Feature Selection using Random Forests and Bootstrap Sampling on Cancer Drug Screening Data

Anahita Gorgipour

A thesis presented for the degree of
Bachelor of Science

Algorithmic Bioinformatics
Heinrich Heine University Düsseldorf
Germany
12th September, 2022

Acknowledgments

I would like to express my deepest gratitude to Prof. Dr. Gunnar Klau for breathing life into the proposed topic during our initial conversations, further piquing my interest in the topic. I would also like to thank my second assessor Prof. Dr. Tobias Marschall for graciously offering his time for this purpose. Last but not least, I would like to thank Nguyen Khoa Tran in the only way that words allow me to and express my sincerest appreciation for his time, advice, valuable feedback, and insightful conversations.
Thank you.

Abstract

Many scientific fields majorly benefit from technological advancements and research. Among those are medicine and biomedical sciences where this benefit is often the result of the analysis of large volumes of data in various ways, however, these analyses tend to involve numerous steps, one of which is the challenge of tackling high dimensional data. As the exploration of data continues, the high dimensionality of some datasets can prove to be a hindrance to the achievement of the relevant goals as a result of both higher storage requirements, higher computational costs, and an increased risk of overfitting.

For instance, the LOBICO algorithm includes an ILP capable of constructing an optimal formula in disjunctive normal form which provides information as to how the presence and absence of a number of genetic features correspond to drug sensitivity or resistance in cancer cell lines for various drugs individually. Unfortunately, LOBICO is also affected by the challenges that a high dimensional dataset can present, since the dataset that it uses for its predictions contains information regarding the mutations of 1508 genetic features corresponding to 808 cancer cell lines. The belief is, therefore, that LOBICO would greatly benefit from feature selection, a process by which the most significant subset of features is selected as a way of reducing the number of features used while ensuring that the most important features are kept.

The goal of this thesis is to explore random forests and bootstrap sampling as means of feature selection and determine which method is a viable choice for selecting the genetic features that should be used as one of LOBICO's inputs. To do so, we used random forest regressors to evaluate the importance of individual genetic features for each drug and selected the features with an importance that is higher than the average feature importance. We also attempted to select the most important subset of genetic features by using bootstrap sampling to sample the genetic features included in the dataset with replacement, then using the selected sample as the input for LOBICO, and finally selecting the features that frequently appeared in LOBICOs output.

A comparison of the reduction in prediction errors showed an average decrease of 5.195×10^{-3} in the prediction errors when a random forest regressor was used to select the most important features. By contrast, using bootstrap sampling in conjunction with LOBICO as the feature selection method resulted in an average prediction error increase of 3.569×10^{-3}. It can, therefore, be concluded that random forests achieve the goal of feature selection while simultaneously improving the model's performance. Bootstrap sampling, on the other hand, appears to be much less suitable for this purpose.

Contents

1 Introduction

Finding appropriate therapies for various types of cancer has been a topic of research and discussion for many years. The genomic instability of cancer, however, makes it very difficult to find a universal therapy that is effective on all cancer types for all patients. To combat this issue, personalized cancer therapy aims to improve each patient's outcome while simultaneously decreasing the toxicity levels suffered by the patient by tailoring the therapy to the individual [8]. This can be done by analyzing the genomic data of the cancer and basing the treatment approach on its results.

LOBICO is an algorithm that can be used for the purposes of such analyses, since its output is a logical formula containing information about which genetic mutations should be present or absent in the cell line in order for a particular drug to be an effective therapy [6].

In this thesis, we tackle feature selection using two different methods as a way of combating the inherent issues of using high dimensional data. The effectiveness of various Feature Selection methods is investigated and noted in many scientific papers and books, such as in [3, 2, 4]. In Section 2, we briefly outline and explain some topics such as ILPs, regression trees, and random forest regressors which serve as the basis for our explorations of the feature selection methods outlined in Section 3 which make use of random forest regressors along with bootstrap sampling in conjunction with LOBICO. We then offer an overview of the results of our experiments and explorations in Section 4 followed by a discussion surrounding explanations for the choices made throughout this thesis along with potential causes of the results and ideas for future improvements in Section 5. To finalize our work, we provide a conclusion in Section 6 with the aim of clearly answering the question we set out to answer.

> *Can random forests or bootstrap sampling be used to select the most important features within a high dimensional dataset effectively and efficiently?*

2 Preliminaries

The following sections provide an overview of the various topics which appear frequently throughout this work and serve as a way of ensuring that the reader is provided with all the pieces of information they require to fully understand the sections that follow. It is expected that the reader be familiar with some foundational topics such as *binary trees* and *Boolean logic formulas*, also called *Boolean equations* or *Boolean functions*.

2.1 DNF Formula

A Boolean logical formula that is in *disjunctive normal form* (DNF) is made up of literals, minterms, logical negations ($\neg$), logical conjunctions ($\wedge$), and logical disjunctions ($\vee$) where a literal refers either to a single variable or the negation of a variable. More specifically, formulas in disjunctive normal form are a disjunction of minterms which are made up of a conjunction of literals. It is also important to note that within a DNF formula, a negation can only appear along with a literal and not a minterm.
In the example $\Phi(a, b, c) = (a \wedge \neg b \wedge \neg c) \vee (a \wedge b)$, the formula is made up of the disjunction of two minterms and each minterm consists of the conjunction of three and two of the literals a, b, and c respectively.

2.2 Integer Linear Programs

An *Integer Linear Program* (ILP) refers to a mathematical program which can be used for solving linear optimization problems and consists of a linear objective function which is subject to a number of linear constraints. Additionally, ILPs requires all variables to be positive integers. An ILP can be represented as follows:

$$\max\ c^T x \tag{1}$$

$$\text{s. t.}\ \ Ax \leq b \tag{2}$$

$$x \in \mathbb{N}^+ \tag{3}$$

Given the above representation of an ILP, the matrix A, and the vectors b and c, the linear objective function [1] is subject to the constraints [2] while [3] states that the variables in vector x must be positive integers. To solve such a program, an ILP solver selects appropriate values for the variables in x in order to optimize the objective function. It is also important to note that ILPs can be expressed in different ways, however a conversion of one expressions to another is possible.

2.3 LOBICO

LOBICO is an algorithm which includes multiple parts such as the binarization of the drug response data provided in the input using a binarization threshold, the calculation of weights for each cell line, and an ILP which finds an optimal logical function in disjunctive normal form providing information as to how the presence and absence of a number of genetic features corresponds to drug sensitivity or resistance in cancer cell lines for various drugs individually [6].

Given the focus of this thesis and our requirements, we will mostly refer to the ILP portion of LOBICO and take some liberties as far as the other steps are concerned. These alterations are outlined and justified in Section 3. Having stated this, while delving into the details of LOBICO and its constraints are outside of the scope of this thesis and will, therefore, not be explained here, it is important to point out what LOBICO requires as its input in order to produce the aforementioned logical function. This is especially relevant as the input relates directly to the need for a feature selection step and as previously stated, is subject to minor changes in further steps. LOBICO takes the following as its input:

1. $N \times P$ binary matrix with N cell lines and P genetic features,
2. $N \times 1$ binary vector with binarized drug response values for each cell line,
3. $N \times 1$ binary vector with weights for each of the N cell lines.

It should also be mentioned that the number of minterms (K) and the maximum number of literals (M) in the output of LOBICO can be altered and adjusted according to the requirements of the use case. For instance, an arbitrary example for the output of LOBICO with $K = 2$ and $M = 3$ could look like the function $(a \wedge b \wedge c) \vee (\neg a \wedge d \wedge \neg e)$ wherein the literals a, b, c, d, and e represent the features and the information conveyed by this function states that both the simultaneous presence of mutations in the features a, b, and c and the simultaneous absence of mutations in the features a and e along with the presence of a mutation in d result in sensitivity to the drug for which LOBICO calculated the given logical function.

2.4 Feature Selection

Feature selection refers to the process of selecting a subset of features which contains the most relevant pieces of information. In doing so, it allows for the reduction of memory and storage requirements along with the reduction of the training and utilization times [7]. Additionally, feature selection combats some common issues that can arise when dealing with high dimensional data such as performance degradation as a result of data sparsity, model's inability to process a large number of features, and overfitting.

Feature selection methods can generally be classified into multiple categories. Filter methods are those where the feature selection is based on a set of predetermined criteria and is independent of the learning algorithm, whereas wrapper methods are those which rely on

specific learning algorithms to iteratively improve the quality of the chosen features. Finally, embedded feature selection methods make use of the embedded feature selection step within a learning algorithm and are a combination of filter and wrapper methods [5].
As far as the classification of the feature selection methods explored in this work is concerned, while feature selection using random forests can easily be classified as an embedded method due to the existence of an embedded feature selection approach in random forests, the second approach involving bootstrap sampling and LOBICO is harder to categorize. This is due to the fact that this approach does not solely focus on statistical calculations in order to select the most important features as filter methods do, however, it also does not consider all subsets of features including all combinations of features as a part of a greedy approach towards feature selection using a learning algorithm as wrapper methods do.

2.5 Regression Trees

Also known as *decision tree regressors*, *regression trees* are a method of supervised learning which make predictions in the form of continuous values based on a set of features and the corresponding observations.
At a higher level, regression trees can be described as tree structures where each leaf node represents a prediction and each internal node represents a split in the data which minimizes the variance of the target variable as a result of the split in question. *Sum of squared errors* (SSE), *mean squared error* (MSE), and *mean absolute error* (MAE) are some of the most prominent measures of variance with both mean squared error and mean absolute error along with the poisson deviance appearing as parameter options for the *criterion* parameter of the *DecisionTreeRegressor* included in the *scikit-learn* library [9].

In this thesis, the split quality at each node excluding leaf nodes is measured based on its ability to reduce the variance calculated using the formula below for mean squared error calculation where n represents the number of observations, $x_i{}'$ the predicted value, and x_i the original value of the observation target.

$$\text{MSE} = \frac{\text{SSE}}{n} = \frac{\sum_{i=1}^{n}(x_i{}' - x_i)^2}{n} \tag{4}$$

The example below showcases the regression tree for an imaginary training dataset where t denotes the continuous target, f the continuous feature upon which t depends, and t' the prediction in the leaf node. When making predictions on unseen data, one can follow the paths included in the regression tree in order to reach a leaf node containing a target prediction for an observed feature. It should be noted that the given example focuses on a solitary feature, however, in most scenarios, the target value is dependent on a much higher number of features as is the case in the scenario explored in this thesis. Another point worth mentioning is that, depending on the criteria used for stopping further splits at a node, the target prediction at the leaf node can be the target value for a single observation or the average target value for multiple observations.

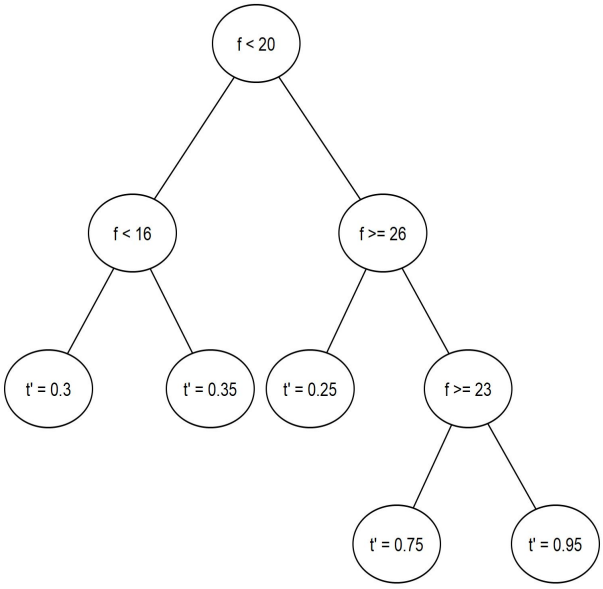

Figure 1: Example for a regression tree.

2.6 Random Forest Regressors

Random forest regressors are a combination of a predetermined number of regression trees where each tree is grown independently using a random sample of observations.

For regression problems, as with the one handled here, the final prediction of the random forest is the average value of the independent trees' predictions. Due to the low correlation between the individual trees, random forests typically outperform individual regression trees [1].

2.7 Data Splitting

In order to assess the ability of a trained machine learning model to make high quality predictions, a set of observations which are unknown to the model are required. This, therefore, calls for splitting the observations included in the dataset into two individual sets with one being used to train the model, appropriately referred to as the training dataset, and the other, aptly called the test dataset, being used to test whether the model is able to make good predictions for some unseen data.

Provided that a dataset contains enough observations for a productive split to be possible, $70-80\%$ of the observations are typically used for training and the rest are used for testing purposes. The `train_test_split` function of the *scickit-learn* library [9] provides a convenient way of splitting a dataset into training targets, training features, test targets, and test features. As the name suggests, a model can be trained to predict target values using the training features along with the training targets. The test features can be used to make target predictions and test targets can be used to measure the quality of the predictions.

Another method for assessing a model's performance that we make use of in Section 3 is *k-fold cross validation* which differs from the previous method, for the most part, in the way the observations are split. This method splits the observations in the dataset into k random groups. One group is then designated as the test dataset and the rest are used to train the model. Once the model has been evaluated using the designated test set, its performance is recorded and the model itself is discarded in order for a new model to be trained and tested using another group of observations, one that was previously used in the training phase, as the new test dataset and designating the rest of the groups including the previous test dataset as the new training dataset. This process is repeated until each group has served as the test dataset once. In this way, each group is used once as the test dataset and $k-1$ times as the training dataset.

3 Methods

In this section we describe the steps and considerations that we took during our exploration of random forest regressors and bootstrap sampling as methods of feature selection for the given dataset and outline the proposed workflows for this purpose.

3.1 Data and Preprocessing

We compiled the dataset used throughout this work using the data provided in the *Genomics of Drug Sensitivity in Cancer* (GDSC) dataset which contains information about the mutations in the (genetic) features of a cancer cell line along with the drug response for each combination of a drug and a cell line [10].

In an attempt to create the most complete dataset possible for the purposes of this thesis, the cancer drug response data and the genetic feature data of 31 different tissue types excluding cell lines without a classification in *The Cancer Genome Atlas Program* (TCGA) were combined and reshaped following a number of steps. First, the drug response data and the genetic feature data for all tissue types were combined independently.
The resulting datasets were then reshaped such that one column contains the cell line name followed by columns containing the drug response data as measured by the *Area Under Curve* (AUC) for each individual drug. These columns are followed by genetic feature columns containing information as to whether a mutation is present (=1) or absent (=0).
The final dataset contains 809 rows and 1700 columns with each row representing a cancer cell line, 1508 columns representing the name of the genetic features, and 192 columns representing the drug names. A sample of the preprocessed dataset can be seen in the table below.

Due to the limited timeframe within which this thesis should be completed, we made a number of data related decisions. Firstly, a total of 20 drugs with the highest standard deviation in their AUC values were selected to be used in the processes outlined in Section 3. This is to ensure that the process of feature selection using bootstrap sampling and LOBICO can be completed within a reasonable amount of time. The selected drugs are BI-2536, Vincristine, Gemcitabine, BMS-754807, CDK9-5038, AZD5991, Mitoxantrone, Docetaxel, ABT737, Trametinib, AZD5582, Pevonedistat, Daporinad, AZD5153, Navitoclax, Vinblastine, Staurosporine, Topotecan, Teniposide, Talazoparib.

Furthermore, while the original dataset contains two measures of drug responses, AUC and IC50, we made the decision to discard the IC50 values and use the AUC values in both methods of feature selection explored here. Considering the high correlation between the two measures as evidenced by the correlation coefficients of 0.91 or higher between the two values for all drugs, we expect this alteration to have close to no negative impact on the results. In addition to being highly correlated with the IC50 values, the AUC values also allow for the selection of a universal arbitrary binarization threshold which serves to simplify the LOBICO data preparation step, thus reducing computation times. It is important to note that

the binarization threshold only plays a role in the process of selecting the most important features using bootstrap sampling and LOBICO and has no role in the process involving random forests. This is due to the fact that the binarized drug response data is one of the outputs of LOBICO.

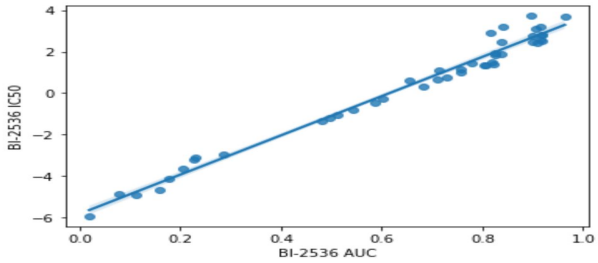

Figure 2: Correlation between the AUC and IC50 values for BI-2536.

		1508 genetic features			192 drugs		
	Cell Line	ABCB1	. . .	ARFGAP3	AMG-319	. . .	AZ960
808 cell lines	BICR10	1		0	0.962623		0.689235
	BICR22	0		1	0.978974		0.688967
	⋮	⋮		⋮	⋮		⋮
	RCC10RGB	0		0	0.888376		0.924589
	JVM-3	0		0	0.610823		0.428227

Table 1a: Sample from the processed dataset with continuous AUC values.

		1508 genetic features			192 drugs		
	Cell Line	ABCB1	. . .	ARFGAP3	AMG-319	. . .	AZ960
808 cell lines	BICR10	1		0	1		0
	BICR22	0		1	1		0
	⋮	⋮		⋮	⋮		⋮
	RCC10RGB	0		0	1		1
	JVM-3	0		0	0		0

Table 1b: Sample from the processed dataset with binarized AUC values using a threshold of 0.7 where the value 1 states that the cell line is sensitive to the given drug and 0 conveys the resistance of the cell line to the drug.

3.2 Hyperparameter Tuning

Due to its wide popularity and availability, we have used the implementations of the *RandomForestReressor* included in the *scikit-learn* library [9] which grows its trees in the form of *DecisionTreeRegressors* for the purposes of feature selection using random forests. It is, therefore, of great importance to ensure that appropriate parameters are chosen for growing the trees in the random forest, since a number of significant parameters directly impact the predictions of each independent tree, which in turn impacts the final prediction of the random forest. These significant parameters include, but are not limited to `min_samples_split`, `max_depth`, `max_features`, and `min_samples_leaf`.

Undeniably, a rather significant parameter when growing regression trees is the parameter `min_samples_split` which dictates how many samples should exist at a node in order for a split to take place. If the number of samples at a node is smaller than this predetermined number, then no further splits are possible and the leaf node which holds a prediction will hold the average value of all the samples at that node.
Generally, a minimum sample split of one should be avoided, due to the high likeliness of it resulting in overfitting, thus causing poor performance on unseen data. This is why the values 2, 5, and 10 were chosen as potential values to be considered for this parameter during the hyperparameter tuning step.
Furthermore, the parameters `max_depth`, `max_features`, and `min_samples_leaf` respectively dictate the maximum depth of the tree, the maximum number of features considered for each split, and the minimum number of samples that must be left on each side of a branch after a split.

In addition to the parameters used for the regression trees within the random forest, the parameters `n_estimators` and `bootstrap` were also included in the hyperparameter tuning step, due to the fact that the `n_estimators` parameter determines how many trees should be grown in a random forest, which in turn affects the prediction results, since the prediction results are the average of individual tree predictions.
The bootstrap parameter, on the other hand, determines whether the samples used for growing the trees are drawn with replacement. It is important to note that the sample size used for growing each individual tree always remains the same and this parameter solely determines whether repetitions within the selected samples are desired.

While it is fairly common to use grid search using GridSearchCV [9] in the hyperparameter tuning step to find the optimal parameters within a given range, this method poses the risk of increasing the computation times significantly when used with a broad range of parameters. This is due to the fact that all combinations of the given parameters are considered when using GridSearch which slows down the process by a significant amount. As a result, this method is primarily suitable for parameter searches within a smaller grid.

A randomized search using RandomizedSearchCV [9] avoids this problem by allowing the user to dictate how many combinations should be considered. Despite its ability to reduce

computation times, a trade-off that should be considered when using RandomizedSearchCV is that it does not guarantee that an optimal set of parameters can be found, due to the fact that it considers a limited number of random combinations of parameters.

Due to the time constraints present, this trade-off was seen as acceptable, and, therefore, RandomizedSearchCV with 100 parameter combinations and 5-fold cross validation was used in the hyperparameter tuning step of the feature selection process for each drug .
The arbitrarily chosen options provided for each of the discussed parameters are as follows:

```
n_estimators = [400, 560, 720, 880, 1040, 1200, 1360, 1520, 1680, 1840, 2000]
max_features = [1.0, 'sqrt']
max_depth = [None, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110]
min_samples_split = [2, 5, 10]
min_samples_leaf = [1, 2, 4]
bootstrap = [True, False]
```

3.3 Random Forest Regression

Given the dataset described in the previous section, the following steps were carried out for each one of the drugs out of the 20 drugs with the highest standard deviations. First, we carried out a hyperparameter tuning step using all the features in the dataset along with randomized search with cross validation.
Once a set of suitable parameters were found, a random forest regressor was fitted using all existing features along with the target AUC values for the drug. When a random forest regressor is fitted, each feature is assigned a feature importance based on its ability to reduce variance. We then used the *SelectFromModel* function of the *scikit-learn* library [9] with default parameters to select the most important subset of features present in the dataset for the drug in question. The selected features are those with a feature importance that lies above the threshold which is the mean of importance of all features by default. As a result, the number of selected features can vary greatly between different drugs.

In addition to the aforementioned steps, in order to determine whether a second hyperparameter tuning step using the selected features was needed, we trained three separate random forests using the reduced features; once using the default parameters of the *RandomForestRegressor*, once using newly selected parameters by performing a second round of randomized search, and once using the parameters that were selected in the initial hyperparameter tuning step. The reduction in the RMSE values which can be calculated by taking the square root of the formula outlined in Section 2.5 was used as the comparison criterion for deciding which method was the most effective.

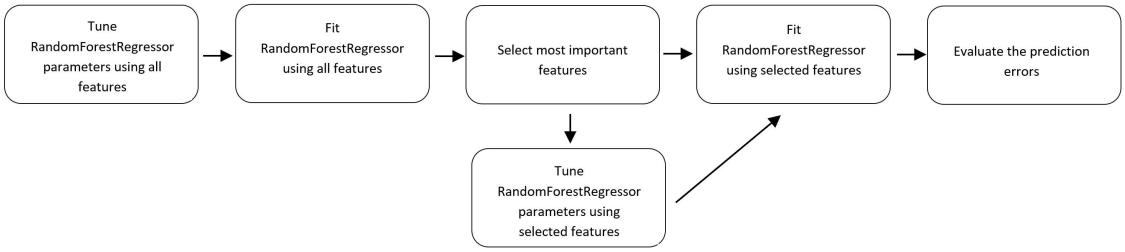

Figure 3: Steps in feature selection using random forest regressors.

3.4 Bootstrap Sampling

At its core, this approach can be summarized in three main steps with the first being the repeated sampling of all the features in the dataset with replacement, the second being the construction of a logical formula in disjunctive normal form using LOBICO, and the third being the filtering of the features based on a given criterion.

The filtering stage can occur in two different ways. One proposed approach is to select a set of features based on whether the frequency of their appearance in the LOBICO output exceeds a selection threshold calculated by dividing the number of samplings by two. The second approach, however, involves determining a minimum number of features desired and adjusting the selection threshold accordingly. While the first approach is more likely to result in the most important features being selected, it can also potentially result in the selection of no features. It is, therefore, worthwhile to also explore the potentials of the second approach.

In order to ensure that each feature is given an equal chance of appearing in LOBICO's input, we decided that each feature should be sampled 30 times. This was achieved by shuffling the list of feature names during each sampling step and splitting the 1508 features into sets of size n. Any remaining features which did not add up to n features in the last splitting were simply included in the last sample, since we can otherwise not ensure that an appropriate number of features is included in all samples. It should be noted that both 30 and n along with the selection threshold were selected arbitrarily based on a set of assumptions and additional experiments. One such experiment involved carrying out this process with samples of size $s = 50$, $s = 100$, and $s = 250$ in an attempt to answer the question of whether increasing the sample size, which in turn increases the required time for this process would produce significantly better results such that the increase in computation time could be justified.

In addition to the arbitrarily selected values mentioned above, we also decided to limit the number of minterms and the maximum number of literals per minterm to 3 for LOBICO to ensure that only the most important features were detected and the process could be completed within a reasonable amount of time.

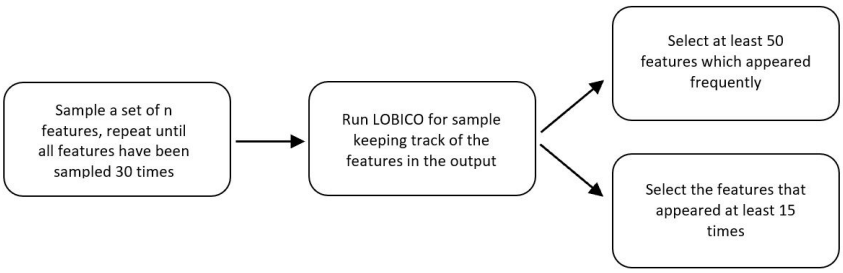

Figure 4: Steps in feature selection using bootstrap sampling and LOBICO.

3.5 Method Evaluation

In order to examine whether each one of the feature selection methods were successful, the selected features using each method were used to train random forests for each drug and the RMSE values were calculated based on the prediction and target values. These were then compared with the RMSE values calculated for a random forest trained on all the features in the dataset. A decrease in the RMSE values points at a successful feature selection process whereas an increase in the RMSE values suggests that some important features have been discarded.

4 Results

In this section, we present the results of feature selection using random forest regression and bootstrap sampling in conjunction with LOBICO. We will first showcase the results of each iteration of each method and elaborate on the modifications made in each iteration, the expected result, and the actual outcome following the implementation of those modifications. We will then present the evaluation results of both methods.

4.1 Random Forest Regression

When sufficiently trained on a training dataset with continuous target values and a set of features which affect the target value, a random forest regressor is expected to be able to make predictions about the target values of the unseen observations included in the test dataset. The quality of a prediction depends on its proximity to the actual value of the observed target and can be measured using various metrics such as MAE (mean absolute error) and RMSE (root mean squared error). We have chosen RMSE values as the evaluation metric for the performance of random forests, since RMSE penalizes larger errors relatively strongly and is measured in the same units of measurement as the target data unlike MSE and MAE.

As shown in the tables below, training a random forest regressor on all the features in the training dataset containing 75% of the observations and the default parameters resulted in an average RMSE value of 0.192 when the model was tested on the test dataset containing the remainder of the observations with the highest RMSE value corresponding to the drug BI-2536 and the lowest RMSE value belonging to the drug Teniposide.

Repeating the same process after tuning the parameters of the random forest regressor resulted in an average RMSE of 0.182 with the highest and lowest RMSE values corresponding to the drugs BI-2536 and Teniposide respectively. We can, therefore, observe a decrease in RMSE values across all drugs in comparison to the RMSE values for the random forest which was trained using default parameters. Moreover, training a random forest on the same percentage of observations using default parameters and the features with the highest importance resulted in an average RMSE value of 0.186. This suggests an improvement in the overall performance of the random forest with the drugs BI-2536 and AZD5153 having the largest decrease in their RMSE values.

The addition of a secondary hyperparameter tuning step to the feature selection and evaluation process using only the selected features during the evaluation process, resulted in an average RMSE of 0.180 which presents a decrease compared to both the initial random forest trained on all features and the random forest trained on the selected features using default parameters.

Drug Name	Count of Selected Features
BI-2536	199
Vincristine	439
Gemcitabine	458
BMS-754807	283
CDK9-5038	471
AZD5991	345
Mitoxantrone	421
Docetaxel	498
ABT737	397
Trametinib	306
AZD5582	277
Pevonedistat	472
Daporinad	269
AZD5153	279
Navitoclax	432
Vinblastine	450
Staurosporine	423
Topotecan	396
Teniposide	402
Talazoparib	384
	$\bar{x}$ = 380.05

Table 2: Individual and average number of selected features for each drug using random forest regressors with $\bar{x}$ representing the average number of selected features.

	RMSE— All Features		RMSE— Selected Features	
Drug Name	Default Parameters	Tuned Parameters	Default Parameters	Tuned Parameters
BI-2536	0.311	0. 293	0. 260	0. 272
Vincristine	0. 211	0. 210	0. 209	0. 207
Gemcitabine	0. 208	0. 208	0. 207	0. 206
BMS-754807	0. 237	0. 207	0. 233	0. 212
CDK9-5038	0. 188	0. 187	0. 186	0. 184
AZD5991	0. 224	0. 189	0. 218	0. 191
Mitoxantrone	0. 186	0. 177	0. 185	0. 177
Docetaxel	0. 180	0. 175	0. 179	0. 175
ABT737	0. 211	0. 198	0. 210	0. 198
Trametinib	0. 169	0. 155	0. 165	0. 152
AZD5582	0. 184	0. 173	0. 179	0. 168
Pevonedistat	0. 177	0. 177	0. 176	0. 175
Daporinad	0. 176	0. 173	0. 170	0. 169
AZD5153	0. 181	0. 165	0. 170	0. 164
Navitoclax	0. 185	0. 174	0. 180	0. 172
Vinblastine	0. 169	0. 163	0. 168	0. 163
Staurosporine	0. 158	0. 158	0. 157	0. 157
Topotecan	0. 162	0. 152	0. 160	0. 152
Teniposide	0. 146	0. 138	0. 144	0. 139
Talazoparib	0. 170	0. 161	0. 167	0. 162

Table 3: The RMSE values for random forests trained on all features and selected features using different parameters.

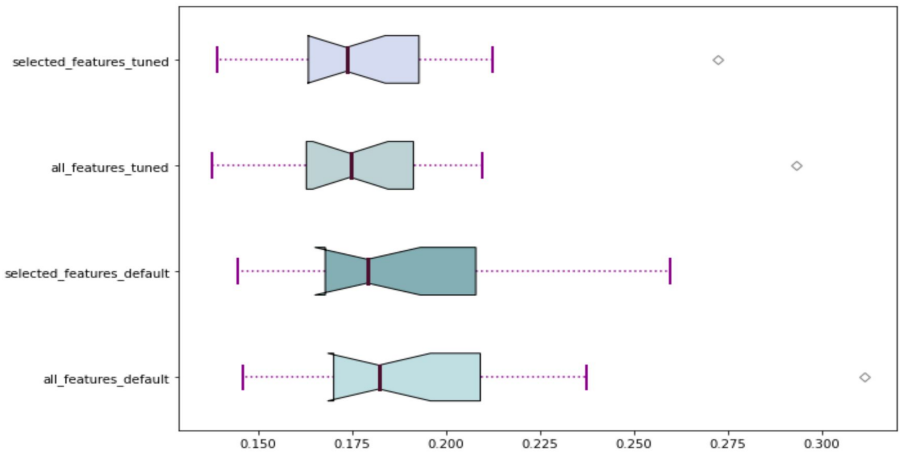

Figure 5: Plot of the RMSE values of random forest regressors trained on different features using default and tuned parameters.

4.2 Bootstrap Sampling

We proposed two approaches for selecting the most important features in Section 3 and carried out the proposed method for feature selection using bootstrap sampling in conjunction with LOBICO for a number of different sample sizes using both approaches for determining the selection threshold. The chosen sample sizes were $s = 50$, $s = 100$, and $s = 250$.

Using the first approach wherein features which appeared more than 15 times after each feature was sampled a total of 30 times, yielded no selected features when each of the samples in a sampling round contained 100 or 250 features. While still underwhelming, the results for the iteration using samples of size 50 showed more promise with only two drugs having 0 features and only one drug having 1 feature selected at the end of the process. The number of selected features using this sample size and the threshold 15 was 14 for the drugs Vincristine and Teniposide, 22 for Gemcitabine, 19 for CDK9-5038, 15 FOR AZD5991 and Mitoxantrone, 34 for Docetaxel, 17 for ABT737, 18 for Trametinib, 12 for AZD5582 and AZD5153, 26 for Pevonedistat, 23 for Navitoclax, 13 for Vinblastine and Topotecan, 21 for Staurosporine, and 20 for Talazoparib.

Though the results of the iteration using $s = 50$ and a selection threshold of 15 may appear counterintuitive at first sight, considering the fact that LOBICO gets closer to finding the optimal DNF as the number of features in its input increase, these results are expected due to the fact that the number of minterms (K) and the maximum number of literals (M) in LOBICO's output were left unchanged in the initial rounds of experimentation regardless of sample size. The lower individual count for each genetic feature can, therefore, be attributed to the decrease in the number of samples as the size of the samples was increased.

Aside from prompting further experiments focused on determining the impact of adjusted minterm and literal counts, these results also confirmed the need for the second approach which sets a requirement for the minimum number of selected features and lowers the threshold accordingly to ensure that at least the predetermined minimum of features are selected for

each drug. Despite the increase in the number of selected features when using this method, the quality of the predictions as measured by the RMSE values calculated for the predictions of the random forests trained on the features selected using the bootstrap process for different sample sizes, did not improve and remained the same. The number of features selected using this approach with a minimum feature number of 50 and the number of selected features using the first approach along with the shared RMSE values of the random forests trained on the selected features using the aforementioned sample sizes and selection approaches can be seen for each sample size and each drug in the tables below.

	Adjusted Threshold			Threshold = 15		
Drug Name	50 Features	100 Features	250 Features	50 Features	100 Features	250 Features
BI-2536	88	187	117	0	0	0
Vincristine	77	53	121	14	0	0
Gemcitabine	56	66	135	22	0	0
BMS-754807	80	77	122	0	0	0
CDK9-5038	84	101	130	19	0	0
AZD5991	92	108	126	15	0	0
Mitoxantrone	83	53	63	15	0	0
Docetaxel	79	64	64	34	0	0
ABT737	105	52	68	17	0	0
Trametinib	101	66	60	18	0	0
AZD5582	71	64	125	12	0	0
Pevonedistat	63	65	59	26	0	0
Daporinad	54	90	95	1	0	0
AZD5153	86	80	52	12	0	0
Navitoclax	58	55	125	23	0	0
Vinblastine	86	96	52	13	0	0
Staurosporine	55	65	72	21	0	0
Topotecan	79	96	124	13	0	0
Teniposide	72	99	124	14	0	0
Talazoparib	53	65	131	20	0	0
	$\bar{x}$ = 76.1	$\bar{x}$ = 81.1	$\bar{x}$ = 98.25	$\bar{x}$ = 15.45	$\bar{x}$ = 0	$\bar{x}$ = 0

Table 4: Number of selected features for each drug using different number of features and selection approaches.

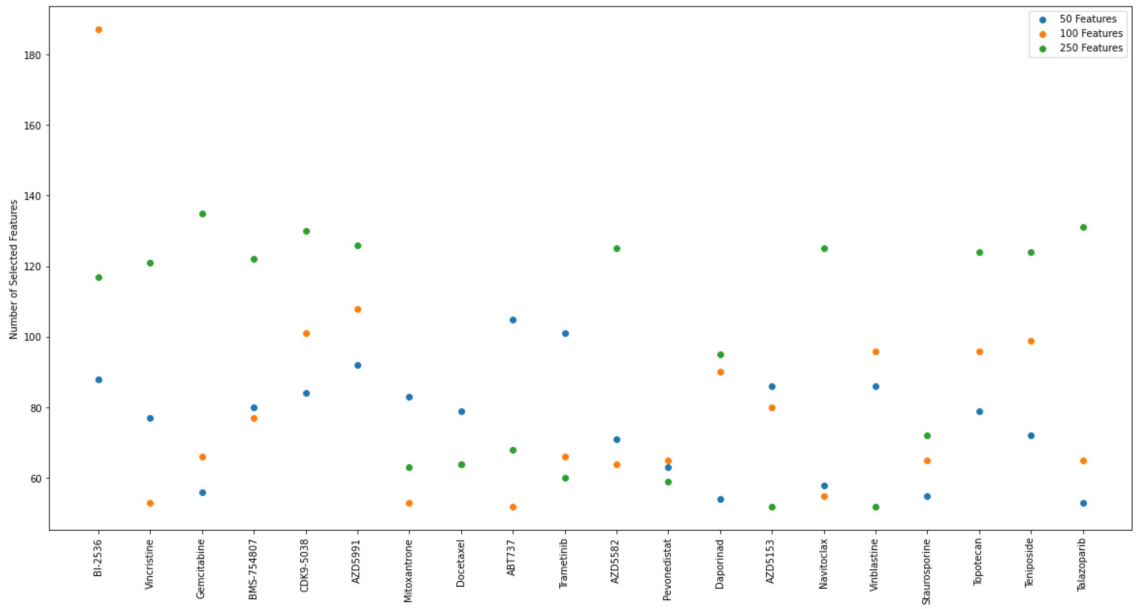

Figure 6: Individual and average number of selected features using an adjusted threshold for different sample sizes

In order to determine how each of the proposed methods performs compared to the other, we used random forest regressors trained on the features selected by each method using default parameters. Comparing the resulting RMSE values containing information about the performance of the models trained on the selected features provides insight into whether both methods can be deemed suitable feature selection methods for the given type of dataset. While a small decrease in RMSE values compared to a random forest trained on all features can be observed for some of the drugs for the models trained using the features selected by the bootstrap feature selection method, an average increase of 3.569×10^{-3} in RMSE across all the observed drugs rather suggests that this method was not successful at extracting the most important features included in the dataset.

The observed decrease in RMSE values for the models trained using the features selected by the random forest feature selection method, however, hints at the suitability of this method as a feature selection method, as it shows that the selected featured contained enough relevant information for the newly trained model to be able to make significantly better predictions for the AUC values in the test dataset. The improvement in performance compared to the random forest regressor trained on all features also provides some information about the high likelihood of overfitting occurring when the entire dataset was used to train the model. This information can be directly inferred from the fact that the model trained on all features was not able to perform as well as the model trained on a reduced number of features when it made predictions about unseen values in the test dataset. This strongly indicates that the high number of features caused the model to be overfitted to the training dataset and not be able to make accurate predictions on unseen data. The RMSE values for the random forest regressors trained on the different sets of features can be found in the table below.

Drug Name	All Features	Random Forest Selected Features	Bootstrap Selected Features
BI-2536	0.309	0.260	0.292
Vincristine	0.211	0.209	0.284
Gemcitabine	0.208	0.208	0.222
BMS-754807	0.237	0.234	0.218
CDK9-5038	0.187	0.186	0.209
AZD5991	0.223	0.217	0.194
Mitoxantrone	0.186	0.185	0.202
Docetaxel	0.180	0.179	0.188
ABT737	0.211	0.210	0.208
Trametinib	0.169	0.165	0.174
AZD5582	0.183	0.179	0.172
Pevonedistat	0.177	0.176	0.192
Daporinad	0.176	0.170	0.189
AZD5153	0.181	0.170	0.174
Navitoclax	0.185	0.180	0.180
Vinblastine	0.169	0.168	0.170
Staurosporine	0.158	0.156	0.168
Topotecan	0.161	0.160	0.169
Teniposide	0.146	0.144	0.154
Talazoparib	0.170	0.167	0.175

Table 5: RMSE values for random forest regressors trained on different sets of features.

Having observed that the having a fixed number of minterms and literals in the output of LOBICO regardless of the sample size resulted in no features being selected when a selection threshold of 15 was enforced, we decided to experiment with modified values for the count of minterms and the maximum count of literals within the DNF delivered by LOBICO. Due to the time constraints, we chose a single sample size, namely 100 and repeated the feature selection and evaluation steps using bootstrap sampling and LOBICO with both adjusted and unadjusted thresholds along with different pairs of values for K and M, such as ($K = 3$, $M = 6$), ($K = 4$, $M = 4$), and ($K = 6$, $M = 3$).

The results show that the number of selected features improved dramatically for ($K = 3$, $M = 6$), suggesting that an increase in the number of the minterms is required as the size of the individual samples increases in order to be able to select some features without an adjustment to the selection threshold being necessary. Despite this, no features were selected for two drugs when $K = 3$ and $M = 6$ and many more drugs were left without any selected features for the other two K – M value pairs. Given the number of drugs with no selected features when the threshold had the value of 15, we again turned our focus to the selected features using an adjusted threshold and used random forest regressors as described in Section 3 to evaluate the importance of the selected features by comparing the prediction quality of predictions made by a model trained on features selected by bootstrap and those made by a model trained on features selected by random forests. Disappointingly, the RMSE results remained the same, suggesting that the amount of information contained in the selected features remained the same, despite the changes in the number of selected features. An important factor contributing

to these results is the small sample sizes which prevents the detection of valuable information conveyed by the interaction of various features as a result of the nature of bootstrap sampling which leads to some features never appearing in a sample together.

	Adjusted Threshold			Threshold = 15		
Drug Name	K = 6,M= 3	K=4,M=4	K=3,M=6	K=6,M=3	K=4,M=4	K=3,M6
BI-2536	81	153	62	0	0	0
Vincristine	72	96	55	12	0	0
Gemcitabine	95	79	60	17	0	0
BMS-754807	93	55	53	0	0	0
CDK9-5038	80	90	107	15	0	0
AZD5991	92	94	68	19	0	0
Mitoxantrone	80	66	88	12	1	0
Docetaxel	67	68	81	24	2	0
ABT737	63	67	71	24	2	0
Trametinib	50	74	79	20	0	0
AZD5582	85	90	86	14	0	0
Pevonedistat	60	62	77	25	0	0
Daporinad	72	59	79	6	0	0
AZD5153	53	99	89	19	0	0
Navitoclax	63	51	75	23	3	0
Vinblastine	101	64	62	18	2	0
Staurosporine	63	67	60	23	0	0
Topotecan	84	98	56	15	0	0
Teniposide	74	54	97	12	0	0
Talazoparib	58	80	77	22	3	1
	$\bar{x}$ = 74.3	$\bar{x}$ = 78.3	$\bar{x}$ = 74.1	$\bar{x}$ = 16	$\bar{x}$ = 0.65	$\bar{x}$ = 0.05

Table 6: Number of selected features for each drug using different number minterms and literals and selection approaches.

5 Discussion

In this section, we analyze and reflect on the decisions made when designing the processes outlined in section 3 and propose ideas for future improvements of the processes.

5.1 Data and Preprocessing

As stated previously, LOBICO performs calculations for one drug at a time, however, a total of 192 drugs are included in the drug screening data found in the GDSC dataset. While the ultimate goal would be to consider all 192 drugs in this exploration, due to the time constraints, we made the decision to initially focus on 20 drugs which had the highest standard deviation and highly correlated values for both measures of drug sensitivity in the dataset, namely the IC50 and AUC values. The expansion of the list of drugs used can potentially provide more insight into the suitability of either of the discussed methods as ways of performing feature selection.

5.2 Random Forest Regression

Despite the fact that the addition of hyperparameter tuning steps resulted in the improvement of the performance of random forests, it is important to note that only a limited number of random parameter combinations are considered during this process which can result in the selection of parameters which are far from the optimal parameters.

Potential approaches for ensuring better outcomes as far as the random forest regressors's parameters are concerned could involve a secondary grid search using a smaller range of potential parameters by narrowing down the options around the parameters selected using randomized search and considering all combinations of parameters. It should, however, be noted that this approach would significantly increase the time required to complete the overall process of feature selection.

Considering the fact that the most time consuming step of the feature selection using this method is the hyperparameter tuning step, another approach would be to discard this step entirely and solely use the default parameters, since an improvement was noticed even when using the selected features and default parameters.

5.3 Bootstrap Sampling

Due to the randomized nature of bootstrap sampling, one can not ensure that at least a set of important features is included in every sample. This is further exacerbated by the relatively small sample sizes. It should be noted that increasing the number of features included in each sample from 50 to 250 hardly improved the results of the feature selection in this case, despite the assumption that it would, considering the fact that LOBICO produces outputs that are closer to the optimal logical formula as the number of features in its input increases. This is, in part, due to the randomness of the bootstrap sampling step and the high number of all features compared to the rather small sample sizes considered here in an attempt to ensure that all computations could be successfully finished within a limited timeframe. Another point encouraging the selection of a larger sample size for future improvements is the fact that the interaction of different features can convey more information than individual features.

A further increase of the sample sizes could still, however, lead to better results, since LOBICO gets closer to finding the optimal logic formula as the number of features used increases. This adjustment should ideally, however, be accompanied by an increase in the number of minterms and literals in the output of LOBICO as suggested by the results in the previous section in order to determine whether these adjustments which are thought to be potential improvements can, in fact, improve the quality of the selected features. The first portion of this option along with an adjusted threshold is worth considering due to the fact that only a minor increase of less than 60 minutes in the computation times was observed when the number of sampled features was increased from 100 to 250 features, however, it should be noted that this increase will not be linear as the number of features within a sample are increased and this adjustment could still result in a sharp increase in the amount of time required to complete the process of feature selection using bootstrap sampling and LOBICO. By contrast, increasing the number of minterms in the logical formula should be done cautiously as increasing the number of minterms from 3 to 6 lead to computation times which were twice as high.

6 Conclusions

In this thesis, we compared random forests and bootstrap sampling as methods of feature selection. As a commonly used method for feature selection, random forests were expected to outperform bootstrap in terms of the importance of the features that were selected. This expectation was, in part, also based on the time constraints which applied to the use of bootstrap sampling along with LOBICO for the purpose of feature selection. In addition to the factors mentioned previously, a major limitation of the bootstrap method is the fact that it requires an intricate balance to be reached between the number of features in a sample and the time required to compute the output of LOBICO.

As shown in Section 5, these assumptions and expectations were confirmed by the results of the comparison of the prediction qualities of random forests trained on the features selected using these two methods for the chosen drugs. We can, therefore, conclude that in its current form, the proposed method of feature selection using bootstrap sampling is not suitable for the purpose of selecting the most important genetic features in the dataset and thus unable to extract the genetic features that can be used to determine whether a cancer therapy has high chances of being an effective treatment for a specific type of cancer.

Although bootstrap sampling is determined to be an ineffective method of feature selection for the given dataset, we can conclude based on the results outlined in previous sections that random forests have the potential to aid in selecting the genetic features which convey the necessary information required by an additional learning algorithm such as LOBICO to make predictions about the effectiveness of a cancer drug.

Data and code availability.
Data and code for all methods is available at https://gitlab.cs.uni-duesseldorf.de/albi/albi-students/ba-anahita-gorgipour

References

[1] J. Ali et al. “Random Forests and Decision Trees”. In: *International Journal of Computer Science Issues(IJCSI)* 9 (Sept. 2012).

[2] C. Chen et al. “Ensemble feature selection in medical datasets: Combining filter, wrapper, and embedded feature selection results”. In: *Expert Systems* 37.5 (Apr. 2020). DOI: 10.1111/exsy.12553. URL: https://doi.org/10.1111/exsy.12553.

[3] P. Frasconi et al., eds. *Machine Learning and Knowledge Discovery in Databases*. Springer International Publishing, 2016. DOI: 10.1007/978-3-319-46128-1. URL: https://doi.org/10.1007/978-3-319-46128-1.

[4] M. Huang et al. “SVM and SVM Ensembles in Breast Cancer Prediction”. In: *PLOS ONE* 12.1 (Jan. 2017). Ed. by Enrique Hernandez-Lemus, e0161501. DOI: 10.1371/journal.pone.0161501. URL: https://doi.org/10.1371/journal.pone.0161501.

[5] A. Jovic, K. Brkic, and N. Bogunovic. “A review of feature selection methods with applications”. In: *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. IEEE, May 2015. DOI: 10.1109/mipro.2015.7160458. URL: https://doi.org/10.1109/mipro.2015.7160458.

[6] T. A. Knijnenburg et al. “Logic models to predict continuous outputs based on binary inputs with an application to personalized cancer therapy”. In: *Scientific Reports* 6.36812 (2016), pp. 1–14. DOI: 10.1038/srep36812.

[7] J. Li et al. “Feature Selection: A Data Perspective”. In: *ACM Computing Surveys* 50.94 (2017), pp. 1–45. DOI: 10.1145/3136625.

[8] T. R. Lu. “Personalized Cancer Therapy: A Perspective”. In: *Clinical and Experimental Pharmacology* 04.02 (2014). DOI: 10.4172/2161-1459.1000153. URL: https://doi.org/10.4172/2161-1459.1000153.

[9] F. Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[10] W. Yang et al. “Genomics of Drug Sensitivity in Cancer (GDSC): a resource for therapeutic biomarker discovery in cancer cells”. In: *Nucleic Acids Research* 41.D1 (2012), pp. D955–D961. ISSN: 0305-1048. DOI: 10.1093/nar/gks1111.

A Appendix

In order to recreate the dataset used throughout this thesis using the data found in the GDSC dataset [10], follow the steps below:

1. Download both the genetic feature data and the drug response data for each individual tissue type.
2. Concatenate all genetic features and save the resulting dataset into one csv file.
3. Remove any duplicate rows containing the same cell line name and genetic feature while keeping the last occurance.
4. Use the values of the column `genetic_feature` containing the name of each genetic feature along with the column `is_mutated` containing information about whether a genetic feature is mutated to pivot the table while using the `cell_line_name` as the index and reshape the data such that each column aside from the cell line name column holds the name of a genetic feature and its rows contain the mutation information for each cell line – genetic feature pair.
5. Replace any empty fields with 0 and remove the substring *mut* from the column names.
6. Repeat these steps for the drug response data by using the appropriate columns for each step, namely `cell_line_name`, `drug_name`, and `AUC` while keeping in mind that empty fields should be left empty in the drug response data and not replaced with 0 or any other values.
7. Ensure that the column holding the cell line name has the same name and perform an inner join on the two datasets by merging them horizontally using the cell line names.