been created with gene expression data and the values set to K=L=20 and T=5. The red box highlights the case of the mechanism of action *proteasome inhibitor*.

vectors, colouring each point according to easy MoAs (Fig. 16) and difficult MoAs (Fig. 17) original association. The differences between both set of figures can not corroborate the aforementioned hypothesis, since in both cases, many of the MoAs are not well defined with an independent cluster of conditions; instead, there are a lot of conditions with different MoAs in the same place of the planes.

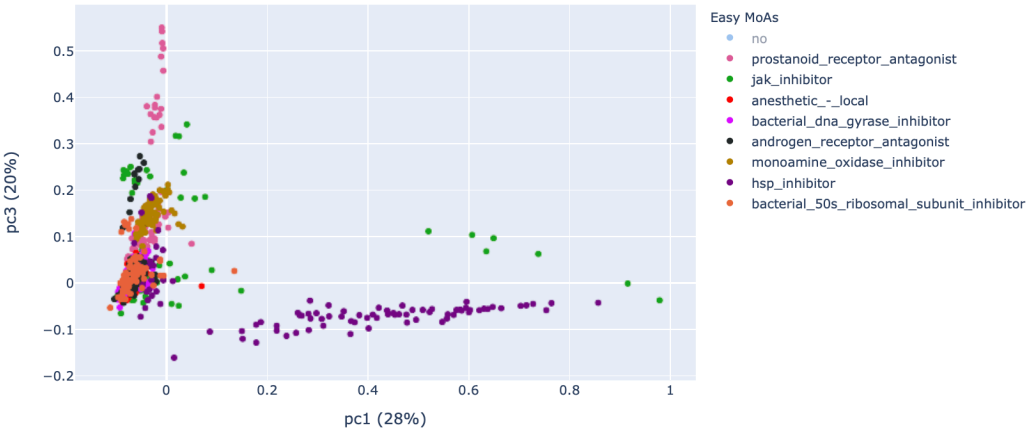
On the other hand, in order to understand the interactions between groups of conditions and MoAs and between groups of conditions and genes, we can graphically visualize the probability matrices. Given the probability matrix **p** (i.e., the matrix that links groups of conditions and MoAs), in Fig. 18, we can observe that there are just a few combinations of conditions and MoAs groups that have a probability higher than the rest of being positively associated, with the link 1. From these, there is a combination that stands out, which is the one that combines the 8th conditions group with the 1st MoA group. Interestingly, we noticed that *proteasome inhibitor*, which is the second most common MoA and the one that still scored a relatively low log-loss, has a probability close to one of belonging to the 1st MoA group, which concentrates many positive labels when linked with the 8th group of conditions.

Finally, we want to analyse the probability matrix for conditions and genes (q). Fig. 19 shows q, including the three dimensions of the matrix that correspond to the three different discrete values for gene expression data (-1 for under-expressed, 0 for normally-expressed and 1 for over-expressed). It demonstrates a considerable amount of gene groups that have a high probability of being associated with conditions for being normally-expressed (i.e., high probability of link 0). Therefore, the SBM makes it notorious that most of the conditions have a standard gene expression signature. Nevertheless, the 8th group of conditions is actually the group that holds high probabilities of having under-expressed and over-expressed genes

among their signatures. Fig. 19.A shows a high probability of having a link -1 when connected with the 0th genes group, while Fig.19.C, a high probability of having a link 1 when associated with the 3rd group of genes.



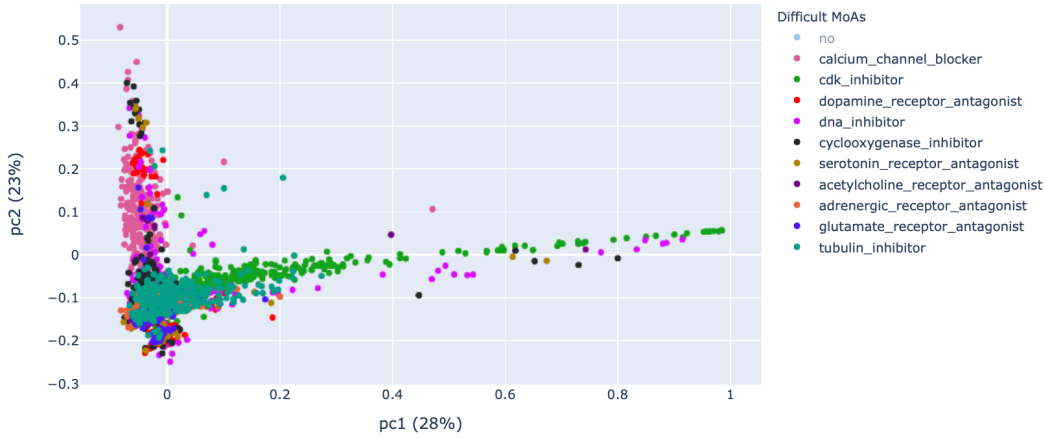
(a) PC1 vs PC2



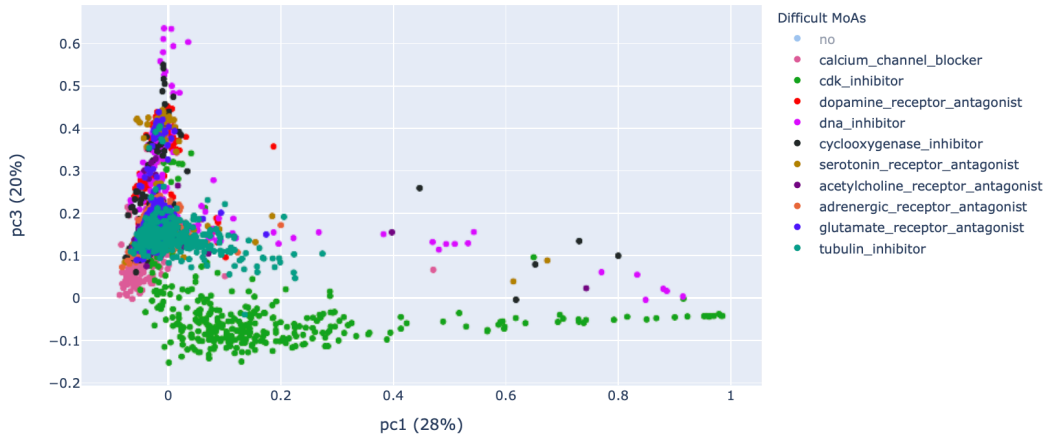
(b) PC1 vs PC3

(c) PC2 vs PC3

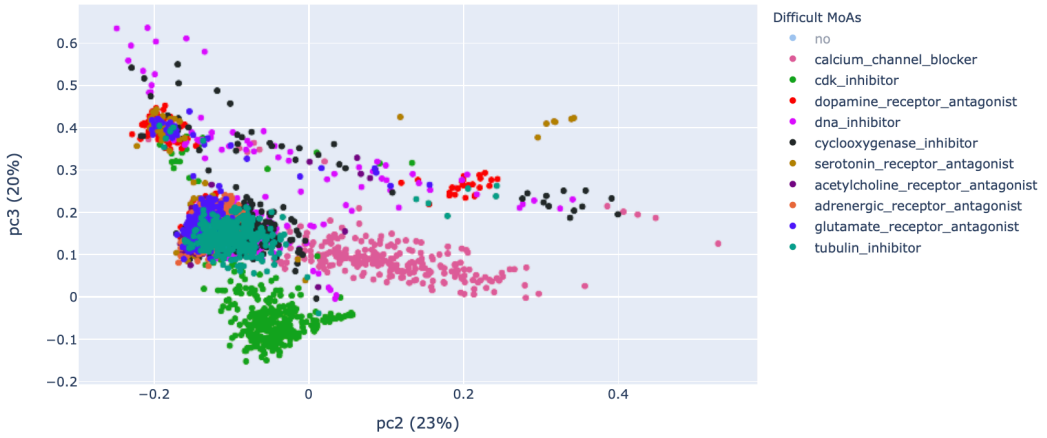
Figure 16. PCA analysis for the study of mechanism of action based on the membership vectors of the conditions. The coloured points account for those mechanisms that have between 80 and 95 conditions of the whole dataset associated with, which still scored a low log-loss (i.e., easy to predict mechanisms).



(a) PC1 vs PC2



(b) PC1 vs PC3



(c) PC2 vs PC3

Figure 17. PCA analysis for the study of mechanisms of action based on the membership vectors of the conditions. The coloured points account for the eight mechanisms that scored the worst log-loss (i.e., difficult to predict).

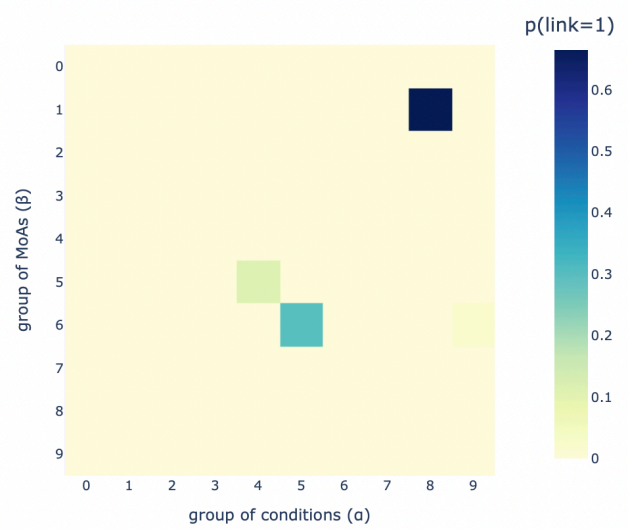
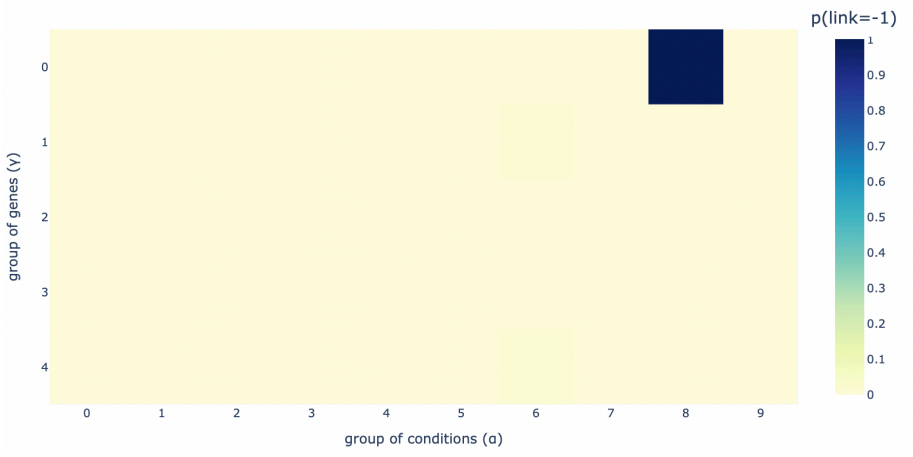
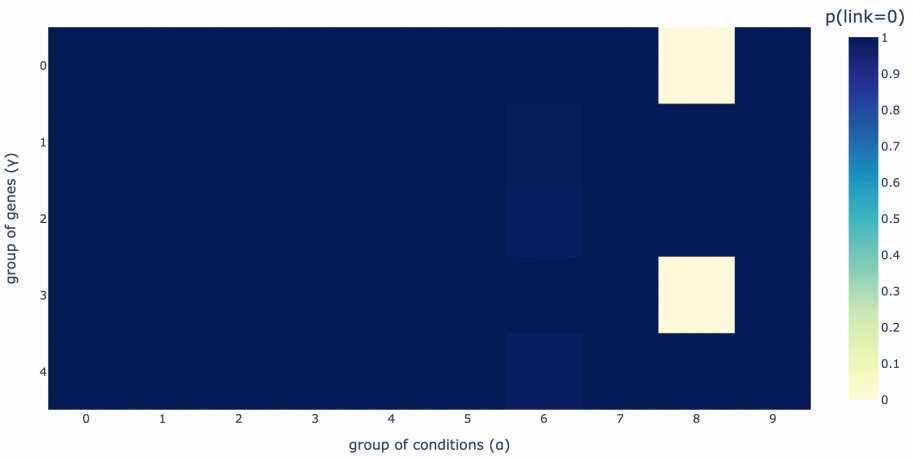


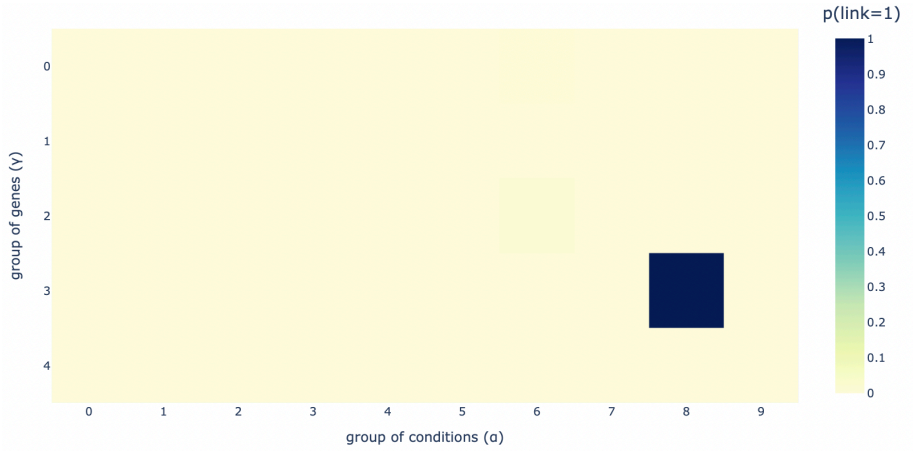
Figure 18. Probability matrix of conditions and mechanisms of action groups (p). The values correspond to the probability of a condition of a group α being linked with a 1 with a mechanism of group β.



(a) Probability of link -1.



(b) Probability of link 0.



(c) Probability of link 1.

Figure 19. Probability matrix of conditions and genes groups (q). The values correspond to the probability of a condition of a group α being linked with the value specified in each subcaption with a gene of group γ.

5 Discussion

This section will take all the results together to discuss what we found to be more significant along the experimental workflow and the limitations of our proposed methodology.

According to the algorithms selected Off-The-Shelf, we have compared the performance between different algorithms and how they are affected by their respective parameters. Overall, selecting the optimal parameters that particularise each method plays an important role, especially in RFs and SVMs. In the case of RF, we could demonstrate the importance of the number of trees or estimators used to build the forest and the minimum samples a leaf must have to continue splicing the features space. The model's performance went from a log-loss score of 0,0544 to 0,0174, just by changing such parameters. Similarly, the SVM was affected by the kernel parameter, which is responsible for the shape used to split the features space as well. A linear kernel, or boundary, is not sufficient to represent the complexity of the data. Instead, using an RBF kernel offered additional flexibility needed to shape this data.

The LR worked incredibly fine, although removing a meaningful amount of features, from 875 to 15. Its log-loss was still comparable to the RFs' and SVMs' performance, making it a potential algorithm to consider for this kind of data. Nevertheless, the NB, which also considered this reduced representation of features, became the worst model for predicting MoAs, scoring a log-loss that was neither in the same magnitude order as the rest. Such a result reassures the fact of the NB being a bad estimator [47].

Creating metamodels based on the results obtained from these algorithms we have mentioned above did not result in a better performance. The reason for the *best-of-all* strategy not outperforming the SVM is that the selection of the best model to predict a particular MoA can be different for the training and the test set. On the other hand, the relatively high value of the log-loss scored by the metamodel that considered the probabilities resulting from the other models (i.e., RF, SVM and LR) as its features suggests that the underlying probabilities and, in turn, the submodels might not be the optimal ones.

To conclude with this first discussion of the models' performance, we want to highlight that, surprisingly, the SBM was not significantly affected by the values that characterize how the model represents the complex data (i.e., number of groups of conditions, MoAs, genes and cells, as well as the weight or importance assigned to gene expression and cell viability). No matter how hard we tried to make it better by incrementing the number of groups, the log-loss was always around 0,02, albeit we could detect slight decreases. Given the results of the interpretation of the SBM, in 4.2.3, we could also detect that ten groups of conditions and MoAs resulted in highly sparse probability matrices, suggesting that fewer groups are needed. These matrices, especially Fig. 20, also insinuate that the discretisation we used for gene expression does not work well with this model due to this results' spareness.

Regarding the importance of the features to predict the MoAs, we can conclude that partitioning the data based on time-dosage combinations is not worthy. Woo et al. [31] also worked with gene expression data coming from different time points experiments and noticed that integrating these signatures over all time points gave better results than looking at the gene expression of the optimal time point, which strengthen our finding. Other experiments that corroborate this fact is that (1) when assessing feature importance for the RF, these features accounting for experiments information were not among the top features, and (2) when we compared the SVM considering only gene and cells features with the SVM considering

genes, cells and experimental data, the log-loss obtained by both models were alike.

Moreover, focusing on gene expression and cell viability data, we have detected other interesting relationships concerning their importance when predicting the mechanisms. Even though the results of assessing feature importance for the RF determined that most of the critical variables to make the predictions are those associated with gene expression, other results confirm that this is not always the case. When checking the effect of the features selection in SVMs (Table 5), the log-loss scores demonstrate that having only gene expression features is similar to having only cell viability data, no matter if they were discretised or not. However, when only gathering discrete gene and cell data, the SVM performed notably poorly compared to the other cases, suggesting that this melange of data can be confounding for predicting the MoA of a drug. This last statement was also noticed by Szalai et al. [36], who mentioned there was a risky correlation between cell death and gene signatures that could damage these predictions, although in our case, it did not happen if the data was continuous. In addition, this confounding factor did not highly affect the SBM since the score obtained when adding cell viability to the model was similar to not having it.

Once we discussed the models' performance and the importance of the features, we want to emphasise the most significant limitation of the project: the metrics used to validate the algorithms, the logistic loss. We have been using it because it was how the *Kaggle* competition evaluated the probabilities given by the participants, and we naïvely thought it was the best option to assess such a sparse problem. We made some submissions to *Kaggle* during the competition period to check if our scores were similar to what other groups had submitted, and we were close to the winners. So, this limitation could be a general issue for the competition overall. We have demonstrated that the log-loss scores grew linearly with the frequency of the MoA (i.e., number of associated conditions or number of 1s), except for some mechanisms. To illustrate the problem this represents, we could imagine having a model which only predicts extremely low probabilities, which slightly happens with SBM. If most of the MoAs included in the data are uncommon, the target dataset columns would have much more zeros than ones. When computing the log-loss, most of the times, we would be comparing a low probability with a zero, which would be beneficial for the log-loss, and just a few times, the probability would be compared with what should be a one. This latter case would raise the log-loss, but it would not penalise that much due to the low number of ones in the dataset. Therefore, we can believe that the logistic loss metric is not robust enough for this problem.

Finally, aiming to understand why some MoAs did not follow the rule we noticed between the log-loss and the prevalence of the mechanism in the dataset, we analysed what happened with the *proteasome inhibitor*, which in both SVM and SBM scored a better log-loss than expected given the large number of conditions it had annotated. Even though the SVM is not highly interpretable, the probabilities given by the model for those conditions that should have the MoA differ significantly from the probabilities assigned to conditions that should not have the MoA; the former probabilities are almost one, whereas the latter values are close to zero, making it a well-classified mechanism. On the other hand, we can not observe such differences with the SBM due to the low probabilities it assigns. However, we can demonstrate that even though the SBM may not work well to predict the MoA, it can represent the data correctly and give insight into the problem. To exemplify it with the same MoA, we noted that the *proteasome inhibitor* belongs with a high probability to the 1st group of MoAs, which is, in fact, the group of mechanisms that is densely connected to a group of conditions (Fig.

18). This group of conditions, the 8th, happens to be the only group containing most of the overexpressed and underexpressed genes (Fig. 19). In a nutshell, this interpretation suggests that the *proteasome inhibitor*, among other mechanisms belonging to that MoA group, is relatively easy to predict because the conditions that have that mechanism are the ones that have a gene signature more uncommon.

6 Conclusions

The project's main objective was to find the best model that could predict the MoAs of any drug, given experimental data from gene expression and cell viability after the treatment of several cells. We have followed a workflow that has allowed us to build and compare different ML algorithms that we thought to be potential for this purpose. We have worked with widespread methods (including random forests, support vector machines, logistic regressions and naïve Bayes classifiers) and a stochastic block model. We have accomplished our principal aim; we can determine that for this project and, among the models we have used, the SVM is the model that could better predict the MoAs.

We also wanted to assess other underlying aims involving, for example, the importance of the features to predict these MoAs. We could determine that partitioning the data depending on treatment time and dosage combinations is not worthy. And, most importantly, that gene expression data is the main responsible for the MoAs the model will predict. However, cell viability, if considered alone, also plays an important role.

We could develop the SBM with gene expression and cell viability data. However, it did not stand out in terms of predictive ability, although we could demonstrate its potential to interpret the data and its interactions, which is also essential for drug design.

Even though not being one of our aims, we can conclude that the log-loss metric is not the optimal way to validate the predictions for this problem. It is not robust enough for such sparse data, and, in turn, it became our principal limitation.

Finally, we would like to suggest future work that could follow this research. First of all, it is vital to look for other metrics to assess the model's performance better. This metric could include penalties for MoAs that are not common among the conditions (i.e., columns with many zeros) since we detected a risky correlation between the log-loss and the commonness of the MoA. Given that there is still room for improvement with the SBM - the final probability matrices were too sparse - we think it would be helpful to pose again two questions: (1) how genes should be discretised and (2) how many groups should the SBM has. This discretisation might need more than three final labels to consider more gene expression states, other than over-, normally- and under-expressed; and, more combinations of number of groups could be tested to find the most optimal choice.

7 Bibliography

[1] J. Drews, "Drug discovery: A historical perspective," *Science*, vol. 287, no. 5460, pp. 1960–1964, 2000, ISSN: 00368075. DOI: 10.1126/science.287.5460.1960.

[2] R. L. Davis, "Mechanism of Action and Target Identification: A Matter of Timing in Drug Discovery," *iScience*, vol. 23, no. 9, pp. 1–6, 2020, ISSN: 25890042. DOI: 10.1016/j.isci.2020.101487.

[3] D. C. Swinney, "Drug discoveries and molecular mechanism of action," in *Successful Drug Discovery*, J. Fischer and D. P. Rotella, Eds., 1st, Wiley-VCH Verlag GmbH & Co. KGaA, 2015, ch. Drug disco, pp. 19–34, ISBN: 9783527336852. DOI: 10.1002/9783527678433.ch2.

[4] M. Gittelman, "The revolution re-visited: Clinical and genetics research paradigms and the productivity paradox in drug discovery," *Research Policy*, vol. 45, no. 8, pp. 1570–1585, 2016, ISSN: 00487333. DOI: 10.1016/j.respol.2016.01.007. [Online]. Available: http://dx.doi.org/10.1016/j.respol.2016.01.007.

[5] I. Khanna, "Drug discovery in pharmaceutical industry: Productivity challenges and trends," *Drug Discovery Today*, vol. 17, no. 19-20, pp. 1088–1102, 2012, ISSN: 13596446. DOI: 10.1016/j.drudis.2012.05.007. [Online]. Available: http://dx.doi.org/10.1016/j.drudis.2012.05.007.

[6] J. W. Scannell, A. Blanckley, H. Boldon, and B. Warrington, "Diagnosing the decline in pharmaceutical R&D efficiency," *Nature Reviews Drug Discovery*, vol. 11, no. 3, pp. 191–200, 2012, ISSN: 14741776. DOI: 10.1038/nrd3681. [Online]. Available: http://dx.doi.org/10.1038/nrd3681.

[7] C. T. Caskey, "The drug development crisis: Efficiency and safety," *Annual Review of Medicine*, vol. 58, pp. 1–16, 2007, ISSN: 00664219. DOI: 10.1146/annurev.med.58.042705.124037.

[8] J. S. Bryans, C. A. Kettleborough, and R. Solari, "Are academic drug discovery efforts receiving more recognition with declining industry efficiency?" *Expert Opinion on Drug Discovery*, vol. 14, no. 7, pp. 605–607, 2019, ISSN: 1746045X. DOI: 10.1080/17460441.2019.1596080. [Online]. Available: https://doi.org/10.1080/17460441.2019.1596080.

[9] A. Mullard, "2018 FDA drug approvals," *Nature Reviews Drug Discovery*, vol. 18, no. February, pp. 85–89, 2019. DOI: 10.1038/d41573-019-00014-x.

[10] ——, "2019 FDA drug approvals," *Nature Reviews Drug Discovery*, vol. 19, no. January, pp. 79–84, 2020. DOI: 10.1038/d41573-020-00001-7.

[11] ——, "2020 FDA drug approvals," *Nature Reviews Drug Discovery*, vol. 20, no. January, pp. 85–90, 2021. DOI: 0.1038/d41573-021-00002-0.

[12] S. M. Paul, D. S. Mytelka, C. T. Dunwiddie, C. C. Persinger, B. H. Munos, S. R. Lindborg, and A. L. Schacht, "How to improve RD productivity: The pharmaceutical industry's grand challenge," *Nature Reviews Drug Discovery*, vol. 9, no. 3, pp. 203–214, 2010, ISSN: 14741776. DOI: 10.1038/nrd3078.

[13] M. Wehling, "Assessing the translatability of drug projects: What needs to be scored to predict success?" *Nature Reviews Drug Discovery*, vol. 8, no. 7, pp. 541–546, 2009, ISSN: 14741776. DOI: 10.1038/nrd2898. [Online]. Available: http://dx.doi.org/10.1038/nrd2898.

[14] R. K. Harrison, "Phase II and phase III failures: 2013-2015," *Nature Reviews Drug Discovery*, vol. 15, no. 12, pp. 817–818, 2016, ISSN: 14741784. DOI: 10.1038/nrd.2016.184. [Online]. Available: http://dx.doi.org/10.1038/nrd.2016.184.

[15] J. Arie Vonk, R. Benigni, M. Hewitt, M. Nendza, H. Segner, D. Van De Meent, and M. T. Cronin, "The use of mechanisms and modes of toxic action in integrated testing strategies: The report and recommendations of a workshop held as part of the European union OSIRIS integrated project," *ATLA Alternatives to Laboratory Animals*, vol. 37, no. 5, pp. 557–571, 2009, ISSN: 02611929. DOI: 10.1177/026119290903700512.

[16] D. C. Swinney and J. Anthony, "How were new medicines discovered?" *Nature Reviews Drug Discovery*, vol. 10, no. 7, pp. 507–519, 2011, ISSN: 14741776. DOI: 10.1038/nrd3480.

[17] M. Schenone, V. Dančík, B. K. Wagner, and P. A. Clemons, "Target identification and mechanism of action in chemical biology and drug discovery," *Nature Chemical Biology*, vol. 9, no. 4, pp. 232–240, 2013, ISSN: 15524450. DOI: 10.1038/nchembio.1199.

[18] L. B. Tulloch, S. K. Menzies, R. P. Coron, M. D. Roberts, G. J. Florence, and T. K. Smith, "Direct and indirect approaches to identify drug modes of action," *IUBMB Life*, vol. 70, no. 1, pp. 9–22, 2018, ISSN: 15216551. DOI: 10.1002/iub.1697.

[19] C. Réda, E. Kaufmann, and A. Delahaye-Duriez, "Machine learning applications in drug development," *Computational and Structural Biotechnology Journal*, vol. 18, no. December, pp. 241–252, 2020, ISSN: 20010370. DOI: 10.1016/j.csbj.2019.12.006.

[20] J. Vamathevan, D. Clark, P. Czodrowski, I. Dunham, E. Ferran, G. Lee, B. Li, A. Madabhushi, P. Shah, M. Spitzer, and S. Zhao, "Applications of machine learning in drug discovery and development," *Nature Reviews Drug Discovery*, vol. 18, no. 6, pp. 463–477, 2019, ISSN: 14741784. DOI: 10.1038/s41573-019-0024-5. [Online]. Available: http://dx.doi.org/10.1038/s41573-019-0024-5.

[21] J. W. Cassidy, "Applications of Machine Learning in Drug Discovery I: Target Discovery and Small Molecule Drug Design," in *Artificial Intelligence in Oncology Drug Discovery and Development*, J. W. Cassidy and B. Taylor, Eds., Rijeka: IntechOpen, 2020, ch. 2. DOI: 10.5772/intechopen.93159. [Online]. Available: https://doi.org/10.5772/intechopen.93159.

[22] L. Patel, T. Shukla, X. Huang, D. W. Ussery, and S. Wang, "Machine Learning Methods in Drug Discovery," *Molecules*, vol. 25, no. 22, 2020, ISSN: 14203049. DOI: 10.3390/molecules25225277.

[23] Deloitte Centre for Health Solutions, "Unlocking R&D productivity: Measuring the return from pharmaceutical innovation 2018," Tech. Rep., 2018, pp. 1–32. [Online]. Available: https://heatinformatics.com/sites/default/files/images-videosFileContent/deloitte-uk-measuring-return-on-pharma-innovation-report-2018.pdf.

[24] Q. Li and C. Kang, "Mechanisms of action for small molecules revealed by structural biology in drug discovery," *International Journal of Molecular Sciences*, vol. 21, no. 15, pp. 1–18, 2020, ISSN: 14220067. DOI: 10.3390/ijms21155262.

[25] Z. E. Perlman, M. D. Slack, Y. Feng, T. J. Mitchison, L. F. Wu, and S. J. Altschuler, "Multidimensional drug profiling by automated microscopy," *Science*, vol. 306, no. 5699, pp. 1194–1198, 2004, ISSN: 00368075. DOI: 10.1126/science.1100709.

[26] D. W. Young, A. Bender, J. Hoyt, E. McWhinnie, G. W. Chirn, C. Y. Tao, J. A. Tallarico, M. Labow, J. L. Jenkins, T. J. Mitchison, and Y. Feng, "Integrating high-content screening and ligand-target prediction to identify mechanism of action," *Nature Chemical Biology*, vol. 4, no. 1, pp. 59–68, 2008, ISSN: 15524469. DOI: 10.1038/nchembio.2007.53.

[27] F. Reisen, A. Sauty De Chalon, M. Pfeifer, X. Zhang, D. Gabriel, and P. Selzer, "Linking Phenotypes and Modes of Action Through High-Content Screen Fingerprints," *Assay and Drug Development Technologies*, vol. 13, no. 7, pp. 415–427, 2015, ISSN: 15578127. DOI: 10.1089/adt.2015.656.

[28] M. Hofmarcher, E. Rumetshofer, D. A. Clevert, S. Hochreiter, and G. Klambauer, "Accurate Prediction of Biological Assays with High-Throughput Microscopy Images and Convolutional Networks," *Journal of Chemical Information and Modeling*, vol. 59, no. 3, pp. 1163–1171, 2019, ISSN: 15205142. DOI: 10.1021/acs.jcim.8b00670.

[29] F. Iorio, R. Bosotti, E. Scacheri, V. Belcastro, P. Mithbaokar, R. Ferriero, L. Murino, R. Tagliaferri, N. Brunetti-Pierri, A. Isacchi, and D. Di Bernardo, "Discovery of drug mode of action and drug repositioning from transcriptional responses," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 107, no. 33, pp. 14 621–14 626, 2010, ISSN: 00278424. DOI: 10.1073/pnas.1000138107.

[30] A. C. Ravindranath, N. Perualila-Tan, A. Kasim, G. Drakakis, S. Liggi, S. C. Brewerton, D. Mason, M. J. Bodkin, D. A. Evans, A. Bhagwat, W. Talloen, H. W. Göhlmann, Z. Shkedy, and A. Bender, "Connecting gene expression data from connectivity map and in silico target predictions for small molecule mechanism-of-action analysis," *Molecular BioSystems*, vol. 11, no. 1, pp. 86–96, 2015, ISSN: 17422051. DOI: 10.1039/c4mb00328d.

[31] J. H. Woo, Y. Shimoni, W. S. Yang, P. Subramaniam, A. Iyer, P. Nicoletti, M. Rodríguez Martínez, G. López, M. Mattioli, R. Realubit, C. Karan, B. R. Stockwell, M. Bansal, and A. Califano, "Elucidating Compound Mechanism of Action by Network Perturbation Analysis," *Cell*, vol. 162, no. 2, pp. 441–451, 2015, ISSN: 10974172. DOI: 10.1016/j.cell.2015.05.056.

[32] K. Zhao and H.-C. So, "Drug Repositioning for Schizophrenia and Depression/Anxiety Disorders: A Machine Learning Approach Leveraging Expression Data.," eng, *IEEE journal of biomedical and health informatics*, vol. 23, no. 3, pp. 1304–1315, May 2019, ISSN: 2168-2208 (Electronic). DOI: 10.1109/JBHI.2018.2856535.

[33] E. Gonçalves, A. Segura□Cabrera, C. Pacini, G. Picco, F. M. Behan, P. Jaaks, E. A. Coker, D. Meer, A. Barthorpe, H. Lightfoot, T. Mironenko, A. Beck, L. Richardson, W. Yang, E. Lleshi, J. Hall, C. Tolley, C. Hall, I. Mali, F. Thomas, J. Morris, A. R. Leach, J. T. Lynch, B. Sidders, C. Crafter, F. Iorio, S. Fawell, and M. J. Garnett, " Drug mechanism□of□action discovery through the integration of pharmacological and CRISPR screens," *Molecular Systems Biology*, vol. 16, no. 7, pp. 1–14, 2020, ISSN: 1744-4292. DOI: 10.15252/msb.20199405.

[34] J. Lamb, "The Connectivity Map: A new tool for biomedical research," *Nature Reviews Cancer*, vol. 7, no. 1, pp. 54–60, 2007, ISSN: 1474175X. DOI: 10.1038/nrc2044.

[35] A. Musa, L. S. Ghoraie, S. D. Zhang, G. Glazko, O. Yli-Harja, M. Dehmer, B. Haibe-Kains, and F. Emmert-Streib, "A review of connectivity map and computational approaches in pharmacogenomics," *Briefings in Bioinformatics*, vol. 19, no. 3, pp. 506–523, 2018, ISSN: 14774054. DOI: 10.1093/bib/bbw112.

[36] B. Szalai, V. Subramanian, C. H. Holland, R. Alföldi, L. G. Puskás, and J. Saez-Rodriguez, "Signatures of cell death and proliferation in perturbation transcriptomics data - from confounding factor to effective prediction," *Nucleic Acids Research*, vol. 47, no. 19, pp. 10 010–10 026, 2019, ISSN: 13624962. DOI: 10.1093/nar/gkz805.

[37] A. Subramanian, R. Narayan, S. M. Corsello, D. D. Peck, T. E. Natoli, X. Lu, J. Gould, J. F. Davis, A. A. Tubelli, J. K. Asiedu, D. L. Lahr, J. E. Hirschman, Z. Liu, M. Donahue, B. Julian, M. Khan, D. Wadden, I. C. Smith, D. Lam, A. Liberzon, C. Toder, M. Bagul, M. Orzechowski, O. M. Enache, F. Piccioni, S. A. Johnson, N. J. Lyons, A. H. Berger, A. F. Shamji, A. N. Brooks, A. Vrcic, C. Flynn, J. Rosains, D. Y. Takeda, R. Hu, D. Davison, J. Lamb, K. Ardlie, L. Hogstrom, P. Greenside, N. S. Gray, P. A. Clemons, S. Silver, X. Wu, W. N. Zhao, W. Read-Button, X. Wu, S. J. Haggarty, L. V. Ronco, J. S. Boehm, S. L. Schreiber, J. G. Doench, J. A. Bittker, D. E. Root, B. Wong, and T. R. Golub, "A Next Generation Connectivity Map: L1000 Platform and the First 1,000,000 Profiles," *Cell*, vol. 171, no. 6, pp. 1437–1452, 2017, ISSN: 10974172. DOI: 10.1016/j.cell.2017.10.049.

[38] Kaggle, *Mechanisms of Action (MoA) prediction*, 2020. [Online]. Available: https://www.kaggle.com/c/lish-moa/overview (visited on 05/11/2021).

[39] Broad Institute, *Connectivity Map*, 2021. [Online]. Available: https://clue.io (visited on 05/11/2021).

[40] The President and Fellows of Harvard College, *Laboratory for Innovation Science at Harvard*, 2021. [Online]. Available: https://lish.harvard.edu (visited on 05/11/2021).

[41] LINCS Program, *NIH LINCS Program*, 2019. [Online]. Available: https://lincsproject.org (visited on 05/11/2021).

[42] M. B. Eise, P. T. Spellman, P. O. Brown, and D. Botstein, "Cluster analysis and display of genome-wide expression patterns," *Proc. Natl Acad. Sci. USA*, vol. 95, no. 22, pp. 14 863–14 868, 1998.

[43] A. Subramanian, P. Tamayo, V. K. Mootha, S. Mukherjee, B. L. Ebert, M. A. Gillette, A. Paulovich, S. L. Pomeroy, T. R. Golub, E. S. Lander, and J. P. Mesirov, "Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 102, no. 43, pp. 15 545–15 550, 2005, ISSN: 00278424. DOI: 10.1073/pnas.0506580102.

[44] J. Davis, W. Read-Button, and D. E. Peck, "L1000 Sop," Broad Institute, Tech. Rep., 2016.

[45] C. Yu, A. M. Mannan, G. M. Yvone, K. N. Ross, Y. L. Zhang, M. A. Marton, B. R. Taylor, A. Crenshaw, J. Z. Gould, P. Tamayo, B. A. Weir, A. Tsherniak, B. Wong, L. A. Garraway, A. F. Shamji, M. A. Palmer, M. A. Foley, W. Winckler, S. L. Schreiber, A. L. Kung, and T. R. Golub, "High-throughput identification of genotype-specific cancer vulnerabilities in mixtures of barcoded tumor cell lines," *Nature Biotechnology*, vol. 34, no. 4, pp. 419–423, 2016, ISSN: 15461696. DOI: 10.1038/nbt.3460. [Online]. Available: http://dx.doi.org/10.1038/nbt.3460.

[46] G. James, D. Witten, R. Tibshirani, and T. Hastie, *An Introduction to Statistical Learning with Applications in R*, 1st, J. Gareth, Ed. New York: Springer, 2013.

[47] M. Manimaran and K. Kannabiran, "On the Optimality of the Simple Bayesian Classifier under Zero-One Loss," *Machine learning*, vol. 29, no. 2-3, pp. 103–130, 1997, ISSN: 08856125. DOI: https://doi.org/10.1023/A:1007413511361. arXiv: 05218657199780521865715.

[48] G. L. Antonia, R. Guimera, C. Moore, and M. Sales-Pardo, "Accurate and scalable social recommendation using mixed-membership stochastic block models," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 113, no. 50, pp. 14 207–14 212, 2016, ISSN: 10916490. DOI: 10.1073/pnas.1606316113. arXiv: 1604.01170.

[49] O. Fajardo-Fontiveros, M. Sales-Pardo, and R. Guimera, *Node metadata can produce predictability transitions in network inference problems*, 2021. arXiv: 2103.14424 [physics.data-an].

Appendix 1. Programming code

All the codes we have developed for the implementation of the ML algorithms are on the repository: https://github.com/gemmabb/TFG_MoA_MLalgorithms. Attached to the code, there is a *readme* file including all the steps to follow.

Appendix 2. Additional figures

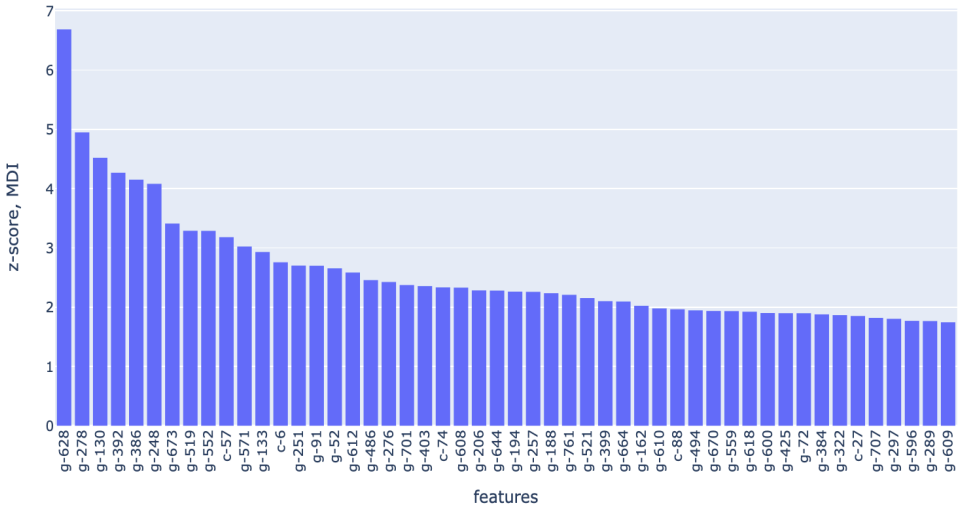


Figure A1. Z-scored feature importance for the prediction of the mechanism of action *acetylcholine receptor antagonist*. It is based on the z-scores of the MDI (Mean Decrease of Impurity) for each feature.