FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Response Prediction to Cancer Cell Lines Treatment

João Tiago Chaves Miranda Ladeiras

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Rui Camacho (FEUP)

Second Supervisor: Miguel Rocha (UMinho)

July 24, 2015

Response Prediction to Cancer Cell Lines Treatment

João Tiago Chaves Miranda Ladeiras

Mestrado Integrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

Chair: João Pedro Mendes Moreira
External Examiner: Sérgio Matos
Supervisor: Rui Camacho

July 24, 2015

Abstract

Cancer is one of the diseases with the highest mortality rate in the world. To understand the different origins of the disease, and to facilitate the development of new ways to treat it, laboratories cultivate, *in vitro*, cancer cells (cell lines), taken from patients with cancer. These cell lines enable researchers to test new approaches and to have an appropriate procedure for comparison of results.

At EMBL-EBI Institute (Cambridge, UK) an initial study was performed in which the effect of a large number of molecules was tested, in laboratory, in the treatment of cell lines with various types of cancer. This study also included the use of Machine Learning algorithms to build models to predict the degree of efficacy of those drugs in cancer treatment.

The methods used in the reported initial study were based on algorithms that construct "propositional like" models. The results reported are promising but, we think, can be improved. Another limitation of the algorithms used in the original study is the absence or severe lack of comprehensibility of the models constructed. In areas of Life Sciences, the possibility of understanding the forecast model is an asset to help the specialist to understand the phenomenon that produced the data.

Our thesis work has two main objectives: i) improve the performance of forecasting methods; and ii) construct understandable models. To meet these objectives we proposed the use of Graph Mining and Inductive Logic Programming (ILP) algorithms.

Resumo

O cancro é uma das doenças com maior índice de mortalidade em todo mundo. Para perceber as diferentes origens da doença, bem como para facilitar o desenvolvimento de novas formas de a tratar, os laboratórios cultivam, *in vitro*, células cancerígenas (*cell lines*), colhidas em pacientes com cancro. Estas linhas celulares permitem aos investigadores testar novas abordagens e terem um processo adequado para comparação de resultados.

No instituto EMBL-EBI (Cambridge, Reino Unido) foi realizado um estudo inicial em que o efeito de um grande número de moléculas (fármacos) foi laboratorialmente testado no tratamento de linhas celulares com diversos tipos de cancro. Esse estudo incluiu ainda a utilização de algoritmos de Aprendizagem Computacional (*Machine Learning*) na construção de modelos usados para prever o grau de eficácia de fármacos no tratamento do cancro.

Os métodos usados no estudo original baseiam-se em algoritmos que constroem modelos com um nível de representação equivalente a lógica proposicional. Os resultados reportados nesse estudo são promissores mas, pensamos nós, podem ser melhorados. Outra limitação dos algoritmos usados no estudo original é a ausência ou forte limitação de compreensibilidade dos modelos construídos. Em áreas das Ciências da Vida, a possibilidade de compreender o modelo de previsão é uma mais-valia para ajudar o especialista a compreender o fenómeno que produziu os dados.

O trabalho da dissertação tem dois objetivos: i) melhorar o desempenho dos métodos de previsão; e ii) melhorar a compreensibilidade dos modelos construídos. Para satisfazer estes objetivos foram utilizados algoritmos de *Graph Mining* e *Inductive Logic Programming* (ILP).

Acknowledgements

I would like first of all to express my sincere gratitude to my supervisor Rui Camacho. Without his constant effort, interest and availability, this project would not be possible.

I will be eternally grateful to my parents Augusto and Manuela, and my sisters Ana and Rita, for their support, guidance and dedication.

Finally, a special thank you to my friends and colleagues that were with me over these last years, for all the joint effort and for all the good moments and laughs.

João Tiago Ladeiras

"If you torture the data enough, nature will always confess."

Ronald Harry Coase

Contents

List of Figures

LIST OF FIGURES

List of Tables

Abbreviations and Symbols

ASCII	American Standard Code for Information Interchange
BK	Background Knowledge
CART	Classification And Regression Tree
CNV	Copy Number Variation
CV	Cross Validation
DM	Data Mining
DT	Decision Trees
EA	Evolutionary Algorithms
GM	Graph Mining
IC50	Inhibitory Concentration 50
ILP	Inductive Logic Programming
KDD	Knowledge Discovery in Databases
ML	Machine Learning
MIS	Microsatellite Instability Status
NN	Neural Networks
PaDEL	Pharmaceutical Data Exploration Laboratory
RF	Random Forest
RMSE	Root-Mean-Square Error
RSS	Residual Sum of Squares
SAR	Structure-Activity Relationship
SMILES	Simplified Molecular-Input Line-Entry System
SV	Sequence Variation
SVM	Support Vector Machine

Chapter 1

Introduction

This work falls within the scope of Bioinformatics and Machine Learning and addresses the problem of using Machine Learning methods to improve the development procedures of new drugs for cancer treatment.

1.1 Motivation and Goals

Besides being one of the diseases with the highest mortality in the world, cancer is also one of the diseases with the highest morbidity, i.e., patients suffering from the disease are subject to a very low quality of life. It is estimated that the disease remains currently the leading cause of death in developed countries, and the second leading cause of death in under-developed countries [MFB08], according to predictions based on estimates made in 2012 [JIM+13].

Methods used in the disease treatment often lack specificity in each case, i.e., although there are many similarities between cases, most treatments are not customized. The lack of treatment customization, may reduce the success of the treatment, and can also lead to its aggravation.

In an attempt to facilitate the development of new forms of treatment, laboratories cultivate cancer cells [Cha04], taken from patients with the disease, called cell lines. These cell lines consist of arrays of genomically identical cells, and are useful both in the study of their properties and as a test of the effects of various drugs, allowing consistency and reproducibility of the studies results.

In an early study [MIG+13], the effects of a large number of chemical compounds in various cell lines of different cancer types were tested in laboratory. The level of sensitivity of cells to these compounds was measured using the IC50, which measures the effectiveness degree that a substance has to inhibit a particular biological function. This study also involved the use of Machine Learning methods to predict the degree of efficacy of drugs in different types of cancer.

The number of possibilities for testing cell-compound combinations is quite extensive. The number of possible laboratory experiments go beyond the feasibility as time and resources are

concerned. It is therefore very useful to use Machine Learning methods to estimate the results of new drugs.

Another interesting use for these kind of methods is the study of already developed drugs with the intention of re-use, i.e., the repurposing of existing compounds used to treat other conditions [AT04] [SK11].

Moreover, it is possible to study the behaviour and the relationship between the various compounds and cells.

In the early study of [MIG+13], the Machine Learning methods used produce only "propositional-like" models. In these type of domains it would be very useful to understand the phenomenon that produces the disease. Therefore, the objective of the thesis work is to construct understandable models using relational learning algorithms. We also aim at improve the accuracy of the results and for that purpose we will investigate the usefulness of other "propositional learners".

1.2 Proposed Solution

In the thesis work we have initially replicated the results obtained in [MIG+13], providing a base line for the performance evaluation of further analysis. Additionally, using the same data preparation, we will run other types of propositional algorithms.

We will be also using Graph Mining (GM) as a way to gather more information about molecule substructures, aiming to complement the drugs' characterization and try to improve the performance of the algorithms.

In a second phase, we have used Inductive Logic Programming (ILP). The ILP system Aleph [Sri03] will then be used to induce relational models. The models produced should result in an understandable hypothesis that can help the biochemist experts to understand what causes, in the molecules uses, the increase in efficiency in cancer treatment.

For the completion of the project we are comparing results between models, expecting increased performance for the relational algorithms.

1.3 Dissertation Structure

After the current introductory chapter we have the following chapters.

Chapter 2 introduces the main concepts of the domain of cancer treatment. It also includes the analysis of the state-of-the-art associated to the domains of our work. This includes Machine Learning methods used in the study of sensitivity of cancer cells to drugs, as well as the source and treatment of the data and their outcomes. In this chapter we review various articles of similar work and how they can be compared with the current project.

In chapter 3 we detail the experiments made for the various types of proposed algorithms. We demonstrate, besides the data pre-processing details, the methods and results of our experiments.

We discuss the overall results of our work in terms of performance and comprehensibility of the models generated by the ILP. We also mention the work that could not be performed and its

motives. It is also proposed not only eventual improvement work, but also that gives continuity to the project in the future.

Chapter 4 draws the conclusions of this project, where we give a final view on the experiments, results and its meaning given the motivation.

Chapter 2

The Background

This thesis addresses the application of Knowledge Discovery in Databases (KDD)[1] methodology to analyze experimental data generated when solving a Life Science domain problem. The specific problem is that of determining, from a set of given drugs (molecules), which (and why) are effective in cancer treatment.

In this chapter we first introduce Biological and Pharmacological basic concepts necessary to understand the rest of the thesis report. We then introduce KDD methodology and an overview of the learning methods involved in our study.

2.1 Domain Concepts

As stated earlier the thesis is concerned with the application of KDD to the construction of predictive (as well as explanatory) models for the efficiency of drug in cancer treatment. We therefore present in this section basic concepts in Molecular Biology and Pharmacology.

In cancer research, and for ethical reasons, it is usual for researchers to collect small samples of cancer tissue (doing a biopsy) and then cultivate cells from those tissues in laboratory (cell lines). This procedure has the advantage of making available for research *a lot of* cancer cells without injuring the patients. It also allows for different techniques to be compared fairly (in the same "tissue"). For the data analysis we did on the thesis, we have to characterize the cells by means of a set of features. We have also to point out that the efficiency of drugs in the cancer cells will be addressed as a Structure-Activity Relationship (SAR) problem. That is to say that the prediction of the effect of a drug will be explained solely by its structure and small set of molecule global features.

[1] KDD is most often referred to as Data Mining (DM). Although DM is one of KDD steps it is, nowadays, quite frequent to use KDD and DM interchangeably. We will be using the term DM to mean both the DM step of KDD and the whole KDD process.

Cell Line Features [MIG+13], are based on cells genomic background. Binary types of features were used, such as microsatellite instability status[2] and sequence variation[3], as well as a ternary type feature, the copy number variation[4].

Molecular Descriptors are a set of characteristics of molecules that might be helpful for SAR problems. In our study they are generated by the PaDEL [Yap11] software, having SMILES as the input format, which contain different characteristics of the drugs present in the data set, such as weight, lipophilicity and rule of five[5]. SMILES [Wei88] [WWW89] [Wei90] is a standard representation of the structure of a chemical compound in a sequence of ASCII characters.

Molecular Fingerprint is a type of encoding, usually binary, that is associated with the molecular structure of a given molecule. E.g., when used as a feature of a compound, it indicates whether or not a known particular molecular substructure or property is part of the substance molecule.

Sensitivity is the target variable of the process which is the degree of effectiveness that drugs have inhibiting cancer cells. The measure used is the IC50, that represents, for a 50% inhibition, which is the concentration of the drug.

2.2 Knowledge Discovery in Databases

There is recently much interest in the use of Machine Learning methods in Chemistry and Life Sciences domains and more particularly assisting in the development of new drugs for treating various diseases. In this particular case, the ability to predict the sensitivity level, which a cancer cell has to a given drug, is of high importance.

Before performing the review of the methods used in previous studies, we present the standard procedure of a Knowledge Extraction (*KDD*) [FPSS96] analysis.

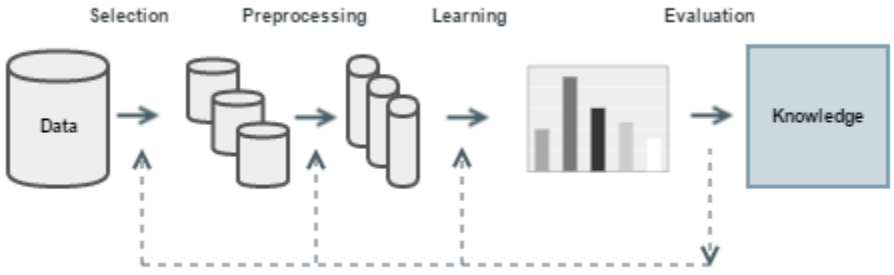


Figure 2.1: Knowledge Extraction Process

[2]The microsatellite instability status indicates whether or not there is an abnormal behavior in correcting DNA errors that occur in its replication.

[3]A sequence variation represents changes in the protein sequence

[4]The copy number variation represents the abnormal copies of one or multiple sections of the DNA sequence

[5]A rule of thumb to evaluate druglikeness or determine if a chemical compound with a certain pharmacological or biological activity has properties that would make it a likely orally active drug in humans.

KDD is a data analysis procedure aiming at the discovery of non-trivial rules or properties that can be useful for business or science studies. One of the core steps of KDD is the application of a data analyses algorithm, typically a Machine Learning algorithm. We will use, from know on, the designation Data Mining (DM) and KDD interchangeably, as is quite frequent in the literature to do so.

DM can be used for several types of data analysis problems. These problem include: classification, regression, clustering, association-rule mining, among others. In this thesis we will address only classification and regression problems. As can be seen in Figure 2.1, initially a pre-processing of the data is made (missing values are removed, inconsistencies in the data are corrected, normalisation of numerical values can be performed, etc). The data is then selected according to criteria related to the study in question and prepared for the learning phase. At this stage a model is created using a selected algorithm, or set of algorithms, which is subsequently tested. The evaluation results of this test show the effectiveness of the model produced.

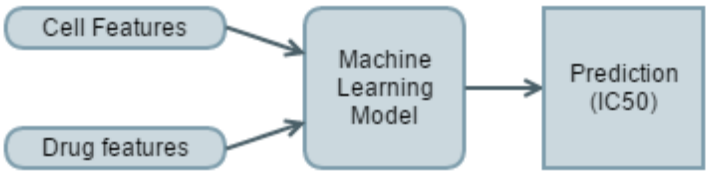


Figure 2.2: IC50 value prediction

The learning method, therefore, involves a selection and processing of data so it can be used in the construction of a predictive model (Figure 2.2). For the current project the input data are characteristics of both cells and drugs, and the output data is the result of a prediction of IC50 values, which can be treated as a regression or classification problem, when discretizing the values to be predicted. Therefore, each data set instance consists not only in a group of features of one drug and one cell, but also in a IC50 value.

2.2.1 Data Pre-processing

Similarly to the Knowledge Extraction process (Figure 2.1), the reviewed studies follow a methodology for data pre-processing. This step is extremely important to obtain good results in the study.

The type of input data was quite challenged in recent studies, being argued that the models that were built were considered too optimistic, result of too simplistic formulations [PAP+14]. This may result in deceptively positive performance evaluations, and should be taken into account for the current problem formulation also in terms of evaluation methods.

In [MIG+13] a study is carried out using Machine Learning for predicting the sensitivity level that various cell lines have to multiple drugs, taking into account the genomic and chemical properties respectively. The current study is based mainly in the recreation of this, and the results used for comparison with a similar study to be carried out using relational algorithms.

2.2.2 Learning Algorithms

In this section we describe the algorithms used both in previous and the current study.

Neural Networks

This kind of learning algorithm, introduced in 1943 [MP43], is inspired by the neural networks present in the central nervous systems of animals. It consists in a group of input and output units called neurons. It is usually divided into layers having an input layer, an output layer and N intermediate layers, called hidden layers. Between each layer there is a connection between each neuron, that represents the connection weight. If the number of hidden layers is 0, the neural network is called as single layer and if it's greater than 0 it's called as multi-layer (Figure 2.3).

Each neuron receives several real values as input, that originated from other neurons in previous layers, and produces a single value as output. The operation consists in the sum of the multiplication of each weight received in each neuron by the input value (0 or 1), are then applying an activation function. This function can take many forms, depending on the problem in question:

Linear Activation:

$$\phi(z) = z \tag{2.1}$$

Logistic Activation:

$$\phi(z) = \frac{1}{1+e^{-az}} \tag{2.2}$$

Threshold Activation:

$$\phi(z) = sign(z) = \begin{cases} 1 & \text{if z} \geq 0 \\ -1 & \text{if z} < 0 \end{cases} \tag{2.3}$$

Hyperbolic tangent Activation:

$$\varphi(u) = tanh(\gamma u) = \frac{1-e^{-2\gamma u}}{1+e^{-2\gamma u}} \tag{2.4}$$

The training of the model with one layer, consists in varying the weights of each connection between neurons, that are initially assigned randomly. For this it is used the evaluation function (Equation 2.5), being r the rate of learning, t the target output and a the current output. The learning rate affects the speed at which the network is trained. If the learning rate is too small or too large the network will learn too slowly or too quickly, which influence the quality of results.

$$w_i = w_i + r(t-a)x_i \tag{2.5}$$

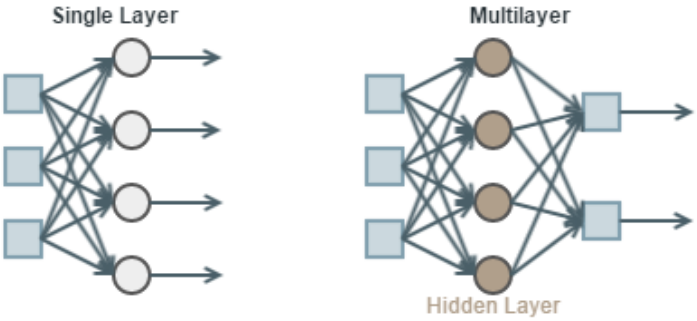


Figure 2.3: Types of Neural Networks

In the case of multilayer networks, the training is usually done using the back-propagation algorithm, which consists of two parts, propagation and update of the weights. In the propagation phase the inputs are updated using Equation 2.6 and the propagation of the output is made using one of the activation functions. In the weights update phase, the error is calculated using Equation 2.7 in an input or output layer, or by Equation 2.8 in an hidden layer. The errors are therefore calculated in the reverse direction of the network, updating the weights (Equation 2.9) and bias (Equation 2.10) of each neuron.

$$I_j = \sum_i x_{ij} O_i + \theta_j \tag{2.6}$$

$$Err_j = O_j(1 - O_j)(T_j - O_j) \tag{2.7}$$

$$Err_j = O_j(1 - O_j) \sum_k Err_k w_{jk} \tag{2.8}$$

$$w_{jk} = w_{jk} + (r) Err_j O_i \tag{2.9}$$

$$\theta_j = \theta_j + (r) Err_j \tag{2.10}$$

This algorithm has the advantage of being extremely versatile and can be used in resolving a large number of problems of different natures. In addition, it is also very easy to implement and leads to fairly good results in a wide variety of problems, such as in drug discovery. On the other hand, neural networks require fairly large processing capacity, making it very costly in terms of time. The models produced by the algorithm are very difficult for human reading, making it hard to interpret.

In [MIG+13], only 3 layers were used for the neural network creation, varying the number of elements in the hidden layer between 1 and 30. After 300 iterations and minimizing the RMSE

(Section 2.2.3), it was concluded that, to avoid overfitting, it would need between 21 and 27 elements in the hidden layer. The overfitting usually happens in very complex models that tend to be too specific to the training data, making it difficult to generalize the model.

The resulting IC50 values present in the data set were normalized, as in this case, in the model training phase, a sigmoid function was used (Equation 2.2), which only returns values between 0 and 1.

Decision Trees

This learning method's name stems from its flowchart structure in a tree shape. Each node corresponds to a test made on a given attribute and each branch corresponds one of the possible outcomes of the test. The leaf elements stores the prediction result (class value) (Figure 2.4).

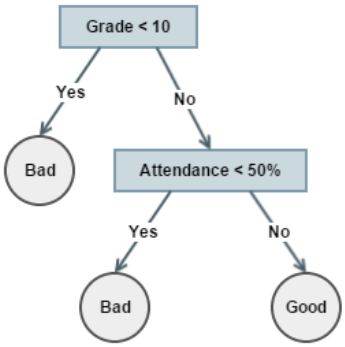


Figure 2.4: Simple example of a Decision Tree for students classification

The construction of the tree is done by first selecting the best feature, which is measured by their ability to divide samples into different groups based on the dependent variable, creating a node. This is done recursively for its branches until it reaches the stopping criterion.

There are several algorithms used for decision trees construction. Among those algorithms there are the CART [BFSO84] and ID3 [Qui86] and its extension C4.5 [Qui93].

C4.5 uses the information entropy in the construction of the decision tree, which allows to measure the information gain in every attempt to split a parameter.

This type of algorithm is usually considered quite easy to read, even by technicians from different areas. Another advantage of the algorithms is the capability of being combined with other types of learning methods.

Random Forests

The Random Forest (RF) is an ensemble Machine Learning method [Bre01]. Ensemble methods combine the output of a set of learning algorithms, called base learners. These type of method is used in order to increase performance. In the case of Random Forests, it uses multiple decision

trees (a version of the CART algorithm) produced with subsets from the data set, randomly selected with replacement.

Each tree is constructed independently of the others, using the same method as described in the previous section (2.2.2), for each random generated subset.

The result of the prediction is the vote of each model produced, and the one of most votes is selected (Figure 2.5). RF can be used for both classification and regression problems.

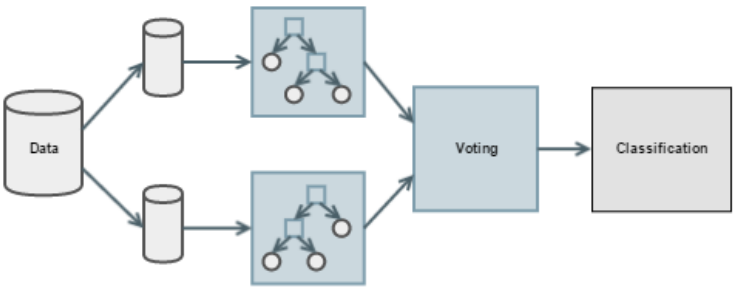


Figure 2.5: Random Forest process

In this project we use the R package "randomForest"[R C15][LW02] to generate our models. This package contains a function ("importance") that allows to measure which are the most important features in our training set. This is done by calculating the total decrease in node impurities from splitting, in the variable, and then averaged over all trees. This impurity is measured by the Gini index in classification and Residual Sum of Squares (RSS) in regression. Gini index measures the inequality between values in a frequency distribution and the RSS measures the discrepancy between actual and predicted values.

Linear Classifier

This method uses the features of the examples in an attempt to find an optimal hyperplane separating the classes (Figure 2.6). This is usually done using a threshold function. The algorithm is adequate for problems where the data is linearly separable [Fis36].

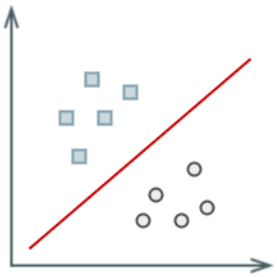


Figure 2.6: Linear Classifier example of a two class and two features problem

Support Vector Machines

Support Vector Machines (SVM) can be used for both classification and regression problems and can be used for linear and nonlinear data [CV95]. The data from two different classes, can always be separated transforming it into a higher dimension, through nonlinear mapping. Therefore, the algorithm searches for the linear optimal separating hyperplane in the new dimension space. This hyperplane is found using support vectors, which are the training points closer to a linear classifier margin. This can be achieved using the "kernel trick", which uses a kernel function that allows to work in a transformed feature space. These functions can take several forms while its choice depends largely on the type of problem. The most used kernel functions are:

Linear:

$$u'v \tag{2.11}$$

Polynomial:

$$(\gamma u'v + coef0)^{degree} \tag{2.12}$$

Radial Basis:

$$e^{-\gamma|u-v|^2} \tag{2.13}$$

Sigmoid:

$$tanh(\gamma u'v + coef0) \tag{2.14}$$

This fairly new method is being widely used due to its high accuracy despite its slow training. SVMs are also know to perform well under very large number of features.

Naive Bayes Classifier

This classifier uses the Bayes theorem (Equation 2.15) to estimate the probability of a given example matching any of the classes, being a a class and b a given feature [RN95].

$$p(a|b) = \frac{p(b|a)p(a)}{p(b)} \tag{2.15}$$

The method assumes independent distribution of the features, so we can calculate the probability of a given class a generating the feature b as in Equation 2.16, being b_i the features of the examples.

$$p(b|a) = p(b_1|a) * p(b_2|a) * ... * p(b_n|a) \tag{2.16}$$

Relational Algorithms

The learning algorithms, described in the previous section, have a set of limitations that make them inadequate for some DM problems. First of all, the information provided to the ML system has to fit in a single table of a relational database. the representation is of type feature/value. If several

relations/tables are stored in a database they have to be converted into a single one. This may lead to loss of information. Moreover the representation scheme of such algorithms is also very limited. They are equivalent to a representation power of propositional logic. When applying these algorithms, a lot of pre-processing and significant increase of the number of features is required to handle data with structure. This is precisely the case in our problem.

Multi-relational algorithms, such as ILP, do not "suffer" from the above mentioned shortcomings. They have no difficulty in representing data with structure and are able to estimate highly complex models.

Inductive Logic Programming

Inductive Logic Programming (ILP) [MDR94] is a sub-area of Machine Learning characterised by using First Order Logic to encode both data and hypotheses (the models). An ILP system accepts three type of input: Background Knowledge (BK); constraints to the hypothesis language; and examples (ILP is a type of supervised learning). The BK is made up of all information the domain experts thinks is valuable to construct the hypothesis. The constraints provided to the system avoid the construction of nonsense hypothesis. Examples are of two types: positive and negative. Positive examples are instances of the target concept and negative examples are not (used to avoid overgeneralization).

The main advantages of ILP over other [propositional] learning algorithms include the following ones. ILP has no difficulty in handling data with structure. Most often the induced models are intelligible. Models can include numerical an relational computations. It is very easy to provide useful information to the systems (no need to constrain the data to an attribute/value format). Data (BK) can be given in diverse formats and be originated from diverse sources.

ILP has proven to be very useful modeling structure-activity relationships (SAR), which are important in drug development studies, whereby it finds relationships between structures and substructures of chemical compounds and the activity (properties) of these [KMLS92].

Graph Mining

As the name suggests, Graph Mining algorithms are designed to find patterns (common substructures/subgraphs) in a given set of graphs. They are adequate to analyze complex substructures as chemical compounds, as the ones we have in our domain. GM algorithms look for isomorphisms between graphs. They have been widely used in rational drug design in the process of searching for common substructures of a set of drugs.

There are several types of algorithms that can find common substructures of a graph database. Among these algorithms there is the gSpan (graph-based Substructure pattern mining) [YH02], used in this project, which showed a superior performance, compared to other methods previously developed. The gSpan allows the pattern mining, given a minimum support and a minimum size (number of vertexes) of the patterns, for a set of graphs.

Another very efficient algorithm is GASTON (GrAph, Sequences and Tree extractiON) [NK], which attempts to improve upon previous methods efficiency, using the "quickstart principle" that takes into account that structures are part of each other.

2.2.3 Model Evaluation

In Machine Learning is important the predictive model quality evaluation. This evaluation allows to estimate the model's performance for any given new input examples. This is done separating the data set into a training and testing sets, avoiding using the same examples in both [K+95]. There are different techniques for splitting the data set, often used in both regression and classification problems, which may vary according to the problem context.

Hold-out: Consists in splitting the data set in two different sets of data with a given ratio, each of which containing proportionally the same amount of classes. This is usually done with 70% for the training set and 30% for the test set. Therefore, the model is trained with the training set and is tested once with the test set.

K-fold Cross-validation: The Cross-validation is done by partitioning the data set in K different blocks of similar size. The method is done K times and, for each iteration, one of the blocks is taken as the test set, and the model is trained with the remaining ones.

Leave-one-out: When the amount of data is small, a special case of K-fold may be used. In the Leave-one-out the number of K is the number of examples.

There are several measures that can be used to assess the performance of a regression or classification algorithm. For a classification problem a confusion matrix is built and a set of measures is calculated [Pow11]. Figure 2.7 shows an example of a confusion matrix for a 3 class classification problem and concerning the prediciton of class 1. The values depicted match the amount of examples classified or misclassified as class 1.

True Positives (tp): Number of examples correctly classified as Class 1.
False Positives (fp): Number of examples incorrectly classified as Class 1.
True Negative (tn): Number of examples correctly classified as not of the Class 1.
False Positive (fp): Number of examples incorrectly classified as not of the Class 1.

Evaluation Metrics

Accuracy: Distance between the predicted data to its true value.

$$accuracy = \frac{tp - tn}{tp + tn + fp + fn} \tag{2.17}$$

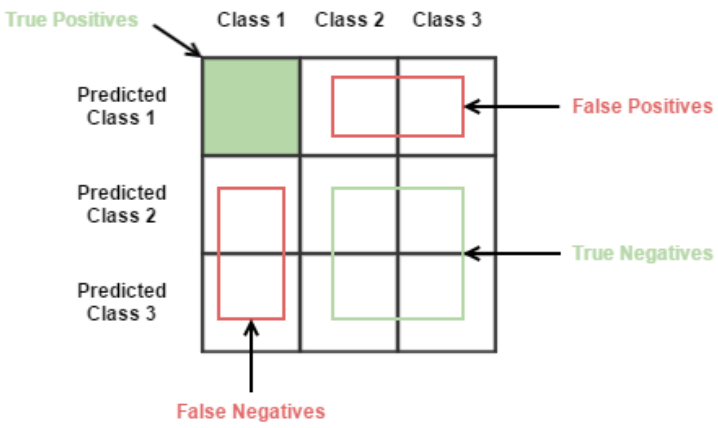


Figure 2.7: Confusion Matrix

Recall: Percentage of relevant examples

$$recall = \frac{tp}{tp+fn} \tag{2.18}$$

Specificity: Percentage of negative examples that are correctly identified

$$specificity = \frac{tn}{tn+fp} \tag{2.19}$$

Precision: Percentage of positive examples that are correctly identified

$$precision = \frac{tp}{tp+fp} \tag{2.20}$$

For regression problems a different set o measures is used.

The Root-mean-square Error (RMSE) measures the distance between the observed and predicted values of the model (Equation 2.21).

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{2.21}$$

where

y_i : actual values

$\hat{y}_i$: predicted values

Other relevant measures can be used, such as the Pearson Correlation Coefficient (Equation 2.22) and the Coefficient of Determination (Equation 2.23). Contrary to RMSE, the Pearson Correlation Coefficient and Coefficient of Determination are independent of scale, where 1 represents a perfect prediction and 0 means that there is absolutely no linear correlation.

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{2.22}$$

where

x_i : actual values

y_i : predicted values

$\bar{x}$: mean of actual values

$\bar{y}$: mean of predicted values

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{2.23}$$

where

y_i : actual values

$\hat{y}_i$: predicted values

$\bar{y}$: mean of actual values

2.3 Chapter Summary

In this chapter we have reviewed machine learning methods and techniques relevant for the thesis work. Most of the experiments in the thesis involve the use of propositional algorithms. Some of them were solely used with the aim of obtaining accurate models (NN, SVM) and have the extra objective of constructing comprehensible models (RF, ILP). There is little study that involves relational algorithms for drug sensitivity in cancer cells, being one of the key aspects of this type of algorithms the ability to use complex structures and relations between them as input data.

Chapter 3

Experiments

This chapter reports on the experimental work we have done to improve the original work by [MIG+13]. We have replicated their original results and have also achieved improvements. We have improved on their regression models and we have also transformed the original regression problem into a classification one.

We empirically evaluated several classification algorithms, some of which produce comprehensible models.

3.1 The Data Set

The data used in our experiments is the result of the work done in the "Genomics of Drug Sensitivity in Cancer" project [GEH+12], and its pre-processing follows the same approach used in [MIG+13]. The original data set, publicly available in the "Genomics of Drug Sensitivity in Cancer" project website, consists of measured IC50 values for various cell lines and compound pairs. It contains 639 different cell lines, each with 77 gene mutation properties. Each cell line also has information about its microsatellite instability status (MIS), cancer type and correspondent tissue. The IC50 value is available in its natural logarithmic form, ranging from -18.92 (6.07E-9 raw form) to 15.27 (4.28E6 raw form). Each gene mutation is described by its sequence variation and copy number variation.

The data set contains 131 drugs which translates to 83709 potential IC50 values. The information in the data set is not complete. In the first version of the data set, about 58% of the IC50 values, for each cell-drug pair, are available (Figure 3.1). In an updated version some values were added, amounting to about 18% of the total number of results.

The pre-processing was made having in mind a resulting data set with both cell mutation properties and drug properties, in this case, molecular descriptors and fingerprints generated with the same version of PaDEL used in [MIG+13]. All the data was therefore compiled in different instances, each having one IC50 value and correspondent cell and drug properties.

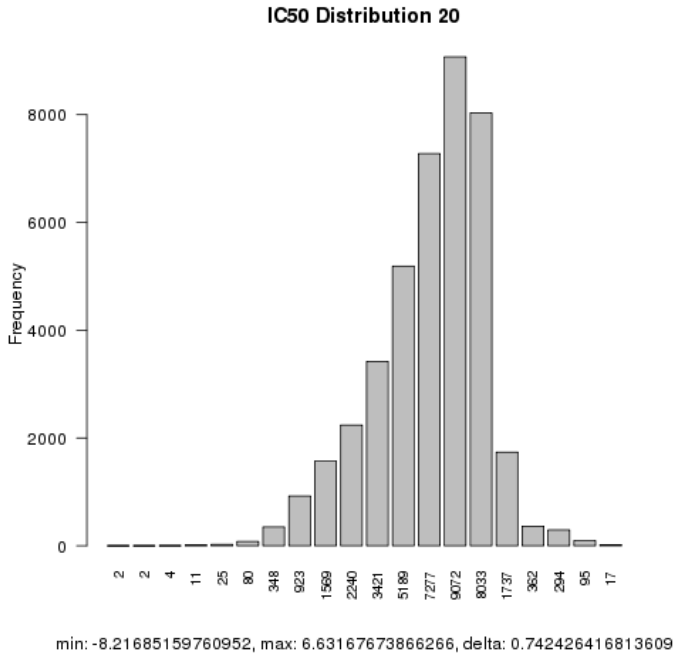


Figure 3.1: IC50 Distribution

In a first step of the pre-processing, all the missing values were discarded. The oncogene properties were transformed into binary and ternary values except for the microsatellite instability status which is already represented as such, being 0 if stable and 1 if unstable. For the sequence variation, the values with any change in protein sequence (p.*) were given the value 1, and 0 if a wild type (wt). For the copy number variation, the values with amplification (>=8) were given the value 1, the wild type values (0<cn<8) were given the value 0, and for the values with no copy number (nci) it was given the value -1. Any property (sequence variation and copy number variation) not available for any oncogene was removed from all cells. Any cell with more than 15 missing features was removed. Initially there were 77 sequence variations, 77 copy number variations and 1 microsatellite value for each cell, which results in a total of 155 cell features. After removing all unavailable features, the total amount of features was reduced to 142.

For each compound, a SMILES was generated using the API available in the Pubchem website (http://pubchem.ncbi.nlm.nih.gov/), using the corresponding drug Pubchem Id [WXS+09]. The SMILES data was then used to generate the molecular descriptors and fingerprints for each drug, using the PaDEL software.

At this point all the drugs for which the SMILES information was unavailable were removed from the data set. The 2D and 3D descriptors and fingerprints were generated using the default properties of the PaDEL software. Some descriptors and fingerprints were not possible to be calculated by PaDEL, so the drugs were reduced to 110 as a final value. The total amount of features for the compounds at this point was 1603. The missing features were then removed from all the compounds in the data set, as well as the features with the same value for all the compounds,

resulting in 790 final features.

The IC50 values were converted from natural log to base 10 log, for comparison purposes, since it was the base used in [MIG+13]. So, the converted IC50 values range from -8.22 to 6.63.

The final amount of cell line features was 142, and the final amount of drug features was 790, resulting in a total of 932 features plus the IC50 value. The final data set resulted in 40691 instances for the first version, and an additional 15578 for the updated one.

3.2 Methods and Algorithms

As in [MIG+13], we have used both Neural Networks and Random Forests algorithms as the base experiments. For assessing the models we have used 8-fold cross-validation. So the data from the first version of the data set (as described in [MIG+13]) was divided into 8 different bins. In each iteration, 6 bins are used for training, 1 bin for testing and 1 bin for cross-testing (Figure 3.2).

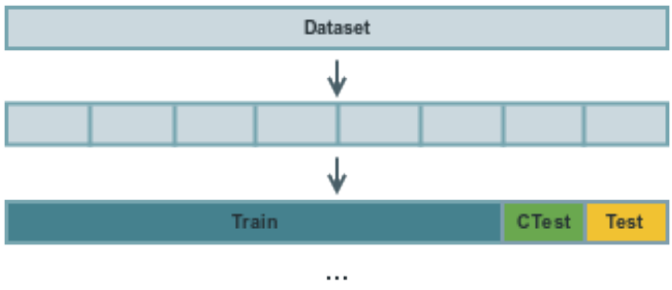


Figure 3.2: 8-fold Cross-Validation with Cross-Testing

This last bin is used for parameter tuning, testing various parameter values, such as the number of hidden layers in the Neural Networks, in each iteration, and selecting the one with the smaller error. For the second stage algorithms this bin is disposed in the first tests, so the parameter tuning is done in a second step.

Again, according to [MIG+13], two versions of the data set splitting were generated, one being the random splitting of the instances in 8 bins with equal size, and the other divided so that each training, testing and cross-testing don't share any cell line, which was called "stringent". The stringent form of the data set aims to show the predictive power of the models.

After the replication of the original results we have made a set of new experiments. In this new set of experiments we made parameter tuning for the Random Forest, varying the number of trees to grow and the number of variables randomly sampled as candidates at each split.

We also introduced the Support Vector Machines (SVM) algorithm, also with parameter tuning for the type of kernel, cost and gamma.

These algorithms were trained and tested both with the stringent and non stringent data set versions. We also made a blind test, using the additional instances of the updated version of the data set as the test set (with the extra 18% examples).

In this section we detail the replicated experiments from [MIG+13] (NN and RF without parameter tuning and using the default values) as well as the respective parameter tuning and the SVM as an additional algorithm. In the following sections we show the results of the same algorithms for variations of the data set, introducing molecular substructures as new features.

We also detail the experiments where we performed classification using the same algorithms used for regression, as well as Inductive Logic Programming (ILP) algorithm.

For the algorithms performance evaluation present in this section, we measured the RMSE, the Pearson Correlation Coefficient and the Coefficient of Determination. This metrics are calculated gathering the actual and predicted IC50 values of each fold for the tests. The reported results show the average and standard deviation for each metric.

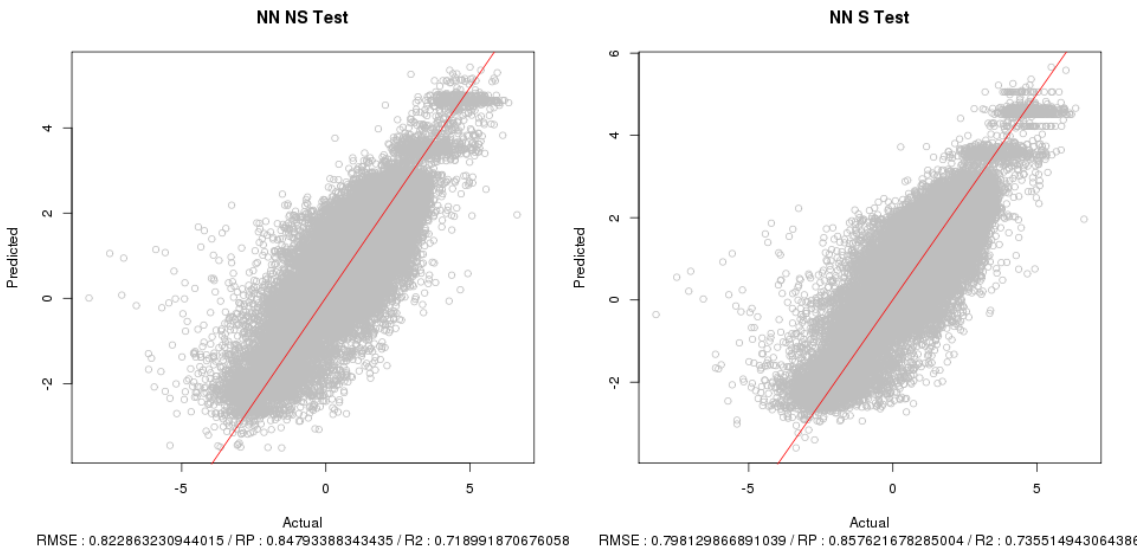
3.2.1 Result Replication

Neural Networks

For this algorithm we used the same version of Encog Machine Learning Framework for Java as in [MIG+13]. To perform the algorithm's parameter tuning, the number of iterations range from a minimum of 100 to a maximum of 400 and the number of hidden layers from 1 to 30. The raw IC50 values were normalized and substituted using the function:

$$norm(y) = \frac{1}{1+y^{-0.1}} \tag{3.1}$$

For comparison purposes, the predicted values are denormalized and converted back to log 10 form after running the models.



(a) Neural Networks Non Stringent Results

(b) Neural Networks Stringent Results

Figure 3.3: Neural Networks Test Results

	RMSE	RP	R2
Non Stringent	0.82 +/- 0.014	0.85 +/- 0.003	0.72 +/- 0.005
Stringent	0.8 +/- 0.013	0.86 +/- 0.005	0.74 +/- 0.009

Table 3.1: Neural Networks Results

As we can see in the results (Figure 3.3a, Figure 3.3a and Table 3.1), the values fall within the expected, comparing them to the original. The stringent and non stringent results, for this algorithm, show little difference, proving the performance efficiency.

Using the additional data set, created from the updated data, we performed a blind test using our models.

The results (Table 3.2) show an expected lower performance. This can be explained by the introduction of new cell lines and compounds that were not present in the training data.

	RMSE	RP	R2
Non Stringent Blind Test	0.91 +/- 0.005	0.8 +/- 0.002	0.65 +/- 0.003
Stringent Blind Test	0.9 +/- 0.003	0.81 +/- 0.001	0.65 +/- 0.002

Table 3.2: Neural Networks Blind Tests Results

Random Forest

For this algorithm we used the R's library package "randomForest" [R C15][LW02]. We conducted an experiment with 500 trees and the default values for the remaining algorithm's parameters.

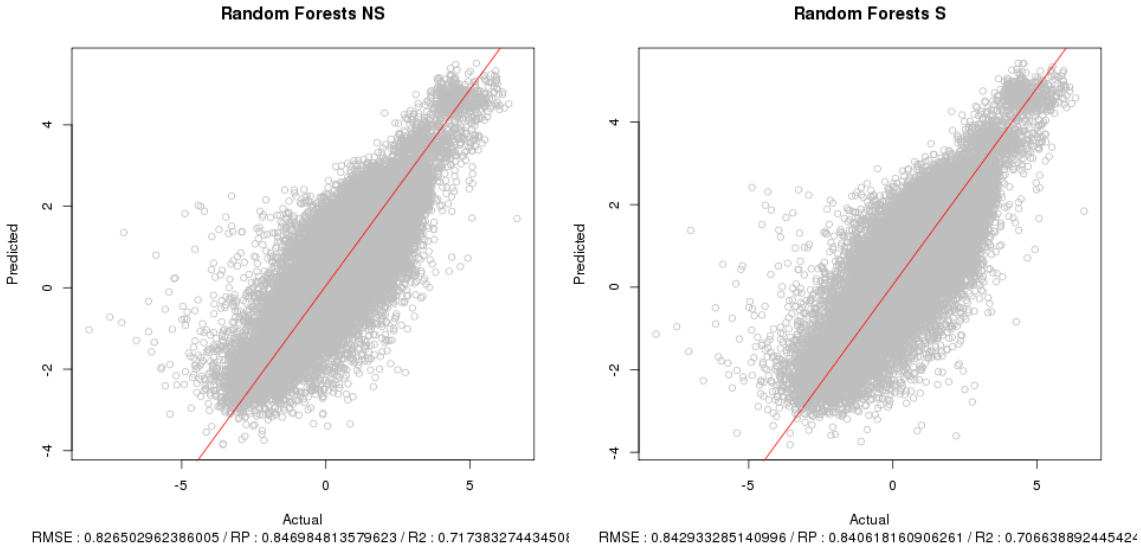
	RMSE	RP	R2
Non Stringent	0.83 +/- 0.014	0.85 +/- 0.004	0.72 +/- 0.006
Stringent	0.84 +/- 0.018	0.84 +/- 0.008	0.71 +/- 0.013

Table 3.3: Random Forest Results

The results (Figure 3.4a, Figure 3.4b and Table 3.3), show, as in the previous test, values that fall within the expected. The stringent and non-stringent show almost no differences in performance.

After training our models we calculated the importance of each feature, which corresponds to the total decrease in node impurities from splitting on each variable, in this case (regression), the residual sum of squares. This values, obtained by each model, were then averaged and sorted.

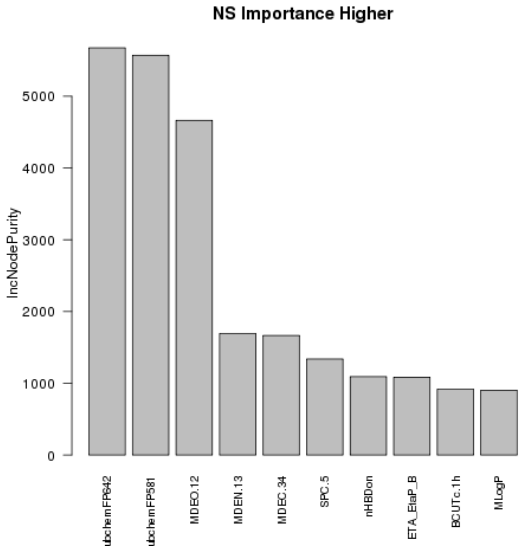
In Figures 3.5 and 3.6, we can see the most important features in both stringent and non-stringent tests.



(a) Random Forest Non Stringent Results (b) Random Forest Stringent Results

Figure 3.4: Random Forest Test Results

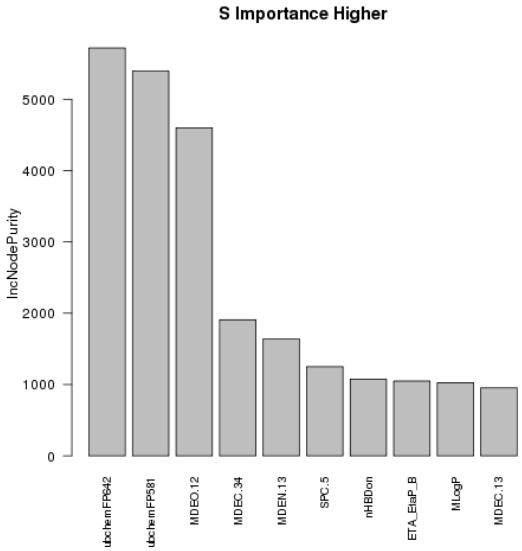
All of the top most important features belong to the compound features, being fingerprints the two most important.



Feature	IncNodePurity
PubchemFP642	5674.09
PubchemFP581	5570.68
MDEO.12	4663.65
MDEN.13	1692.19
MDEC.34	1666.11
SPC.5	1338.39
nHBDon	1091.73
ETA_EtaP_B	1082.55
BCUTc.1h	916.17
MLogP	902.71

Figure 3.5: Random Forest Importance Non Stringent Top 10 Results

In Table 3.4 we can see the results for the blind tests, using our models, which shows the worst results obtained so far.



Feature	IncNodePurity
PubchemFP642	5719.00
PubchemFP581	5394.85
MDEO.12	4599.63
MDEC.34	1908.00
MDEN.13	1637.08
SPC.5	1250.56
nHBDon	1074.86
ETA_EtaP_B	1049.02
MLogP	1024.70
MDEC.13	953.55

Figure 3.6: Random Forest Importance Stringent Top 10 Results

	RMSE	**RP**	**R2**
Non Stringent Blind Test	0.94 +/- 0.002	0.79 +/- 0.001	0.62 +/- 0.001
Stringent Blind Test	0.95 +/- 0.005	0.79 +/- 0.003	0.62 +/- 0.004

Table 3.4: Random Forest Results

3.2.2 First Improvement - Parameter Tuning

Random Forest

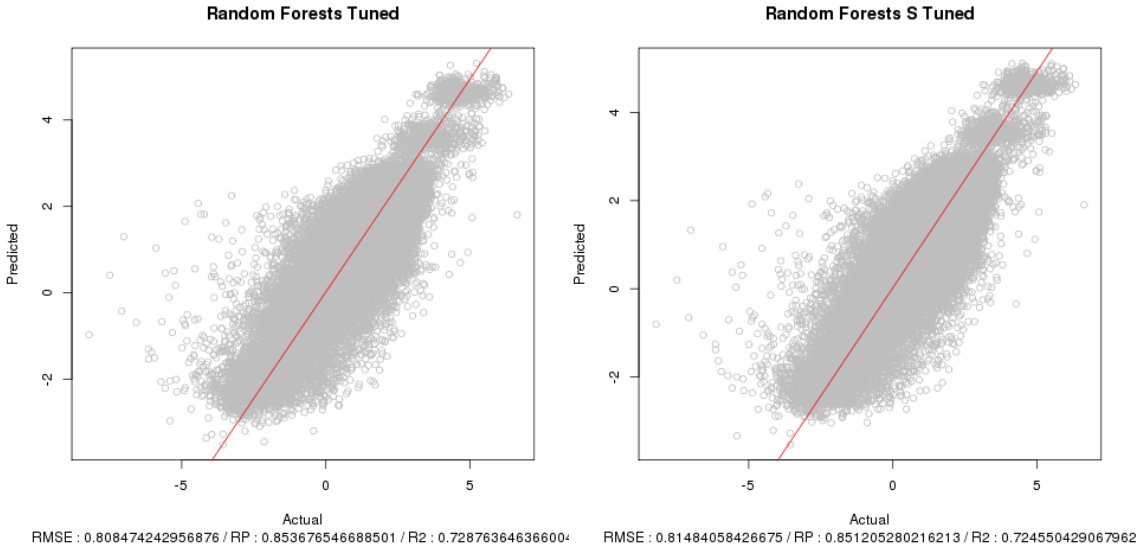
After the replicated experiments we decided to improve the Random Forest results. To do this, we changed multiple key parameter, run the training and testing again, but at each iteration, we compare the error that results from the testing of the cross-testing bin. We then chose the model that results in the lowest error.

For comparison purposes, the cross-test bin was discarded in the default run.

To perform this parameter tuning we vary the number of trees with 128, 256 and 512, and the number of variables randomly sampled as candidates at each split with an initial value of 25% of the number of features divided by 3. Then we vary an increase of this value for 3 iterations at each number of trees.

	RMSE	**RP**	**R2**
Non Stringent	0.81 +/- 0.015	0.85 +/- 0.003	0.73 +/- 0.005
Stringent	0.81 +/- 0.017	0.85 +/- 0.007	0.73 +/- 0.012

Table 3.5: Tuned Random Forest Normal Test Results



(a) Tuned Random Forest Non Stringent Results (b) Tuned Random Forest Stringent Results

Figure 3.7: Tuned Random Forest Test Results

In Figures 3.7a and 3.7b and in Table 3.5 we show the performance of the algorithm after performing the parameter tuning.

We can see that the error was slightly decreased for both stringent and non-stringent versions, as well as for the blind tests (Table 3.6)

	RMSE	**RP**	**R2**
Non Stringent Blind Test	0.9 +/- 0.002	0.81 +/- 0.001	0.65 +/- 0.001
Stringent Blind Test	0.91 +/- 0.003	0.81 +/- 0.001	0.65 +/- 0.002

Table 3.6: Tuned Random Forest Blind Test Results

Support Vector Machines

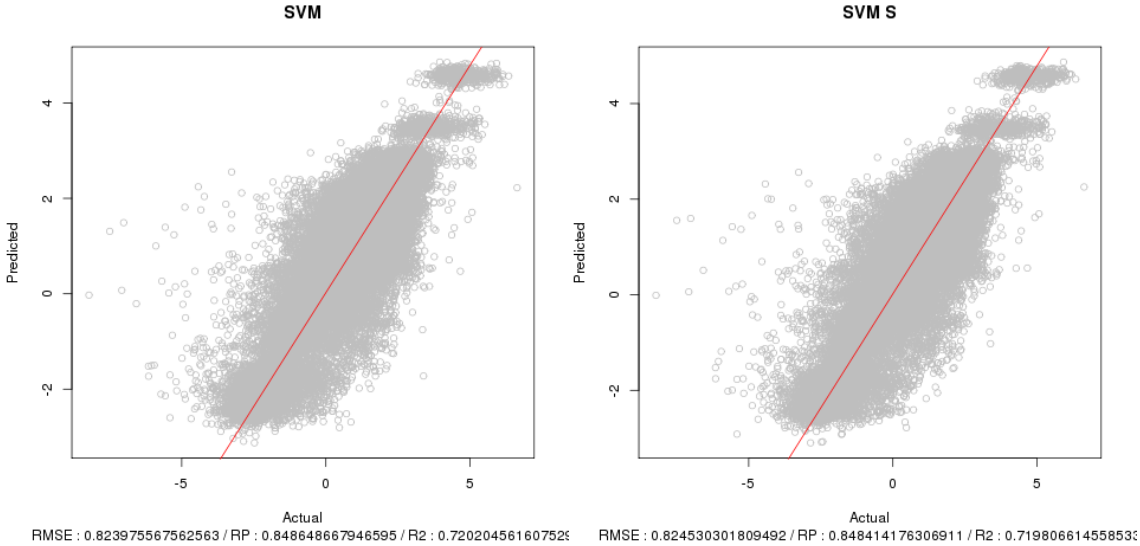
The SVM was chosen to be the additional propositional algorithm used in our experiments, as it has not only shown empirically seamless performance in cancer and drug discovery [ZBI+03] related studies, but also due to its popularity. For this experiment we used the SVM Light implementation [Joa98]. In an initial iteration we used the radial kernel, a gamma of 1 divided by the number of features and a cost of 1, keeping the rest of the parameters as default. These were chosen to be the most approximate form of default parameters used not only in this implementation but also in the R's package "e1071" [MDH+14]. The results of this test, as well as the blind tests, can be seen in Table 3.7.

As in the Random Forest experiment, we also performed parameter tuning. To do this, we tested multiple kernel types (polynomial, radial and sigmoid) and, for all kernels, we varied the

	RMSE	RP	R2
Non Stringent	0.83 +/- 0.016	0.84 +/- 0.004	0.71 +/- 0.006
Stringent	0.84 +/- 0.018	0.84 +/- 0.008	0.71 +/- 0.013
Non Stringent Blind Test	0.88 +/- 0.001	0.82 +/- 0.0	0.67 +/- 0.001
Stringent Blind Test	0.88 +/- 0.001	0.82 +/- 0.001	0.67 +/- 0.001

Table 3.7: SVM Results

gamma by the double of the initial one. For the radial kernel we also varied the cost with 1, 2 and 4, and an additional triple of the initial gamma value.



(a) Tuned SVM Non Stringent Results

(b) Tuned SVM Stringent Results

Figure 3.8: Tuned Support Vector Machines Test Results

	RMSE	RP	R2
Non Stringent	0.82 +/- 0.016	0.85 +/- 0.004	0.72 +/- 0.006
Stringent	0.82 +/- 0.019	0.85 +/- 0.008	0.72 +/- 0.013
Non Stringent Blind Test	0.89 +/- 0.002	0.81 +/- 0.001	0.66 +/- 0.001
Stringent Blind Test	0.89 +/- 0.005	0.81 +/- 0.002	0.66 +/- 0.004

Table 3.8: Tuned SVM Results

Conclusions of First Improvement

In Table 3.9 we show the results summary of all the experiments mentioned in this section. The replicated results (Neural Networks and Random Forest) fall within the expected values. Since the

RMSE for the Neural Networks is the lowest, and therefore better, this will be used as the base result for the performance of the other algorithms.

Comparing the results obtained for non-stringent and stringent versions, these are overall, as expected, worst for the stringent version. This can be explained by the large difference between the cell's features of the training and test sets. Moreover the results of blind tests are the worst due to the introduction of new cell lines and drugs that were not used in training. These results, despite having a higher error, they have a very close accuracy.

			Non Stringent	Stringent	Non Stringent Blind Test	Stringent Blind Test
NN	**Results**	**RMSE**	0.82 +/- 0.014	0.8 +/- 0.013	0.91 +/- 0.005	0.9 +/- 0.003
		RP	0.85 +/- 0.003	0.86 +/- 0.005	0.8 +/- 0.002	0.81 +/- 0.001
		R2	0.72 +/- 0.005	0.74 +/- 0.009	0.65 +/- 0.003	0.65 +/- 0.002
RF	**Results**	**RMSE**	0.83 +/- 0.014	0.84 +/- 0.018	0.94 +/- 0.002	0.95 +/- 0.005
		RP	0.85 +/- 0.004	0.84 +/- 0.008	0.79 +/- 0.001	0.79 +/- 0.003
		R2	0.72 +/- 0.006	0.71 +/- 0.013	0.62 +/- 0.001	0.62 +/- 0.004
	Tuned	**RMSE**	0.81 +/- 0.015	0.81 +/- 0.017	0.9 +/- 0.002	0.91 +/- 0.003
		RP	0.85 +/- 0.003	0.85 +/- 0.007	0.81 +/- 0.001	0.81 +/- 0.001
		R2	0.73 +/- 0.005	0.73 +/- 0.012	0.65 +/- 0.001	0.65 +/- 0.002
SVM	**Results**	**RMSE**	0.83 +/- 0.016	0.84 +/- 0.018	0.88 +/- 0.001	0.88 +/- 0.001
		RP	0.84 +/- 0.004	0.84 +/- 0.008	0.82 +/- 0.0	0.82 +/- 0.001
		R2	0.71 +/- 0.006	0.71 +/- 0.013	0.67 +/- 0.001	0.67 +/- 0.001
	Tuned	**RMSE**	0.82 +/- 0.016	0.82 +/- 0.019	0.89 +/- 0.002	0.89 +/- 0.005
		RP	0.85 +/- 0.004	0.85 +/- 0.008	0.81 +/- 0.001	0.81 +/- 0.002
		R2	0.72 +/- 0.006	0.72 +/- 0.013	0.66 +/- 0.001	0.66 +/- 0.004

Table 3.9: Results Summary

3.3 Graph Mining

A study was conducted to understand the distribution of compound's molecules frequent substructures in the data set along the IC50 distribution. In Figure 3.9 we show the amount of results divided in different numbers of bins. Each bin represents an interval, of equal length, of the IC50 values.

The first goal was to find groups of molecule substructures (fragments) that can be associated with the good (low) and bad (high) IC50 values. The frequent substructures were calculated using Xifeng Yang's gSpan implementation (Graph-Based Substructure Pattern Mining [YH02]).

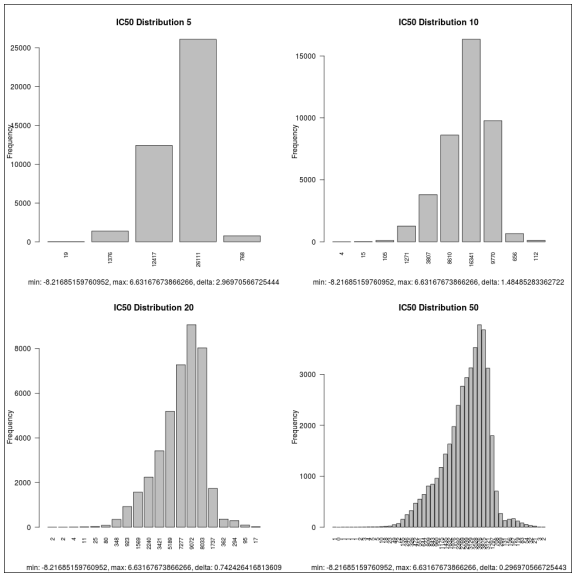


Figure 3.9: IC50 Distribution

The input used for the gSpan execution was the drug molecules graphs data, created using the SMILES information. The minimum support range used was from 20% to 60% increasing 5% in each iteration. The support stands for the minimum percentage of the molecules in which the substructures occur. The minimum size of the fragments range from 3 to 20 vertexes. The number of fragments found was limited from 200 to 500.

Fulfilling this criteria, 10 different groups of fragments were found (Figure 3.10). This resulted in 10 new data sets that have, as additional features, the presence of each fragment ("1" present and "0" not present) found previously, for each example's compound. It was also created another data set version in which the descriptors and fingerprints were removed, having as the only compounds properties, the fragment information.

The results were used to create an histogram of the fragment occurrence in each bin (Figure 3.10), in other words, the number of molecule fragments that are part of the drug for which the IC50 was calculated and is present in the bin interval.

Having in mind that the goal is to find a differentiation in the fragments present in the higher and lower ranges of IC50, we decided to discard fragments that have a low frequency difference

Support	Number of Vertexes	Number of Fragments
0.2	7	420
0.2	8	237
0.25	5	463
0.25	6	347
0.25	7	230
0.3	3	348
0.3	4	315
0.3	5	255
0.35	3	249
0.35	4	219

Table 3.10: Groups of Fragments

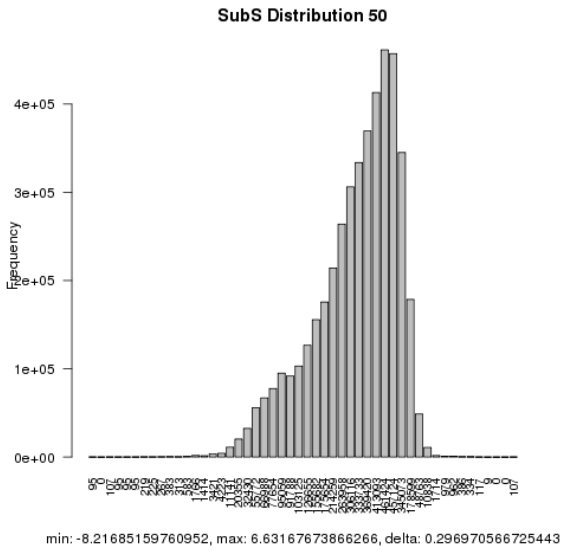
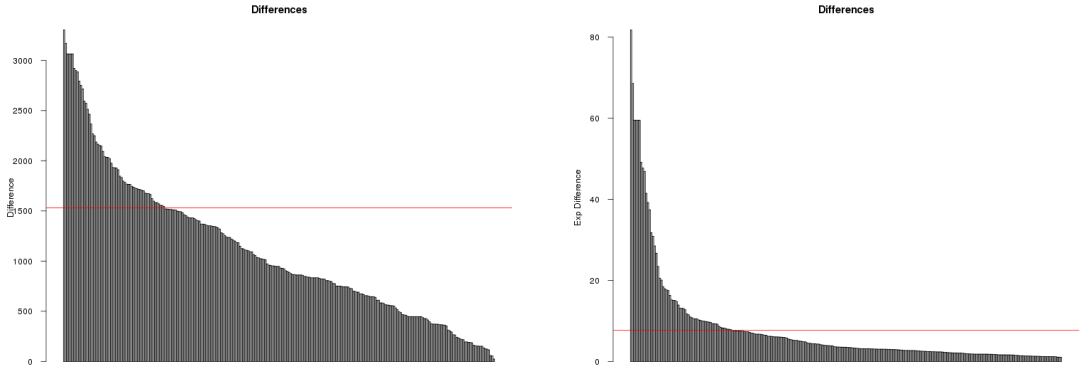


Figure 3.10: Fragment Number Differences (50 bins)

between these ranges. The high and low ranges were limited by measuring the maximum value of the histogram (Figure 3.10) and discarding, besides this bin, the 4 bins behind and ahead. To calculate the threshold we created an histogram that measures the difference value between ranges for each fragment (Figure 3.11a).

For better understand the decay in the histogram, we decided to calculate the exponential value of each frequency (Equation 3.2, Figure 3.11b).

$$f(x) = e^{\frac{x}{750}} \tag{3.2}$$



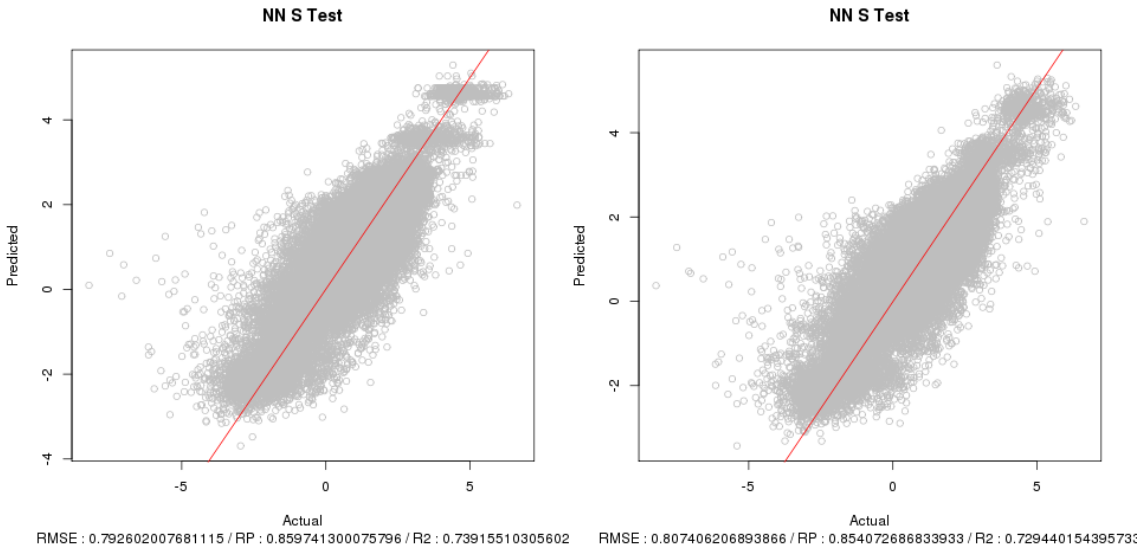
(a) Differences Histogram (b) Differences Exponential Histogram

Figure 3.11: Differences between high and low IC50 values fragment number histogram

The result corresponds to a "long tail" like division, making clear the approximate threshold value.

The divisor value in Formula 3.2 was chosen to avoid R's supported maximum integer value and that better represents the long tail given the scale.

Therefore, given the threshold, we proceeded to discard, from the data sets, the fragment values which are below this value.



(a) Descriptors and Fragments Best Result (b) Fragments Only Best Result

Figure 3.12: Fragments data sets Best Results

We then decided to run the same algorithms used previously for all the 20 data sets. As in the previous runs, the data set was divided into 8 bins for the stringent and non stringent versions, and the validation was made with 8-fold cross-validation. The parameter range for the parameter

tuning was kept unchanged. The additional values from the updated version of the original data set were also used for the blind tests.

The sample results shown in the Tables 3.11 and 3.12 match the best results obtained for both the data sets versions. The best result for the version that contains all the features (descriptors, fingerprints and fragments) match the data set obtained with 20% support and a minimum of 8 vertexes. On the other hand the best result for the version that contains only fragments, match the data set obtained with 30% support and a minimum of 3 vertexes. In Figures 3.12a and 3.12b we show the best results within all the algorithms and type of splitting (stringent and non-stringent) that were tested for both types of data sets.

The remaining results can be found in the Appendix B, Section B.2.1.

			Non Stringent	Stringent	Non Stringent Blind Test	Stringent Blind Test
NN	**Results**	**RMSE**	0.82 +/- 0.015	0.79 +/- 0.013	0.88	0.88
		RP	0.85 +/- 0.003	0.86 +/- 0.007	0.81	0.81
		R2	0.72 +/- 0.005	0.74 +/- 0.011	0.66	0.66
RF	**Results**	**RMSE**	0.83 +/- 0.014	0.84 +/- 0.017	0.94	0.94
		RP	0.85 +/- 0.004	0.84 +/- 0.009	0.79	0.79
		R2	0.72 +/- 0.006	0.71 +/- 0.016	0.63	0.63
SVM	**Results**	**RMSE**	0.84 +/- 0.016	0.84 +/- 0.017	0.88	0.88
		RP	0.84 +/- 0.004	0.84 +/- 0.01	0.82	0.82
		R2	0.71 +/- 0.006	0.71 +/- 0.017	0.67	0.67
	Tuned	**RMSE**	0.82 +/- 0.016	0.82 +/- 0.018	0.89	0.88
		RP	0.85 +/- 0.004	0.85 +/- 0.01	0.82	0.82
		R2	0.72 +/- 0.006	0.72 +/- 0.016	0.67	0.67

Table 3.11: Descriptors and Fragments Results; Support 20%; Minimum Vertexes 8

			Non Stringent	Stringent	Non Stringent Blind Test	Stringent Blind Test
NN	**Results**	**RMSE**	0.83 +/- 0.013	0.81 +/- 0.013	0.88	0.88
		RP	0.85 +/- 0.004	0.85 +/- 0.009	0.81	0.81
		R2	0.72 +/- 0.007	0.73 +/- 0.015	0.66	0.66
RF	**Tuned**	**RMSE**	0.81 +/- 0.014	0.81 +/- 0.017	0.91	0.91
		RP	0.85 +/- 0.003	0.85 +/- 0.009	0.81	0.81
		R2	0.73 +/- 0.005	0.72 +/- 0.016	0.65	0.65
SVM	**Results**	**RMSE**	0.87 +/- 0.017	0.89 +/- 0.022	0.92	0.92
		RP	0.83 +/- 0.005	0.82 +/- 0.01	0.8	0.8
		R2	0.69 +/- 0.008	0.68 +/- 0.016	0.63	0.63
	Tuned	**RMSE**	0.8 +/- 0.015	0.84 +/- 0.021	0.9	0.89
		RP	0.86 +/- 0.003	0.84 +/- 0.009	0.81	0.81
		R2	0.74 +/- 0.006	0.71 +/- 0.016	0.66	0.66

Table 3.12: Fragments Only Results; Support 30%; Minimum Vertexes 3

3.4 Classification

Another interesting experiment is to show how the same algorithms and data sets, used in the previous section, would handle classification, as well as being used as a result basis for the ILP experiment evaluation. Therefore we decided to divide the data into 2 different classes, "good" and "bad".

The data discretization and data set generation was done by calculating the tertiles of the IC50 distribution, which were used as upper and lower threshold values in the decision of the class. Therefore, examples of the data set are considered as "good" below the bottom and "bad" above the upper threshold. The remaining elements are disposed.

The discretization results in a total of 27120 examples for the first data set and a total of 10427 examples for the additional one.

The tests involved 3 different data sets with different drug features, keeping the same cell line features unchanged across all of them. The first with the molecular descriptors and fingerprints (790 drug features), the second with only fragments information (136 drug features) and a third with both (926 drug features). The fragments are the result of the graph mining experiment done previously in the generation of the various data sets.

The fragments information used was extracted from the data set that contained the highest number of substructures after the graph mining experiment.

We used the same algorithms, that is, Neural Networks, Random Forests and Support Vector Machines for the classification of the examples. We also made the division in their stringent and non-stringent versions for each one of the 3 data sets.

The validation is also performed using 8-fold cross-validation. However, since that, in this experiment, we are not performing parameter tuning, the cross-training bins were merged with the training ones.

In the following section we show the best results of the experiment, for each data set, using confusion matrices (Tables 3.13, 3.15 and 3.17) and metrics arising therefrom (Tables 3.14, 3.16, 3.18), such as the accuracy, recall, specificity and precision.

3.4.1 Results

Molecular Descriptors data set

		Reference	
		bad	good
Prediction	bad	12148	1252
	good	1413	12307

Table 3.13: RF Non Stringent Confusion Matrix

Accuracy	0.9017
Recall	0.8958
Specificity	0.9077
Precision	0.9066

Table 3.14: RF Non Stringent Results

Fragments data set

		Reference	
		bad	good
Prediction	bad	11780	2075
	good	1781	11484

Table 3.15: RF Stringent Confusion Matrix

Accuracy	0.8578
Recall	0.8687
Specificity	0.8470
Precision	0.8502

Table 3.16: RF Stringent Results

Descriptors and Fragments data set

		Reference	
		bad	good
Prediction	bad	12304	1198
	good	1257	12361

Table 3.17: NN Stringent Consfusion Matrix

Accuracy	0.9095
Recall	0.9073
Specificity	0.9116
Precision	0.9113

Table 3.18: NN Stringent Confusion Results

3.5 Inductive Logic Programming

Using all the features from the data sets used for classification (Section 3.4) we performed classification using the ILP algorithm. For this execution we used the Aleph implementation [Sri03].

The data preparation involves the creation of the background knowledge (BK) and the creation of positive and negative examples.

The background knowledge was created gathering data on descriptors, fingerprints and drug fragments, as well as the cell lines features, for each drug and cell line present in the data sets,

using Prolog language facts and rules. The positive and negative examples were generated as simple facts, indicating the class that each example belongs to.

In this experiment we used both the data sets obtained from the first division obtained from the non-stringent splitting type.

Train Set Results

		Reference	
		bad	good
Prediction	bad	9887	5418
	good	1957	6468

Table 3.19: ILP Non-Stringent Train Set Confusion Matrix

Accuracy	0.6892
Sensitivity	0.8348
Specificity	0.5442
Precision	0.646

Table 3.20: ILP Non-Stringent Train Set Results

Test Set Results

		Reference	
		bad	good
Prediction	bad	1411	763
	good	304	912

Table 3.21: ILP Non-Stringent Test Set Confusion Matrix

Accuracy	0.6852
Sensitivity	0.8227
Specificity	0.5445
Precision	0.649

Table 3.22: ILP Non-Stringent Test Set Results

3.5.1 Rules

In this section we interpret the rules that were created by the ILP algorithm from both the data sets.

Rule 1: pubchemfp567(Compound), pubchemfp516(Compound), pubchemfp692(Compound) ; Positive cover = 3571, Negative cover = 77.

"If the compound molecule has the substructure O-C-C-O, the substructure [#1]-C=C-[#1] and the substructure O=C-C-C-C-C-C, then the IC50 is considered as good (98% of the covered examples)."

Rule 2: pubchemfp188(Compound) ; Positive cover = 4069, Negative cover = 2915.

"If the compound molecule has 2 or more saturated or aromatic heteroatom-containing ring of size 6, then the IC50 is considered good (58% of the covered examples)."

Rule 3: pubchemfp308(Compound) ; Positive cover = 7462, Negative cover = 2833

"If the compound molecule has the simple atom pair O-H, the the IC50 is considered good (72% of the covered examples)."

3.6 Results Discussion

In this section we discuss the results obtained in our experiments together with the limitations of our work.

Performance of the different algorithms and settings is compared using the average of the RMSE results, in the case of regression data sets, and accuracy for the classification data sets.

3.6.1 Regression

Improving the results on the original data

In our experiments, after the result replication, we started for trying to improve the results that were obtained in [MIG+13]. Our parameter tuning methods have demonstrated that they were successful, although the error reduction might be considered fairly low.

For the Random Forest we observed an improvement of approximately 2.4% for the non-stringent tests and an improvement of 3.6% for the stringent. Comparing the parameter tuning results against the Neural Networks, we observed that these became the best results obtained for the non-stringent tests. We obtained an improvement of 1.2% but an increase in error of 1.2% for the non-stringent and stringent tests respectively.

Support Vector Machines

The introduction of this algorithm demonstrated that it is a fairly good alternative to the Neural Networks and Random Forests, since its results show a very close performance. We consider that the only drawback of the algorithm might be its time complexity, since it was the algorithm that, in all runs, consumed the most time.

Comparing the parameter tuning results against the Neural Networks, we observe no change for the non-stringent tests and a increase in the error of about 2.5% for the stringent ones.

Using Molecular Fragments

When adding the fragment features to our original data set we demonstrated that there was a small increase in performance, both in regression and classification.

When substituting all of the compound features with fragments information only, we can see a considerable increase in the error of regression experiments results, for some groups of fragments. On the other hand we obtained some results that were very close to the results obtained for the original data set, proving the importance of this type of features in our experiment. We can also note that, considering each group of fragments, the error slightly decreases with the increased support for a low minimum substructure size.

It is important to note that the amount of molecular descriptors and fingerprints is 790, for the original data set, while the number of fragments (total number of compound features) for the best result is 97.

Using the Graph Mining algorithm we were looking for fragments (molecular sub-structures) that discriminate efficient drugs from inefficient ones. Therefore, it would be fair to expect that these fragment would be a valuable feature set. While finger prints are generated more "blindly" we must use a lot of finger prints in the hope that the relevant ones appear in the lot.

3.6.2 Classification

The accuracy, for the classification results, ranges from 0.8 to 0.9, which is quite good overall. The runs using both descriptors and fingerprints and fragments produced the best results with an accuracy of 0.9095.

As ILP is concerned we have made a very limited amount of experiments. ILP requites very long times to run. Despite the accuracy results were worse than the ones of propositional learners, ILP found a set of very interesting simple rule with high coverage. These rules, together with this line of investigation could be very rewarding to the biochemist experts.

Chapter 4

Conclusion

The ability to accurately predict or classify *in silico* the sensibility of a specific cancer cell to a given pharmaceutical, is a subject of major importance, not only for its immense potential in decreasing the repetitive, time-consuming and expensive work done *in vitro*, but also as a major step in personalized treatment.

This project aimed to take a step towards exploring new methodologies to increase the performance and understandability, which raises the opportunity of improvement in treatment.

We have introduced not only methods to improve results of previous work and a new propositional algorithm, but also explored possible additional information in terms of drugs molecular structure. We have also introduced the ILP algorithm, which can easily handle structured data, besides its capability to produce understandable rules, which can perform accurate previsions. Although our experiments with ILP were rather limited the results are very promising. We found a set of very simple rules with high predictive power and highly comprehensible.

Despite the limitations, the project offers the possibility to complement future work and create significance in the use of methodologies *in silico* to, ultimately, improve the quality of life of patients and treat the disease.

Limitations and Future Work

During the implementation of the algorithms, we have considered the maximum possible division of jobs, so it could run in parallel, regardless of the type of machine. Even so, all the training executions proved to be quite costly in terms of time. This became a disadvantage in terms of the number of tests that were possible to be performed in a timely manner. The amount of memory and computational power required has proved to be considerably high.

Therefore, this high demand for computational power has become the primary contributor to the impossibility of testing within the field of evolutionary computation as proposed.

One of our proposals for future work is to continue the project using evolutionary algorithms, comparing them with the results already obtained for propositional and ILP algorithms, both for regression and classification. This would make possible the evaluation of the performance of such algorithms against the ones tested in this work.

There is still room for improvement when it comes to parameter tuning, which is only limited by the computing power and associated time. Testing the algorithms with an increased variation of the parameters would result in a slightly more accurate prediction.

While replicating the tests we observed a big difference both in quality and quantity of the results mainly of PaDLE but also of the Encog Framework, using different versions of the software. Also we observed that during project implementation there was a new update of the data set publicly available on the "Genomics of Drug Sensitivity in Cancer" project [GEH+12] website. Therefore, the use of the latest software and latest data sets available, might not only improve the performance of the algorithms, but also produce more reliable results.

References

[AT04] Ted T Ashburn and Karl B Thor. Drug repositioning: identifying and developing new uses for existing drugs. *Nature reviews Drug discovery*, 3(8):673–683, 2004.

[BFSO84] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984.

[Bre01] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[Cha04] Arshad Chaudry. Cell culture. *The Science Creative Qwarterly*, Agosto 2004.

[CV95] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[Fis36] Ronald A Fisher. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188, 1936.

[FPSS96] Usama Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. The kdd process for extracting useful knowledge from volumes of data. *Communications of the ACM*, 39(11):27–34, 1996.

[GEH+12] Mathew J Garnett, Elena J Edelman, Sonja J Heidorn, Chris D Greenman, Anahita Dastur, King Wai Lau, Patricia Greninger, I Richard Thompson, Xi Luo, Jorge Soares, et al. Systematic identification of genomic markers of drug sensitivity in cancer cells. *Nature*, 483(7391):570–575, 2012.

[JIM+13] Ferlay J, Soerjomataram I, Ervik M, Dikshit R, Eser S, Mathers C, Rebelo M, Parkin DM, Forman D, and F. GLOBOCAN Bray. v1. *0, Cancer Incidence and Mortality Worldwide: IARC CancerBase No*, 11, 2013.

[Joa98] T. Joachims. Making large-scale svm learning practical. LS8-Report 24, Universität Dortmund, LS VIII-Report, 1998.

[K+95] Ron Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, volume 14, pages 1137–1145, 1995.

[KMLS92] Ross D King, Stephen Muggleton, Richard A Lewis, and MJ Sternberg. Drug design by machine learning: The use of inductive logic programming to model the structure-activity relationships of trimethoprim analogues binding to dihydrofolate reductase. *Proceedings of the national academy of sciences*, 89(23):11322–11326, 1992.

[LW02] Andy Liaw and Matthew Wiener. Classification and regression by randomforest. *R News*, 2(3):18–22, 2002.

[MDH+14] David Meyer, Evgenia Dimitriadou, Kurt Hornik, Andreas Weingessel, and Friedrich Leisch. *e1071: Misc Functions of the Department of Statistics (e1071), TU Wien*, 2014. R package version 1.6-4.

[MDR94] Stephen Muggleton and Luc De Raedt. Inductive logic programming: Theory and methods. *The Journal of Logic Programming*, 19:629–679, 1994.

[MFB08] Colin Mathers, Doris Ma Fat, and JT Boerma. *The global burden of disease: 2004 update*. World Health Organization, 2008.

[MIG+13] Michael P Menden, Francesco Iorio, Mathew Garnett, Ultan McDermott, Cyril H Benes, Pedro J Ballester, and Julio Saez-Rodriguez. Machine learning prediction of cancer cell sensitivity to drugs based on genomic and chemical properties. *PloS one*, 8(4):e61318, 2013.

[MP43] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[NK] Siegfried Nijssen and Joost N Kok. A quickstart in frequent structure mining can make a difference.

[PAP+14] Tapio Pahikkala, Antti Airola, Sami Pietilä, Sushil Shakyawar, Agnieszka Szwajda, Jing Tang, and Tero Aittokallio. Toward more realistic drug–target interaction predictions. *Briefings in bioinformatics*, page bbu010, 2014.

[Pow11] David Martin Powers. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. 2011.

[Qui86] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[Qui93] John Ross Quinlan. *C4. 5: programs for machine learning*, volume 1. Morgan kaufmann, 1993.

[R C15] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2015.

[RN95] Stuart Russell and Peter Norvig. Artificial intelligence: a modern approach. 1995.

[SK11] Philippe Sanseau and Jacob Koehler. Editorial: computational methods for drug repurposing. *Briefings in bioinformatics*, 12(4):301–302, 2011.

[Sri03] Ashwin Srinivasan. The Aleph Manual, 2003. Available from http://web.comlab.ox.ac.uk/oucl/research/areas/machlearn/Aleph.

[Wei88] David Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of chemical information and computer sciences*, 28(1):31–36, 1988.

[Wei90] David Weininger. Smiles. 3. depict. graphical depiction of chemical structures. *Journal of Chemical Information and Computer Sciences*, 30(3):237–243, 1990.

[WWW89] David Weininger, Arthur Weininger, and Joseph L Weininger. Smiles. 2. algorithm for generation of unique smiles notation. *Journal of Chemical Information and Computer Sciences*, 29(2):97–101, 1989.

REFERENCES

[WXS+09] Yanli Wang, Jewen Xiao, Tugba O Suzek, Jian Zhang, Jiyao Wang, and Stephen H Bryant. Pubchem: a public information system for analyzing bioactivities of small molecules. *Nucleic acids research*, 37(suppl 2):W623–W633, 2009.

[Yap11] Chun Wei Yap. Padel-descriptor: An open source software to calculate molecular descriptors and fingerprints. *Journal of computational chemistry*, 32(7):1466–1474, 2011.

[YH02] Xifeng Yan and Jiawei Han. gspan: Graph-based substructure pattern mining. In *Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on*, pages 721–724. IEEE, 2002.

[ZBI+03] Vladimir V Zernov, Konstantin V Balakin, Andrey A Ivaschenko, Nikolay P Savchuk, and Igor V Pletnev. Drug discovery using support vector machines. the case studies of drug-likeness, agrochemical-likeness, and enzyme inhibition predictions. *Journal of chemical information and computer sciences*, 43(6):2048–2056, 2003.

REFERENCES

Appendix A

Features

A.0.3 Cell Line Features

1. MIS
2. AKT2 CN
3. ALK SV
4. ALK CN
5. APC SV
6. APC CN
7. BRAF SV
8. BRAF CN
9. BRCA1 SV
10. BRCA1 CN
11. BRCA2 SV
12. BRCA2 CN
13. CCND1 CN
14. CCND2 CN
15. CCND3 CN
16. CDH1 SV
17. CDH1 CN
18. CDK4 CN
19. CDK6 CN
20. CDKN2A SV
21. CDKN2A CN
22. CDKN2C SV
23. CDKN2C CN
24. CDKN2a(p14) SV
25. CDKN2a(p14) CN
26. CTNNB1 SV
27. CTNNB1 CN
28. CYLD SV
29. CYLD CN
30. DICER1 SV
31. DICER1 CN
32. EGFR SV
33. EGFR CN
34. EP300 CN
35. ERBB2 SV
36. ERBB2 CN
37. EZH2 SV
38. EZH2 CN
39. FAM123B SV
40. FAM123B CN
41. FBXW7 SV
42. FBXW7 CN
43. FGFR2 CN
44. FGFR3 SV
45. FGFR3 CN
46. FLCN SV
47. FLCN CN
48. FLT3 SV
49. FLT3 CN
50. GNAS SV
51. GNAS CN
52. HRAS SV
53. HRAS CN
54. IDH1 SV
55. IDH1 CN

56. JAK2 SV
57. JAK2 CN
58. KDM5C SV
59. KDM5C CN
60. KDM6A SV
61. KDM6A CN
62. KDR SV
63. KDR CN
64. KIT SV
65. KIT CN
66. KRAS SV
67. KRAS CN
68. MAP2K4 SV
69. MAP2K4 CN
70. MDM2 CN
71. MET SV
72. MET CN
73. MLH1 SV
74. MLH1 CN
75. MLLT3 CN
76. MSH2 SV
77. MSH2 CN
78. MSH6 SV
79. MSH6 CN
80. MYC SV
81. MYC CN
82. MYCL1 CN
83. MYCN CN
84. NF1 SV
85. NF1 CN
86. NF2 SV
87. NF2 CN
88. NOTCH1 SV
89. NOTCH1 CN
90. NPM1 SV
91. NPM1 CN
92. NRAS SV
93. NRAS CN
94. NTRK3 SV
95. NTRK3 CN
96. PALB2 SV
97. PALB2 CN
98. PDGFRA SV
99. PDGFRA CN
100. PHOX2B SV
101. PHOX2B CN
102. PIK3CA SV
103. PIK3CA CN
104. PIK3R1 SV
105. PIK3R1 CN
106. PTCH1 SV
107. PTCH1 CN
108. PTEN SV
109. PTEN CN
110. RB1 SV
111. RB1 CN
112. RET SV
113. RET CN
114. RUNX1 SV
115. RUNX1 CN
116. SETD2 SV
117. SETD2 CN
118. SMAD4 SV
119. SMAD4 CN
120. SMARCA4 SV
121. SMARCA4 CN
122. SMARCB1 CN
123. SMO SV
124. SMO CN
125. SOCS1 SV
126. SOCS1 CN
127. STK11 SV
128. STK11 CN
129. SUFU SV
130. SUFU CN
131. TET2 SV
132. TET2 CN
133. TP53 SV
134. TP53 CN
135. TSC1 SV
136. TSC1 CN
137. TSC2 SV
138. TSC2 CN
139. VHL SV
140. VHL CN
141. WT1 SV
142. WT1 CN

A.0.4 Compound Descriptors

1. nAcid
2. ALogP
3. ALogp2
4. AMR
5. apol
6. naAromAtom
7. nAromBond
8. nAtom
9. nHeavyAtom
10. nH
11. nB
12. nC
13. nN
14. nO
15. nS
16. nP
17. nF
18. nCl
19. nBr
20. nI
21. ATSc1
22. ATSc2
23. ATSc3
24. ATSc4
25. ATSc5
26. ATSm1
27. ATSm2
28. ATSm3
29. ATSm4
30. ATSm5
31. ATSp1
32. ATSp2
33. ATSp3
34. ATSp4
35. ATSp5
36. nBase
37. BCUTw-1l
38. BCUTw-1h
39. BCUTc-1l
40. BCUTc-1h
41. BCUTp-1l
42. BCUTp-1h
43. nBonds
44. nBonds2
45. nBondsS
46. nBondsS2
47. nBondsS3
48. nBondsD
49. nBondsD2
50. nBondsT
51. bpol
52. C1SP1
53. C2SP1
54. C1SP2
55. C2SP2
56. C3SP2
57. C1SP3
58. C2SP3
59. C3SP3
60. C4SP3
61. SCH-3
62. SCH-4
63. SCH-5
64. SCH-6
65. SCH-7
66. VCH-3
67. VCH-4
68. VCH-5
69. VCH-6
70. VCH-7
71. SC-3
72. SC-4
73. SC-5
74. SC-6
75. VC-3

76. VC-4
77. VC-5
78. VC-6
79. SPC-4
80. SPC-5
81. SPC-6
82. VPC-4
83. VPC-5
84. VPC-6
85. SP-0
86. SP-1
87. SP-2
88. SP-3
89. SP-4
90. SP-5
91. SP-6
92. SP-7
93. VP-0
94. VP-1
95. VP-2
96. VP-3
97. VP-4
98. VP-5
99. VP-6
100. VP-7
101. ECCEN
102. nHBa
103. nwHBa
104. nsssB
105. ntsC
106. ndssC
107. naasC
108. naaaC
109. nssssC
110. ntN
111. ndsN
112. naaN
113. nsssN
114. naasN
115. ndO
116. nssO
117. naaO
118. nsF
119. ndsssP
120. ndS
121. nssS
122. naaS
123. nddssS
124. nsCl
125. nsBr
126. nsI
127. SdS
128. mindS
129. maxHBa
130. maxwHBa
131. maxdssC
132. maxaasC
133. maxaaaC
134. maxdsN
135. maxaaN
136. maxsssN
137. maxaasN
138. maxdO
139. maxssO
140. maxsF
141. maxsCl
142. hmax
143. hmin
144. ETA_Alpha
145. ETA_AlphaP
146. ETA_dAlpha_A
147. ETA_dAlpha_B
148. ETA_Epsilon_1
149. ETA_Epsilon_2
150. ETA_Epsilon_3
151. ETA_Epsilon_4
152. ETA_Epsilon_5
153. ETA_dEpsilon_A

154. ETA_dEpsilon_B
155. ETA_dEpsilon_C
156. ETA_Psi_1
157. ETA_dPsi_A
158. ETA_Shape_P
159. ETA_Shape_Y
160. ETA_Shape_X
161. ETA_Beta
162. ETA_BetaP
163. ETA_Beta_s
164. ETA_BetaP_s
165. ETA_Beta_ns
166. ETA_BetaP_ns
167. ETA_dBeta
168. ETA_dBetaP
169. ETA_Beta_ns_d
170. ETA_BetaP_ns_d
171. ETA_Eta
172. ETA_EtaP
173. ETA_Eta_R
174. ETA_Eta_F
175. ETA_EtaP_F
176. ETA_Eta_L
177. ETA_EtaP_L
178. ETA_Eta_R_L
179. ETA_Eta_F_L
180. ETA_EtaP_F_L
181. ETA_Eta_B
182. ETA_EtaP_B
183. ETA_Eta_B_RC
184. ETA_EtaP_B_RC
185. fragC
186. nHBAcc
187. nHBAcc2
188. nHBAcc3
189. nHBAcc_Lipinski
190. nHBDon
191. nHBDon_Lipinski
192. Kier1
193. Kier2
194. Kier3
195. nAtomLC
196. nAtomP
197. nAtomLAC
198. MLogP
199. McGowan_Volume
200. MDEC-11
201. MDEC-12
202. MDEC-13
203. MDEC-14
204. MDEC-22
205. MDEC-23
206. MDEC-24
207. MDEC-33
208. MDEC-34
209. MDEC-44
210. MDEO-11
211. MDEO-12
212. MDEO-22
213. MDEN-11
214. MDEN-12
215. MDEN-13
216. MDEN-22
217. MDEN-23
218. MDEN-33
219. MLFER_A
220. MLFER_BH
221. MLFER_BO
222. MLFER_S
223. MLFER_E
224. MLFER_L
225. PetitjeanNumber
226. nRing
227. n3Ring
228. n4Ring
229. n5Ring
230. n6Ring
231. n7Ring

232. n8Ring
233. n9Ring
234. n10Ring
235. nG12Ring
236. nFRing
237. nF6Ring
238. nF8Ring
239. nF9Ring
240. nF10Ring
241. nF11Ring
242. nF12Ring
243. nFG12Ring
244. nTRing
245. nT4Ring
246. nT5Ring
247. nT6Ring
248. nT7Ring
249. nT8Ring
250. nT9Ring
251. nT10Ring
252. nT11Ring
253. nT12Ring
254. nTG12Ring
255. nRotB
256. LipinskiFailures
257. TopoPSA
258. VABC
259. VAdjMat
260. MW
261. WTPT-1
262. WTPT-2
263. WTPT-3
264. WTPT-4
265. WTPT-5
266. WPATH
267. WPOL
268. XLogP
269. Zagreb

A.0.5 Compound Fingerprints

1. PubchemFP6
2. PubchemFP9
3. PubchemFP10
4. PubchemFP11
5. PubchemFP12
6. PubchemFP13
7. PubchemFP14
8. PubchemFP15
9. PubchemFP16
10. PubchemFP17
11. PubchemFP18
12. PubchemFP19
13. PubchemFP20
14. PubchemFP21
15. PubchemFP22
16. PubchemFP23
17. PubchemFP24
18. PubchemFP25
19. PubchemFP30
20. PubchemFP33
21. PubchemFP34
22. PubchemFP37
23. PubchemFP38
24. PubchemFP43
25. PubchemFP44
26. PubchemFP46
27. PubchemFP115
28. PubchemFP116
29. PubchemFP117
30. PubchemFP118
31. PubchemFP129
32. PubchemFP130
33. PubchemFP132
34. PubchemFP143
35. PubchemFP144
36. PubchemFP145

37. PubchemFP146
38. PubchemFP147
39. PubchemFP148
40. PubchemFP149
41. PubchemFP150
42. PubchemFP152
43. PubchemFP153
44. PubchemFP155
45. PubchemFP156
46. PubchemFP157
47. PubchemFP159
48. PubchemFP160
49. PubchemFP164
50. PubchemFP167
51. PubchemFP178
52. PubchemFP179
53. PubchemFP180
54. PubchemFP181
55. PubchemFP182
56. PubchemFP183
57. PubchemFP184
58. PubchemFP185
59. PubchemFP186
60. PubchemFP187
61. PubchemFP188
62. PubchemFP189
63. PubchemFP190
64. PubchemFP191
65. PubchemFP192
66. PubchemFP193
67. PubchemFP194
68. PubchemFP195
69. PubchemFP199
70. PubchemFP200
71. PubchemFP206
72. PubchemFP213
73. PubchemFP214
74. PubchemFP218
75. PubchemFP219
76. PubchemFP227
77. PubchemFP228
78. PubchemFP232
79. PubchemFP233
80. PubchemFP241
81. PubchemFP246
82. PubchemFP247
83. PubchemFP248
84. PubchemFP252
85. PubchemFP255
86. PubchemFP256
87. PubchemFP257
88. PubchemFP258
89. PubchemFP259
90. PubchemFP260
91. PubchemFP261
92. PubchemFP262
93. PubchemFP274
94. PubchemFP276
95. PubchemFP283
96. PubchemFP284
97. PubchemFP285
98. PubchemFP286
99. PubchemFP287
100. PubchemFP293
101. PubchemFP294
102. PubchemFP297
103. PubchemFP298
104. PubchemFP299
105. PubchemFP300
106. PubchemFP301
107. PubchemFP305
108. PubchemFP308
109. PubchemFP314
110. PubchemFP327
111. PubchemFP328
112. PubchemFP330
113. PubchemFP332
114. PubchemFP333

115. PubchemFP334
116. PubchemFP335
117. PubchemFP336
118. PubchemFP337
119. PubchemFP338
120. PubchemFP339
121. PubchemFP340
122. PubchemFP341
123. PubchemFP342
124. PubchemFP344
125. PubchemFP345
126. PubchemFP346
127. PubchemFP347
128. PubchemFP349
129. PubchemFP350
130. PubchemFP351
131. PubchemFP352
132. PubchemFP353
133. PubchemFP355
134. PubchemFP356
135. PubchemFP357
136. PubchemFP358
137. PubchemFP359
138. PubchemFP360
139. PubchemFP362
140. PubchemFP363
141. PubchemFP364
142. PubchemFP365
143. PubchemFP366
144. PubchemFP367
145. PubchemFP368
146. PubchemFP370
147. PubchemFP371
148. PubchemFP372
149. PubchemFP373
150. PubchemFP374
151. PubchemFP375
152. PubchemFP376
153. PubchemFP377
154. PubchemFP378
155. PubchemFP379
156. PubchemFP380
157. PubchemFP381
158. PubchemFP382
159. PubchemFP383
160. PubchemFP384
161. PubchemFP385
162. PubchemFP386
163. PubchemFP387
164. PubchemFP388
165. PubchemFP389
166. PubchemFP390
167. PubchemFP391
168. PubchemFP392
169. PubchemFP393
170. PubchemFP394
171. PubchemFP395
172. PubchemFP396
173. PubchemFP397
174. PubchemFP398
175. PubchemFP399
176. PubchemFP400
177. PubchemFP403
178. PubchemFP404
179. PubchemFP405
180. PubchemFP406
181. PubchemFP407
182. PubchemFP408
183. PubchemFP409
184. PubchemFP411
185. PubchemFP412
186. PubchemFP413
187. PubchemFP414
188. PubchemFP416
189. PubchemFP417
190. PubchemFP418
191. PubchemFP419
192. PubchemFP420

193. PubchemFP421
194. PubchemFP422
195. PubchemFP423
196. PubchemFP425
197. PubchemFP427
198. PubchemFP428
199. PubchemFP429
200. PubchemFP430
201. PubchemFP431
202. PubchemFP432
203. PubchemFP434
204. PubchemFP435
205. PubchemFP436
206. PubchemFP437
207. PubchemFP438
208. PubchemFP439
209. PubchemFP440
210. PubchemFP441
211. PubchemFP442
212. PubchemFP443
213. PubchemFP445
214. PubchemFP446
215. PubchemFP447
216. PubchemFP448
217. PubchemFP449
218. PubchemFP450
219. PubchemFP451
220. PubchemFP452
221. PubchemFP453
222. PubchemFP454
223. PubchemFP456
224. PubchemFP457
225. PubchemFP458
226. PubchemFP459
227. PubchemFP460
228. PubchemFP461
229. PubchemFP462
230. PubchemFP464
231. PubchemFP465
232. PubchemFP466
233. PubchemFP467
234. PubchemFP469
235. PubchemFP470
236. PubchemFP471
237. PubchemFP472
238. PubchemFP473
239. PubchemFP474
240. PubchemFP475
241. PubchemFP476
242. PubchemFP477
243. PubchemFP480
244. PubchemFP481
245. PubchemFP482
246. PubchemFP483
247. PubchemFP484
248. PubchemFP485
249. PubchemFP486
250. PubchemFP487
251. PubchemFP489
252. PubchemFP490
253. PubchemFP493
254. PubchemFP494
255. PubchemFP495
256. PubchemFP497
257. PubchemFP498
258. PubchemFP499
259. PubchemFP500
260. PubchemFP501
261. PubchemFP504
262. PubchemFP505
263. PubchemFP506
264. PubchemFP507
265. PubchemFP508
266. PubchemFP509
267. PubchemFP514
268. PubchemFP515
269. PubchemFP516
270. PubchemFP517

271. PubchemFP518
272. PubchemFP519
273. PubchemFP520
274. PubchemFP523
275. PubchemFP524
276. PubchemFP530
277. PubchemFP531
278. PubchemFP533
279. PubchemFP534
280. PubchemFP535
281. PubchemFP536
282. PubchemFP537
283. PubchemFP538
284. PubchemFP541
285. PubchemFP543
286. PubchemFP544
287. PubchemFP545
288. PubchemFP547
289. PubchemFP548
290. PubchemFP549
291. PubchemFP550
292. PubchemFP551
293. PubchemFP552
294. PubchemFP553
295. PubchemFP554
296. PubchemFP555
297. PubchemFP556
298. PubchemFP558
299. PubchemFP560
300. PubchemFP563
301. PubchemFP564
302. PubchemFP565
303. PubchemFP566
304. PubchemFP567
305. PubchemFP568
306. PubchemFP569
307. PubchemFP570
308. PubchemFP572
309. PubchemFP573
310. PubchemFP574
311. PubchemFP575
312. PubchemFP577
313. PubchemFP578
314. PubchemFP579
315. PubchemFP580
316. PubchemFP581
317. PubchemFP582
318. PubchemFP583
319. PubchemFP584
320. PubchemFP585
321. PubchemFP586
322. PubchemFP588
323. PubchemFP589
324. PubchemFP591
325. PubchemFP592
326. PubchemFP593
327. PubchemFP594
328. PubchemFP595
329. PubchemFP596
330. PubchemFP597
331. PubchemFP598
332. PubchemFP599
333. PubchemFP600
334. PubchemFP601
335. PubchemFP602
336. PubchemFP603
337. PubchemFP604
338. PubchemFP605
339. PubchemFP606
340. PubchemFP607
341. PubchemFP608
342. PubchemFP609
343. PubchemFP610
344. PubchemFP611
345. PubchemFP612
346. PubchemFP613
347. PubchemFP614
348. PubchemFP615

349. PubchemFP616
350. PubchemFP618
351. PubchemFP619
352. PubchemFP620
353. PubchemFP621
354. PubchemFP622
355. PubchemFP623
356. PubchemFP624
357. PubchemFP625
358. PubchemFP626
359. PubchemFP628
360. PubchemFP629
361. PubchemFP630
362. PubchemFP632
363. PubchemFP633
364. PubchemFP634
365. PubchemFP636
366. PubchemFP637
367. PubchemFP638
368. PubchemFP639
369. PubchemFP640
370. PubchemFP641
371. PubchemFP642
372. PubchemFP644
373. PubchemFP645
374. PubchemFP647
375. PubchemFP648
376. PubchemFP650
377. PubchemFP651
378. PubchemFP652
379. PubchemFP653
380. PubchemFP654
381. PubchemFP655
382. PubchemFP656
383. PubchemFP657
384. PubchemFP658
385. PubchemFP659
386. PubchemFP660
387. PubchemFP661
388. PubchemFP662
389. PubchemFP664
390. PubchemFP665
391. PubchemFP666
392. PubchemFP668
393. PubchemFP669
394. PubchemFP670
395. PubchemFP671
396. PubchemFP673
397. PubchemFP674
398. PubchemFP675
399. PubchemFP676
400. PubchemFP677
401. PubchemFP678
402. PubchemFP679
403. PubchemFP680
404. PubchemFP681
405. PubchemFP682
406. PubchemFP683
407. PubchemFP684
408. PubchemFP685
409. PubchemFP686
410. PubchemFP687
411. PubchemFP688
412. PubchemFP689
413. PubchemFP690
414. PubchemFP691
415. PubchemFP692
416. PubchemFP693
417. PubchemFP694
418. PubchemFP695
419. PubchemFP696
420. PubchemFP697
421. PubchemFP698
422. PubchemFP699
423. PubchemFP700
424. PubchemFP701
425. PubchemFP702
426. PubchemFP703

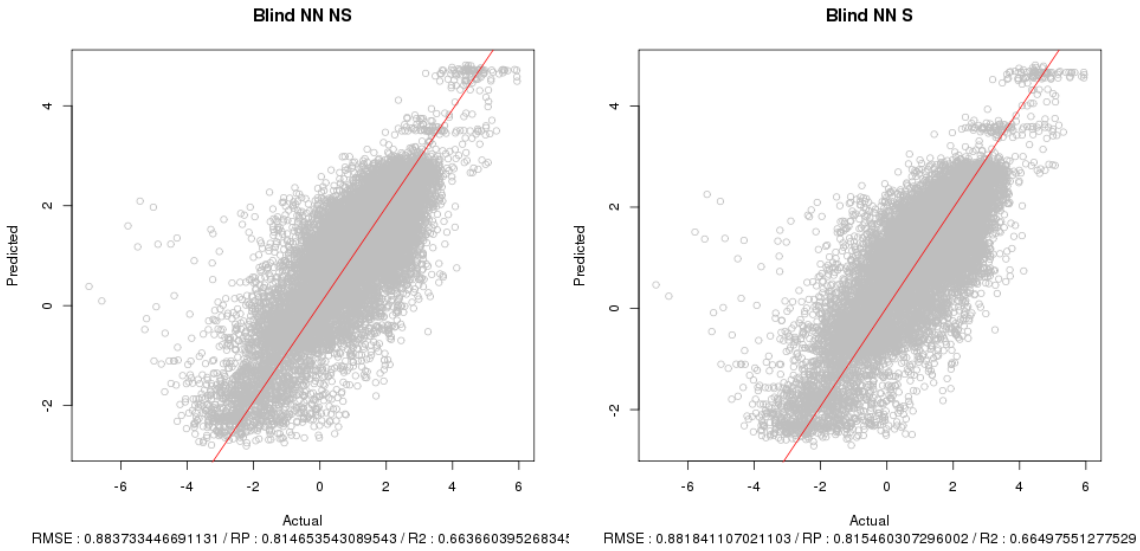
427. PubchemFP704
428. PubchemFP705
429. PubchemFP706
430. PubchemFP707
431. PubchemFP708
432. PubchemFP709
433. PubchemFP710
434. PubchemFP711
435. PubchemFP712
436. PubchemFP713
437. PubchemFP714
438. PubchemFP715
439. PubchemFP716
440. PubchemFP717
441. PubchemFP719
442. PubchemFP721
443. PubchemFP722
444. PubchemFP725
445. PubchemFP728
446. PubchemFP729
447. PubchemFP733
448. PubchemFP734
449. PubchemFP735
450. PubchemFP736
451. PubchemFP737
452. PubchemFP738
453. PubchemFP740
454. PubchemFP742
455. PubchemFP743
456. PubchemFP745
457. PubchemFP746
458. PubchemFP747
459. PubchemFP749
460. PubchemFP750
461. PubchemFP751
462. PubchemFP752
463. PubchemFP755
464. PubchemFP756
465. PubchemFP757
466. PubchemFP758
467. PubchemFP759
468. PubchemFP761
469. PubchemFP762
470. PubchemFP763
471. PubchemFP764
472. PubchemFP766
473. PubchemFP767
474. PubchemFP770
475. PubchemFP771
476. PubchemFP772
477. PubchemFP776
478. PubchemFP777
479. PubchemFP778
480. PubchemFP779
481. PubchemFP780
482. PubchemFP782
483. PubchemFP784
484. PubchemFP785
485. PubchemFP788
486. PubchemFP791
487. PubchemFP792
488. PubchemFP796
489. PubchemFP797
490. PubchemFP798
491. PubchemFP799
492. PubchemFP800
493. PubchemFP801
494. PubchemFP803
495. PubchemFP805
496. PubchemFP806
497. PubchemFP808
498. PubchemFP809
499. PubchemFP810
500. PubchemFP812
501. PubchemFP813
502. PubchemFP814
503. PubchemFP815
504. PubchemFP818

505. PubchemFP819
506. PubchemFP820
507. PubchemFP821
508. PubchemFP822
509. PubchemFP824
510. PubchemFP825
511. PubchemFP826
512. PubchemFP827
513. PubchemFP829
514. PubchemFP830
515. PubchemFP833
516. PubchemFP834
517. PubchemFP835
518. PubchemFP839
519. PubchemFP840
520. PubchemFP860
521. PubchemFP861

Appendix B

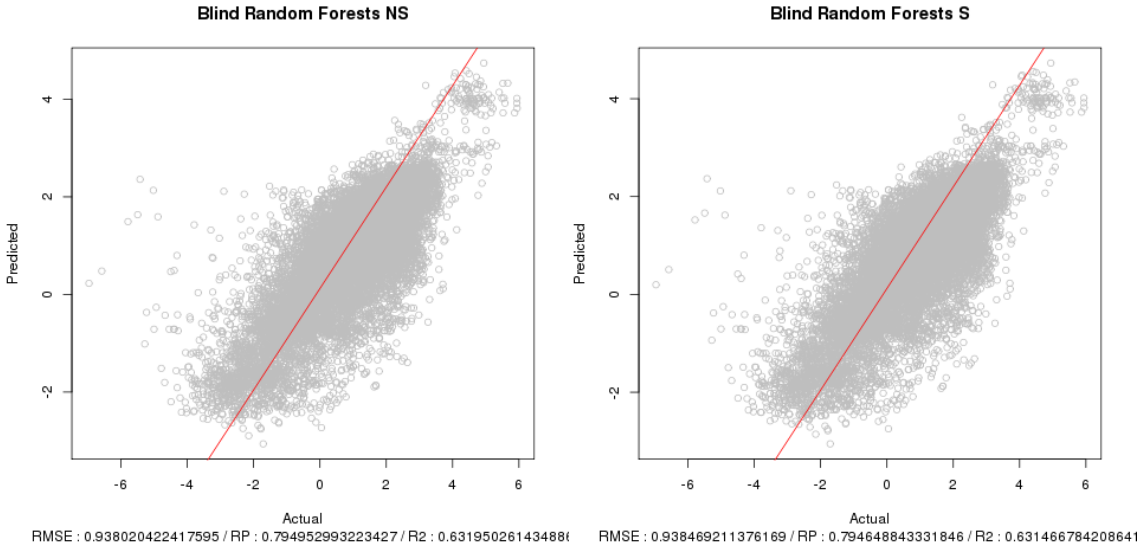
Results

B.1 Test Results



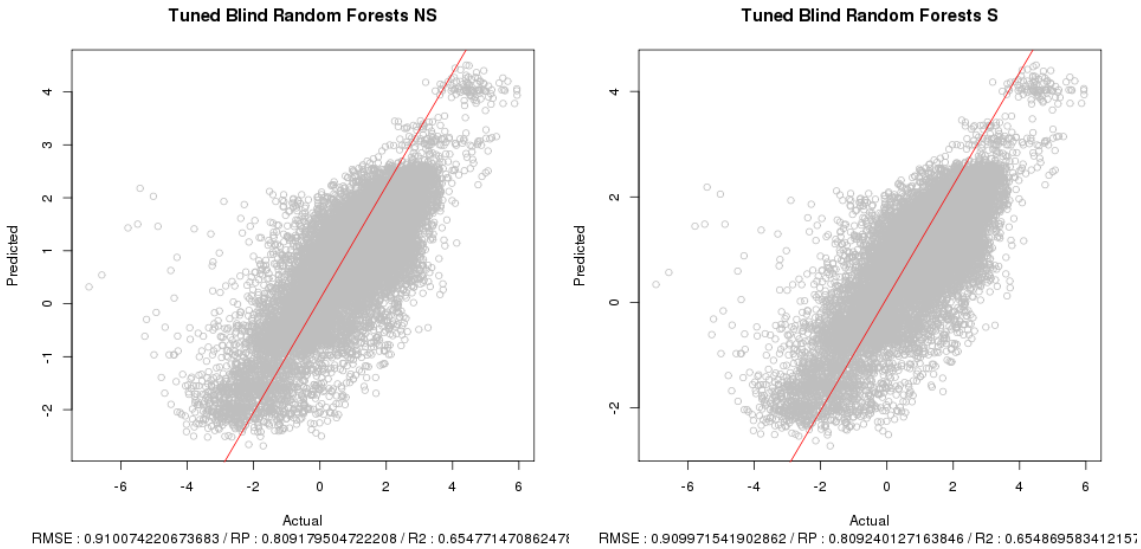
(a) Neural Networks Blind Test Non Stringent Results (b) Neural Networks Blind Test Stringent Results

Figure B.1: Neural Networks Blind Tests Results



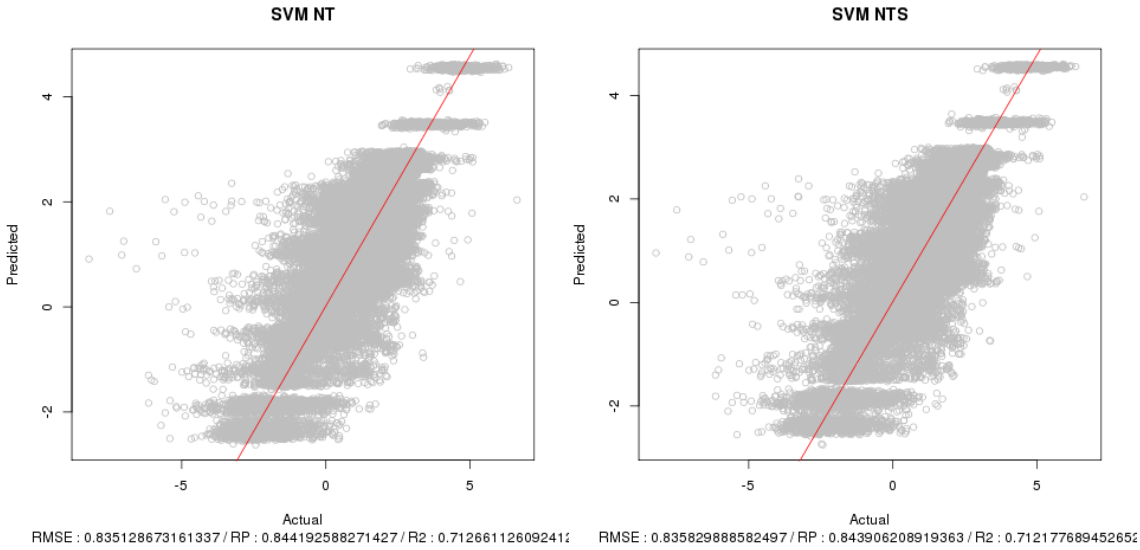
(a) Random Forest Blind Test Non Stringent Results (b) Random Forest Blind Test Stringent Results

Figure B.2: Random Forest Blind Test Results



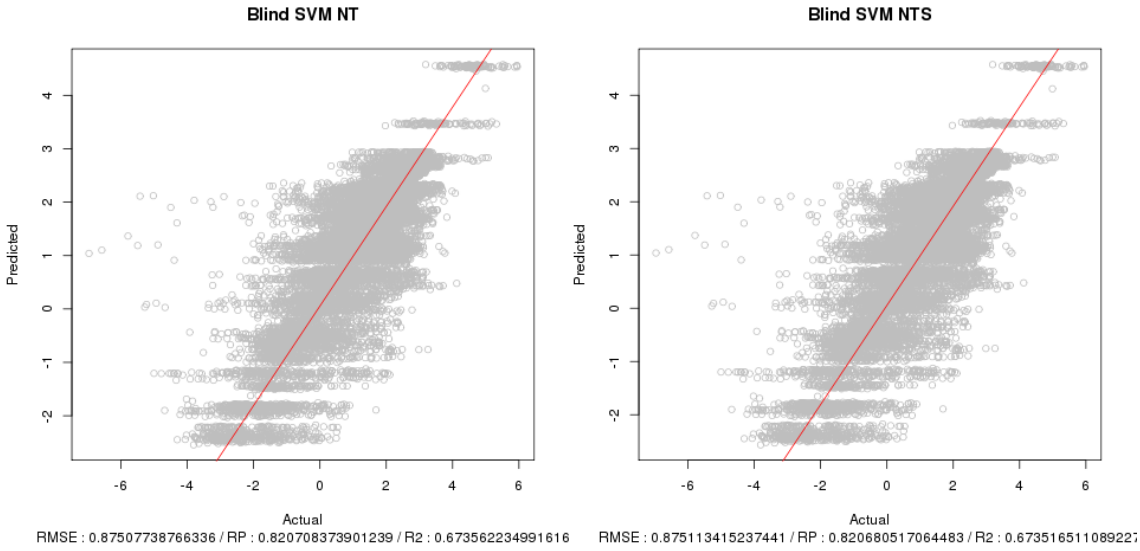
(a) Tuned Random Forest Blind Test Non Stringent Results (b) Tuned Random Forest Blind Test Stringent Results

Figure B.3: Tuned Random Forest Test Results



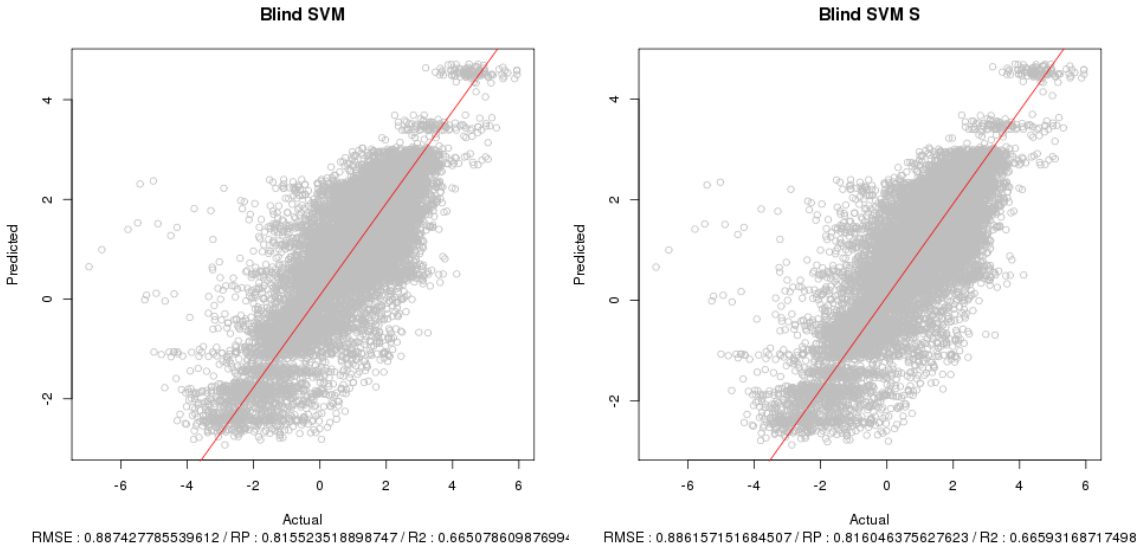
(a) SVM Non Stringent Results (b) SVM Stringent Results

Figure B.4: Support Vector Machine Test Results



(a) SVM Blind Test Non Stringent Results (b) SVM Blind Test Stringent Results

Figure B.5: Support Vector Machine Blind Test Results



(a) Tuned SVM Blind Test Non Stringent Results (b) Tuned SVM Blind Test Stringent Results

Figure B.6: Tuned SVM Test Results

B.2 Regression Results

B.2.1 Descriptors and Fragments Results

			Non Stringent	Stringent	Non Stringent Blind Test	Stringent Blind Test
NN	Results	RMSE	0.82 +/- 0.013	0.8 +/- 0.017	0.88	0.88
		RP	0.85 +/- 0.004	0.86 +/- 0.006	0.81	0.81
		R2	0.72 +/- 0.006	0.73 +/- 0.011	0.66	0.66
SVM	Results	RMSE	0.84 +/- 0.016	0.84 +/- 0.017	0.88	0.88
		RP	0.84 +/- 0.004	0.84 +/- 0.01	0.82	0.82
		R2	0.71 +/- 0.006	0.71 +/- 0.017	0.67	0.67
	Tuned	RMSE	0.82 +/- 0.016	0.82 +/- 0.018	0.89	0.89
		RP	0.85 +/- 0.004	0.85 +/- 0.01	0.82	0.82
		R2	0.72 +/- 0.006	0.72 +/- 0.016	0.67	0.67

Table B.1: Descriptors and Fragments Results; Support 20%; Minimum Vertexes 7

			Non Stringent	Stringent	Non Stringent Blind Test	Stringent Blind Test
NN	Results	RMSE	0.82 +/- 0.012	0.8 +/- 0.013	0.88	0.88
		RP	0.85 +/- 0.003	0.86 +/- 0.008	0.81	0.82
		R2	0.72 +/- 0.006	0.74 +/- 0.013	0.66	0.67
SVM	Results	RMSE	0.84 +/- 0.016	0.84 +/- 0.017	0.88	0.88
		RP	0.84 +/- 0.004	0.84 +/- 0.01	0.82	0.82
		R2	0.71 +/- 0.006	0.71 +/- 0.017	0.67	0.67
	Tuned	RMSE	0.82 +/- 0.016	0.82 +/- 0.018	0.89	0.89
		RP	0.85 +/- 0.004	0.85 +/- 0.01	0.82	0.82
		R2	0.72 +/- 0.006	0.72 +/- 0.016	0.67	0.67

Table B.2: Descriptors and Fragments Results; Support 30%; Minimum Vertexes 4

			Non Stringent	Stringent	Non Stringent Blind Test	Stringent Blind Test
NN	Results	RMSE	0.82 +/- 0.014	0.8 +/- 0.012	0.88	
		RP	0.85 +/- 0.004	0.86 +/- 0.007	0.81	
		R2	0.72 +/- 0.007	0.74 +/- 0.012	0.66	
SVM	Results	RMSE	0.84 +/- 0.016	0.84 +/- 0.017	0.88	0.88
		RP	0.84 +/- 0.004	0.84 +/- 0.01	0.82	0.82
		R2	0.71 +/- 0.006	0.71 +/- 0.017	0.67	0.67
	Tuned	RMSE	0.82 +/- 0.016	0.82 +/- 0.018	0.89	0.89
		RP	0.85 +/- 0.004	0.85 +/- 0.01	0.82	0.82
		R2	0.72 +/- 0.006	0.72 +/- 0.016	0.67	0.67

Table B.3: Descriptors and Fragments Results; Support 30%; Minimum Vertexes 5

			Non Stringent	Stringent	Non Stringent Blind Test	Stringent Blind Test
NN	Results	RMSE	0.83 +/- 0.016	0.79 +/- 0.014	0.88	0.88
		RP	0.85 +/- 0.004	0.86 +/- 0.008	0.82	0.82
		R2	0.72 +/- 0.007	0.74 +/- 0.013	0.66	0.66
SVM	Results	RMSE	0.84 +/- 0.016	0.84 +/- 0.017	0.88	0.88
		RP	0.84 +/- 0.004	0.84 +/- 0.01	0.82	0.82
		R2	0.71 +/- 0.006	0.71 +/- 0.017	0.67	0.67
	Tuned	RMSE	0.82 +/- 0.016	0.82 +/- 0.018	0.89	0.89
		RP	0.85 +/- 0.004	0.85 +/- 0.01	0.82	0.82
		R2	0.72 +/- 0.006	0.72 +/- 0.016	0.67	0.67

Table B.4: Descriptors and Fragments Results; Support 25%; Minimum Vertexes 5

			Non Stringent	Stringent	Non Stringent Blind Test	Stringent Blind Test
NN	Results	RMSE	0.82 +/- 0.015	0.8 +/- 0.01	0.88	0.88
		RP	0.85 +/- 0.005	0.86 +/- 0.008	0.82	0.81
		R2	0.72 +/- 0.008	0.73 +/- 0.013	0.66	0.66
SVM	Results	RMSE	0.84 +/- 0.016	0.84 +/- 0.017	0.88	0.88
		RP	0.84 +/- 0.004	0.84 +/- 0.01	0.82	0.82
		R2	0.71 +/- 0.006	0.71 +/- 0.017	0.67	0.67
	Tuned	RMSE	0.82 +/- 0.016	0.82 +/- 0.018	0.89	0.89
		RP	0.85 +/- 0.004	0.85 +/- 0.01	0.82	0.82
		R2	0.72 +/- 0.006	0.72 +/- 0.016	0.67	0.67

Table B.5: Descriptors and Fragments Results; Support 25%; Minimum Vertexes 6

			Non Stringent	Stringent	Non Stringent Blind Test	Stringent Blind Test
NN	Results	RMSE	0.82 +/- 0.017	0.79 +/- 0.013	0.88	0.88
		RP	0.85 +/- 0.005	0.86 +/- 0.007	0.82	0.82
		R2	0.72 +/- 0.008	0.74 +/- 0.013	0.66	0.67
SVM	Results	RMSE	0.84 +/- 0.016	0.84 +/- 0.017	0.88	0.88
		RP	0.84 +/- 0.004	0.84 +/- 0.01	0.82	0.82
		R2	0.71 +/- 0.006	0.71 +/- 0.017	0.67	0.67
	Tuned	RMSE	0.82 +/- 0.016	0.82 +/- 0.018	0.89	0.89
		RP	0.85 +/- 0.004	0.85 +/- 0.01	0.82	0.82
		R2	0.72 +/- 0.006	0.72 +/- 0.016	0.67	0.67

Table B.6: Descriptors and Fragments Results; Support 35%; Minimum Vertexes 3

			Non Stringent	Stringent	Non Stringent Blind Test	Stringent Blind Test
NN	Results	RMSE	0.91 +/- 0.153	0.79 +/- 0.013	0.89	0.88
		RP	0.82 +/- 0.052	0.86 +/- 0.008	0.81	0.81
		R2	0.67 +/- 0.081	0.74 +/- 0.013	0.66	0.66
SVM	Results	RMSE	0.84 +/- 0.016	0.84 +/- 0.017	0.88	0.88
		RP	0.84 +/- 0.004	0.84 +/- 0.01	0.82	0.82
		R2	0.71 +/- 0.006	0.71 +/- 0.017	0.67	0.67
	Tuned	RMSE	0.82 +/- 0.016	0.82 +/- 0.018	0.89	0.89
		RP	0.85 +/- 0.004	0.85 +/- 0.01	0.82	0.82
		R2	0.72 +/- 0.006	0.72 +/- 0.016	0.67	0.67

Table B.7: Descriptors and Fragments Results; Support 35%; Minimum Vertexes 4

B.2.2 Fragments Only Results

			Non Stringent	Stringent	Non Stringent Blind Test	Stringent Blind Test
NN	Results	RMSE	1.02 +/- 0.023	0.99 +/- 0.017	1.05	1.05
		RP	0.76 +/- 0.007	0.77 +/- 0.008	0.72	0.72
		R2	0.57 +/- 0.01	0.59 +/- 0.012	0.52	0.52
RF	Tuned	RMSE	1.02 +/- 0.023	1.0 +/- 0.019	1.09	1.09
		RP	0.75 +/- 0.006	0.76 +/- 0.009	0.7	0.7
		R2	0.57 +/- 0.009	0.58 +/- 0.014	0.5	0.5
SVM	Results	RMSE	1.1 +/- 0.022	1.11 +/- 0.02	1.14	1.14
		RP	0.71 +/- 0.008	0.7 +/- 0.014	0.67	0.67
		R2	0.5 +/- 0.011	0.49 +/- 0.019	0.44	0.44
	Tuned	RMSE	1.0 +/- 0.024	1.03 +/- 0.026	1.07	1.07
		RP	0.77 +/- 0.006	0.75 +/- 0.01	0.72	0.72
		R2	0.59 +/- 0.01	0.56 +/- 0.016	0.52	0.52

Table B.8: Fragments Only Results; Support 20%; Minimum Vertexes 7

			Non Stringent	Stringent	Non Stringent Blind Test	Stringent Blind Test
NN	Results	RMSE	1.11 +/- 0.031	1.09 +/- 0.019	1.16	1.16
		RP	0.7 +/- 0.01	0.71 +/- 0.008	0.65	0.65
		R2	0.49 +/- 0.014	0.51 +/- 0.011	0.42	0.42
RF	Results	RMSE	1.11 +/- 0.026	1.08 +/- 0.025	1.25	1.24
		RP	0.7 +/- 0.008	0.72 +/- 0.011	0.59	0.59
		R2	0.5 +/- 0.011	0.52 +/- 0.016	0.34	0.35
	Tuned	RMSE	1.07 +/- 0.025	1.05 +/- 0.021	1.18	1.18
		RP	0.73 +/- 0.007	0.73 +/- 0.01	0.63	0.64
		R2	0.53 +/- 0.011	0.54 +/- 0.015	0.4	0.41
SVM	Results	RMSE	1.09 +/- 0.024	1.1 +/- 0.023	1.12	1.12
		RP	0.71 +/- 0.007	0.71 +/- 0.011	0.68	0.68
		R2	0.51 +/- 0.01	0.5 +/- 0.016	0.46	0.46
	Tuned	RMSE	1.05 +/- 0.025	1.07 +/- 0.021	1.13	1.12
		RP	0.74 +/- 0.008	0.73 +/- 0.011	0.68	0.68
		R2	0.55 +/- 0.011	0.53 +/- 0.017	0.46	0.47

Table B.9: Fragments Only Results; Support 20%; Minimum Vertexes 8

			Non Stringent	Stringent	Non Stringent Blind Test	Stringent Blind Test
NN	Results	RMSE	0.97 +/- 0.015	0.94 +/- 0.016	0.97	0.97
		RP	0.78 +/- 0.005	0.79 +/- 0.008	0.77	0.77
		R2	0.61 +/- 0.008	0.63 +/- 0.013	0.59	0.59
RF	Tuned	RMSE	0.97 +/- 0.02	0.94 +/- 0.018	1	1
		RP	0.78 +/- 0.005	0.8 +/- 0.009	0.76	0.76
		R2	0.61 +/- 0.007	0.64 +/- 0.015	0.58	0.58
SVM	Results	RMSE	1.06 +/- 0.023	1.07 +/- 0.026	1.09	1.09
		RP	0.74 +/- 0.006	0.73 +/- 0.013	0.7	0.7
		R2	0.54 +/- 0.01	0.53 +/- 0.019	0.49	0.49
	Tuned	RMSE	0.93 +/- 0.019	0.96 +/- 0.023	0.98	0.98
		RP	0.8 +/- 0.004	0.79 +/- 0.01	0.77	0.77
		R2	0.64 +/- 0.006	0.62 +/- 0.015	0.59	0.59

Table B.10: Fragments Only Results; Support 30%; Minimum Vertexes 5

			Non Stringent	Stringent	Non Stringent Blind Test	Stringent Blind Test
NN	Results	RMSE	0.87 +/- 0.017	0.85 +/- 0.012	0.91	0.91
		RP	0.83 +/- 0.006	0.84 +/- 0.006	0.8	0.8
		R2	0.68 +/- 0.009	0.7 +/- 0.01	0.64	0.64
RF	Tuned	RMSE	0.87 +/- 0.016	0.86 +/- 0.015	0.94	0.94
		RP	0.83 +/- 0.004	0.83 +/- 0.009	0.79	0.79
		R2	0.68 +/- 0.006	0.69 +/- 0.015	0.63	0.63
SVM	Results	RMSE	0.93 +/- 0.018	0.94 +/- 0.022	0.96	0.96
		RP	0.8 +/- 0.004	0.8 +/- 0.01	0.78	0.78
		R2	0.64 +/- 0.007	0.63 +/- 0.015	0.6	0.6
	Tuned	RMSE	0.85 +/- 0.018		0.93	
		RP	0.84 +/- 0.004		0.8	
		R2	0.7 +/- 0.006		0.63	

Table B.11: Fragments Only Results; Support 25%; Minimum Vertexes 5

			Non Stringent	Stringent	Non Stringent Blind Test	Stringent Blind Test
NN	Results	RMSE	1.0 +/- 0.018	0.98 +/- 0.016	1.03	1.03
		RP	0.76 +/- 0.005	0.77 +/- 0.007	0.74	0.74
		R2	0.58 +/- 0.008	0.6 +/- 0.01	0.54	0.55
RF	Results	RMSE	1.07 +/- 0.023	1.01 +/- 0.019	1.11	1.11
		RP	0.73 +/- 0.005	0.76 +/- 0.01	0.69	0.69
		R2	0.53 +/- 0.007	0.58 +/- 0.015	0.48	0.48
	Tuned	RMSE	1.01 +/- 0.023	0.98 +/- 0.019	1.07	1.07
		RP	0.76 +/- 0.005	0.78 +/- 0.009	0.72	0.72
		R2	0.58 +/- 0.007	0.6 +/- 0.014	0.51	0.52
SVM	Results	RMSE	1.09 +/- 0.024	1.1 +/- 0.025	1.12	1.12
		RP	0.72 +/- 0.008	0.71 +/- 0.012	0.68	0.68
		R2	0.52 +/- 0.011	0.5 +/- 0.018	0.46	0.46
	Tuned	RMSE	0.98 +/- 0.021	1.01 +/- 0.025	1.04	1.03
		RP	0.78 +/- 0.004	0.76 +/- 0.01	0.74	0.74
		R2	0.6 +/- 0.007	0.58 +/- 0.016	0.54	0.54

Table B.12: Fragments Only Results; Support 25%; Minimum Vertexes 6

			Non Stringent	Stringent	Non Stringent Blind Test	Stringent Blind Test
NN	Results	RMSE	1.12 +/- 0.028	1.09 +/- 0.018	1.14	1.14
		RP	0.69 +/- 0.012	0.71 +/- 0.008	0.66	0.66
		R2	0.48 +/- 0.016	0.5 +/- 0.011	0.44	0.44
RF	Tuned	RMSE	1.08 +/- 0.025	1.05 +/- 0.022	1.14	1.14
		RP	0.72 +/- 0.007	0.74 +/- 0.009	0.67	0.67
		R2	0.52 +/- 0.01	0.54 +/- 0.014	0.45	0.45
SVM	Results	RMSE	1.22 +/- 0.028	1.23 +/- 0.028	1.26	1.26
		RP	0.62 +/- 0.01	0.62 +/- 0.015	0.58	0.58
		R2	0.39 +/- 0.012	0.38 +/- 0.018	0.33	0.33
	Tuned	RMSE	1.07 +/- 0.027	1.1 +/- 0.029	1.14	1.14
		RP	0.73 +/- 0.008	0.71 +/- 0.012	0.67	0.67
		R2	0.53 +/- 0.012	0.5 +/- 0.017	0.45	0.46

Table B.13: Fragments Only Results; Support 25%; Minimum Vertexes 7

			Non Stringent	Stringent	Non Stringent Blind Test	Stringent Blind Test
NN	Results	RMSE	0.84 +/- 0.012	0.83 +/- 0.014	0.9	0.9
		RP	0.84 +/- 0.004	0.85 +/- 0.009	0.81	0.81
		R2	0.7 +/- 0.007	0.72 +/- 0.016	0.65	0.65
RF	Tuned	RMSE	0.82 +/- 0.015	0.82 +/- 0.018	0.92	0.92
		RP	0.85 +/- 0.004	0.85 +/- 0.01	0.8	0.8
		R2	0.72 +/- 0.006	0.72 +/- 0.016	0.65	0.65
SVM	Results	RMSE	0.91 +/- 0.019	0.93 +/- 0.027	0.97	0.97
		RP	0.81 +/- 0.005	0.81 +/- 0.011	0.77	0.77
		R2	0.66 +/- 0.008	0.65 +/- 0.018	0.6	0.6
	Tuned	RMSE	0.8 +/- 0.017	0.84 +/- 0.024	0.91	0.9
		RP	0.86 +/- 0.004	0.84 +/- 0.01	0.81	0.81
		R2	0.73 +/- 0.006	0.71 +/- 0.017	0.65	0.65

Table B.14: Fragments Only Results; Support 35%; Minimum Vertexes 3

			Non Stringent	Stringent	Non Stringent Blind Test	Stringent Blind Test
NN	Results	RMSE	0.89 +/- 0.017	0.87 +/- 0.018	0.93	0.92
		RP	0.82 +/- 0.007	0.83 +/- 0.008	0.79	0.8
		R2	0.67 +/- 0.011	0.69 +/- 0.014	0.63	0.64
RF	Tuned	RMSE	0.87 +/- 0.016	0.86 +/- 0.016	0.94	0.94
		RP	0.83 +/- 0.004	0.83 +/- 0.01	0.79	0.79
		R2	0.68 +/- 0.007	0.69 +/- 0.016	0.63	0.63
SVM	Results	RMSE	0.99 +/- 0.02	1.0 +/- 0.028	1.04	1.03
		RP	0.78 +/- 0.006	0.77 +/- 0.012	0.74	0.74
		R2	0.6 +/- 0.009	0.59 +/- 0.019	0.55	0.55
	Tuned	RMSE	0.85 +/- 0.018	0.89 +/- 0.021	0.94	0.93
		RP	0.84 +/- 0.005	0.82 +/- 0.009	0.79	0.79
		R2	0.7 +/- 0.008	0.68 +/- 0.014	0.63	0.63

Table B.15: Fragments Only Results; Support 35%; Minimum Vertexes 4

B.3 Classification Results

B.3.1 Original Dataset

Neural Networks (Non Stringent)

		Reference	
		bad	good
Prediction	bad	12178	1341
	good	1383	12218

Table B.16: Original Neural Networks Non-Stringent Test Confusion Matrix

Accuracy	0.8996
Recall	0.8980
Specificity	0.9011
Precision	0.9008

Table B.17: Original Neural Networks Non-Stringent Test Results

Neural Networks (Stringent)

		Reference	
		bad	good
Prediction	bad	12219	1181
	good	1342	12378

Table B.18: Original Neural Networks Stringent Test Confusion Matrix

Accuracy	0.907
Recall	0.9010
Specificity	0.9129
Precision	0.9119

Table B.19: Original Neural Networks Stringent Test Results

Neural Networks Blind Test (Non Stringent)

		Reference	
		bad	good
Prediction	bad	4487	585
	good	627	4728

Table B.20: Original Neural Networks Non-Stringent Blind Test Confusion Matrix

Accuracy	0.8838
Recall	0.8774
Specificity	0.8899
Precision	0.8847

Table B.21: Original Neural Networks Non-Stringent Blind Test Results

Neural Networks Blind Test (Stringent)

		Reference	
		bad	good
Prediction	bad	4487	582
	good	627	4731

Table B.22: Original Neural Networks Stringent Blind Test Confusion Matrix

Accuracy	0.8841
Recall	0.8774
Specificity	0.8905
Precision	0.8852

Table B.23: Original Neural Networks Stringent Blind Test Results

Random Forest (Non Stringent)

		Reference	
		bad	good
Prediction	bad	12148	1252
	good	1413	12307

Table B.24: Original Random Forest Non-Stringent Test Confusion Matrix

Accuracy	0.9017
Recall	0.8958
Specificity	0.9077
Precision	0.9066

Table B.25: Original Random Forest Non-Stringent Test Results

Random Forest (Stringent)

		Reference	
		bad	good
Prediction	bad	12149	1264
	good	1412	12295

Table B.26: Original Random Forest Stringent Test Confusion Matrix

Accuracy	0.9013
Recall	0.8959
Specificity	0.9068
Precision	0.9058

Table B.27: Original Random Forest Stringent Test Results

Random Forest Blind Test (Non Stringent)

		Reference	
		bad	good
Prediction	bad	4485	582
	good	629	4731

Table B.28: Original Random Forest Non-Stringent Blind Test Confusion Matrix

Accuracy	0.8839
Recall	0.8770
Specificity	0.8905
Precision	0.8851

Table B.29: Original Random Forest Non-Stringent Blind Test Results

Random Forest Blind Test (Stringent)

		Reference	
		bad	good
Prediction	bad	4482	581
	good	632	4732

Table B.30: Original Random Forest Stringent Blind Test Confusion Matrix

Accuracy	0.8837
Recall	0.8764
Specificity	0.8906
Precision	0.8852

Table B.31: Original Random Forest Stringent Blind Test Results

Support Vector Machines (Non Stringent)

		Reference	
		bad	good
Prediction	bad	11909	1137
	good	1652	12422

Table B.32: Original Support Vector Machines Non-Stringent Test Confusion Matrix

Accuracy	0.8972
Recall	0.8782
Specificity	0.9161
Precision	0.9128

Table B.33: Original Support Vector Machines Non-Stringent Test Results

Support Vector Machines (Stringent)

		Reference	
		bad	good
Prediction	bad	11901	1126
	good	1660	12433

Table B.34: Original Support Vector Machines Stringent Test Confusion Matrix

Accuracy	0.8973
Recall	0.8776
Specificity	0.9170
Precision	0.9136

Table B.35: Original Support Vector Machines Stringent Test Results

Support Vector Machines Blind Test (Non Stringent)

		Reference	
		bad	good
Prediction	bad	4407	472
	good	707	4841

Table B.36: Original Support Vector Machines Non-Stringent Blind Test Confusion Matrix

Accuracy	0.8869
Recall	0.8618
Specificity	0.9112
Precision	0.9033

Table B.37: Original Support Vector Machines Non-Stringent Blind Test Results

Support Vector Machines Blind Test (Stringent)

		Reference	
		bad	good
Prediction	bad	4407	472
	good	707	4841

Table B.38: Original Support Vector Machines Stringent Blind Test Confusion Matrix

Accuracy	0.8869
Recall	0.8618
Specificity	0.9112
Precision	0.9033

Table B.39: Original Support Vector Machines Stringent Blind Test Results

B.3.2 Fragments Dataset

Neural Networks (Non Stringent)

		Reference	
		bad	good
Prediction	bad	11280	2198
	good	2281	11361

Table B.40: Fragments Neural Networks Non-Stringent Test Confusion Matrix

Accuracy	0.8348
Recall	0.8318
Specificity	0.8379
Precision	0.8369

Table B.41: Fragments Neural Networks Non-Stringent Test Results

Neural Networks (Stringent)

		Reference	
		bad	good
Prediction	bad	11460	1974
	good	2101	11585

Table B.42: Fragments Neural Networks Stringent Test Confusion Matrix

Accuracy	0.8497
Recall	0.8451
Specificity	0.8544
Precision	0.8531

Table B.43: Fragments Neural Networks Stringent Test Results

Neural Networks Blind Test (Non Stringent)

		Reference	
		bad	good
Prediction	bad	4148	1078
	good	966	4235

Table B.44: Fragments Neural Networks Non-Stringent Blind Test Confusion Matrix

Accuracy	0.804
Recall	0.8111
Specificity	0.7971
Precision	0.7937

Table B.45: Fragments Neural Networks Non-Stringent Blind Test Results

Neural Networks Blind Test (Stringent)

		Reference	
		bad	good
Prediction	bad	4165	1111
	good	949	4202

Table B.46: Fragments Neural Networks Stringent Blind Test Confusion Matrix

Accuracy	0.8024
Recall	0.8144
Specificity	0.7909
Precision	0.7894

Table B.47: Fragments Neural Networks Stringent Blind Test Results

Random Forest (Non Stringent)

		Reference	
		bad	good
Prediction	bad	11722	2054
	good	1839	11505

Table B.48: Fragments Random Forest Non-Stringent Test Confusion Matrix

Accuracy	0.8565
Recall	0.8644
Specificity	0.8485
Precision	0.8509

Table B.49: Fragments Random Forest Non-Stringent Test Results

Random Forest (Stringent)

		Reference	
		bad	good
Prediction	bad	11780	2075
	good	1781	11484

Table B.50: Fragments Random Forest Stringent Test Confusion Matrix

Accuracy	0.8578
Recall	0.8687
Specificity	0.8470
Precision	0.8502

Table B.51: Fragments Random Forest Stringent Test Results

Random Forest Blind Test (Non Stringent)

		Reference	
		bad	good
Prediction	bad	4146	1000
	good	968	4313

Table B.52: Fragments Random Forest Non-Stringent Blind Test Confusion Matrix

Accuracy	0.8113
Recall	0.8107
Specificity	0.8118
Precision	0.8057

Table B.53: Fragments Random Forest Non-Stringent Blind Test Results

Random Forest Blind Test (Stringent)

		Reference	
		bad	good
Prediction	bad	4164	1003
	good	950	4310

Table B.54: Fragments Random Forest Stringent Blind Test Confusion Matrix

Accuracy	0.8127
Recall	0.8142
Specificity	0.8112
Precision	0.8059

Table B.55: Fragments Random Forest Stringent Blind Test Results

Support Vector Machines (Non Stringent)

		Reference	
		bad	good
Prediction	bad	11455	2568
	good	2106	10991

Table B.56: Fragments Support Vector Machines Non-Stringent Test Confusion Matrix

Accuracy	0.8277
Recall	0.8447
Specificity	0.8106
Precision	0.8169

Table B.57: Fragments Support Vector Machines Non-Stringent Test Results

Support Vector Machines (Stringent)

		Reference	
		bad	good
Prediction	bad	11443	2544
	good	2118	11015

Table B.58: Fragments Support Vector Machines Stringent Test Confusion Matrix

Accuracy	0.8281
Recall	0.8438
Specificity	0.8124
Precision	0.8181

Table B.59: Fragments Support Vector Machines Stringent Test Results

Support Vector Machines Blind Test (Non Stringent)

		Reference	
		bad	good
Prediction	bad	4136	996
	good	978	4317

Table B.60: Fragments Support Vector Machines Non-Stringent Blind Test Confusion Matrix

Accuracy	0.8107
Recall	0.8088
Specificity	0.8125
Precision	0.8059

Table B.61: Fragments Support Vector Machines Non-Stringent Blind Test Results

Support Vector Machines Blind Test (Stringent)

		Reference	
		bad	good
Prediction	bad	4136	996
	good	978	4317

Table B.62: Fragments Support Vector Machines Stringent Blind Test Confusion Matrix

Accuracy	0.8107
Recall	0.8088
Specificity	0.8125
Precision	0.8059

Table B.63: Fragments Support Vector Machines Stringent Blind Test Results

B.3.3 Descriptors and Fragments Dataset

Neural Networks (Non Stringent)

		Reference	
		bad	good
Prediction	bad	12182	1309
	good	1379	12250

Table B.64: Descriptors and Fragments Neural Networks Non-Stringent Test Confusion Matrix

Accuracy	0.9009
Recall	0.8983
Specificity	0.9035
Precision	0.9030

Table B.65: Descriptors and Fragments Neural Networks Non-Stringent Test Results

Neural Networks (Stringent)

		Reference	
		bad	good
Prediction	bad	12304	1198
	good	1257	12361

Table B.66: Descriptors and Fragments Neural Networks Stringent Test Confusion Matrix

Accuracy	0.9095
Recall	0.9073
Specificity	0.9116
Precision	0.9113

Table B.67: Descriptors and Fragments Neural Networks Stringent Test Results

Neural Networks Blind Test (Non Stringent)

		Reference	
		bad	good
Prediction	bad	4475	595
	good	639	4718

Table B.68: Descriptors and Fragments Neural Networks Non-Stringent Blind Test Confusion Matrix

Accuracy	0.8817
Recall	0.8750
Specificity	0.8880
Precision	0.8826

Table B.69: Descriptors and Fragments Neural Networks Non-Stringent Blind Test Results

Neural Networks Blind Test (Stringent)

		Reference	
		bad	good
Prediction	bad	4497	596
	good	617	4717

Table B.70: Descriptors and Fragments Neural Networks Stringent Blind Test Confusion Matrix

Accuracy	0.8837
Recall	0.8794
Specificity	0.8878
Precision	0.8830

Table B.71: Descriptors and Fragments Neural Networks Stringent Blind Test Results

Random Forest (Non Stringent)

		Reference	
		bad	good
Prediction	bad	12108	1252
	good	1453	12307

Table B.72: Descriptors and Fragments Random Forest Non-Stringent Test Confusion Matrix

Accuracy	0.9003
Recall	0.8929
Specificity	0.9077
Precision	0.9063

Table B.73: Descriptors and Fragments Random Forest Non-Stringent Test Results

Random Forest (Stringent)

		Reference	
		bad	good
Prediction	bad	12135	1270
	good	1426	12289

Table B.74: Descriptors and Fragments Random Forest Stringent Test Confusion Matrix

Accuracy	0.9006
Recall	0.8948
Specificity	0.9063
Precision	0.9053

Table B.75: Descriptors and Fragments Random Forest Stringent Test Results

Random Forest Blind Test (Non Stringent)

		Reference	
		bad	good
Prediction	bad	4478	576
	good	636	4737

Table B.76: Descriptors and Fragments Random Forest Non-Stringent Blind Test Confusion Matrix

Accuracy	0.8838
Recall	0.8756
Specificity	0.8916
Precision	0.8860

Table B.77: Descriptors and Fragments Random Forest Non-Stringent Blind Test Results

Random Forest Blind Test (Stringent)

		Reference	
		bad	good
Prediction	bad	4486	578
	good	628	4735

Table B.78: Descriptors and Fragments Random Forest Stringent Blind Test Confusion Matrix

Accuracy	0.8843
Recall	0.8772
Specificity	0.8912
Precision	0.8859

Table B.79: Descriptors and Fragments Random Forest Stringent Blind Test Results

Support Vector Machines (Non Stringent)

		Reference	
		bad	good
Prediction	bad	11909	1137
	good	1652	12422

Table B.80: Descriptors and Fragments Support Vector Machines Non-Stringent Test Confusion Matrix

Accuracy	0.8972
Recall	0.8782
Specificity	0.9161
Precision	0.9128

Table B.81: Descriptors and Fragments Support Vector Machines Non-Stringent Test Results

Support Vector Machines (Stringent)

		Reference	
		bad	good
Prediction	bad	11901	1126
	good	1660	12433

Table B.82: Descriptors and Fragments Support Vector Machines Stringent Test Confusion Matrix

Accuracy	0.8973
Recall	0.8776
Specificity	0.9170
Precision	0.9136

Table B.83: Descriptors and Fragments Support Vector Machines Stringent Test Results

Support Vector Machines Blind Test (Non Stringent)

		Reference	
		bad	good
Prediction	bad	4407	472
	good	707	4841

Table B.84: Descriptors and Fragments Support Vector Machines Non-Stringent Blind Test Confusion Matrix

Accuracy	0.8869
Recall	0.8618
Specificity	0.9112
Precision	0.9033

Table B.85: Descriptors and Fragments Support Vector Machines Non-Stringent Blind Test Results

Support Vector Machines Blind Test (Stringent)

		Reference	
		bad	good
Prediction	bad	4407	472
	good	707	4841

Table B.86: Descriptors and Fragments Support Vector Machines Stringent Blind Test Confusion Matrix

Accuracy	0.8869
Recall	0.8618
Specificity	0.9112
Precision	0.9033

Table B.87: Descriptors and Fragments Support Vector Machines Stringent Blind Test Results