

*Learning a Predictive Model for Growth Inhibition from the NCI DTP Human Tumor Cell Line Screening Data: Does Gene Expression Make a Difference?*

Lothar Richter, Ulrich Ruckert, and Stefan Kramer

Pacific Symposium on Biocomputing 11:596-607(2006)

**LEARNING A PREDICTIVE MODEL FOR GROWTH  
INHIBITION FROM THE NCI DTP HUMAN TUMOR CELL  
LINE SCREENING DATA:  
DOES GENE EXPRESSION MAKE A DIFFERENCE?**

LOTHAR RICHTER, ULRICH RÜCKERT AND STEFAN KRAMER\*

*Institut für Informatik I12, Technische Universität München,  
Bolmannstr. 3, Garching b. München, Germany*

We address the problem of learning a predictive model for growth inhibition from the NCI DTP human tumor cell line screening data. Extending the classical Quantitative Structure Activity Relationship paradigm, we investigate whether including gene expression data leads to a statistically significant improvement of prediction quality. Our analysis shows that the straightforward approach of including individual gene expression as features does not necessarily improve, but on the contrary, may degrade performance significantly. When gene expression information is aggregated, for instance by features representing the correlation with reference cell lines, performance can be improved significantly. Further improvements may be expected if the learning task is structured by grouping features and instances.

## 1. Introduction

Pharmacogenomics is concerned with linking drug response with genomic as well as transcriptomic and proteomic data. Whereas many studies deal with genomic variation and single nucleotide polymorphisms (SNPs), we aim at extending the classical Quantitative Structure Activity Relationship (QSAR) paradigm by transcriptomic information. That is, we use not only the structural properties of the compounds for predicting pharmacological activity (as in classical QSAR), but also data about the biological environment, such as gene expression measurements of the involved cell lines. In this way we hope to improve the predictive accuracy of the induced models.

The study is based on the NCI DTP human tumor cell line screening database<sup>1</sup>. This database consists, by and large, of three parts: the first part contains measurements of the growth inhibition of human tumor cell

---

\*Corresponding author: [kramer@in.tum.de](mailto:kramer@in.tum.de)

lines caused by chemical compounds. The second part contains gene expression measurements for those cell lines. We do not use the (potentially useful) third part of the database containing data on so-called molecular targets. The NCI database has been the subject of a lot of interest in the past few years. So far, work has focused mainly on descriptive mining (variants of clustering, co-clustering/biclustering, and linking clustering results with other information)<sup>2,3,4,5,6,7,8</sup> and predictive mining for individual compounds or small, selected subsets of compounds<sup>9</sup>. In this paper, we develop a model that not only connects these different types of data, but also can be used to make predictions for new, yet unseen cases. We present the results of experiments in learning predictive models on all cell lines (their gene expression) and all compounds. The envisaged usage of the induced model is: given data on a new compound and a new cell line, predict the growth inhibition of that compound on the cell line. The goal of this study is not to achieve a particularly low error, but to test the null hypothesis that the inclusion of gene expression does not change the error of the predictive models.

This paper is organized as follows: In Section 2, we present the materials and methods. Section 3 describes the experimental set-up and data flow in detail. In Section 4, experimental results are presented both quantitatively and qualitatively, before Section 5 concludes and gives an outlook on future work.

## 2. Materials and Methods

### 2.1. Materials

The NCI DTP has tested tens of thousands of chemical compounds for growth inhibition on human tumor cell lines. Structure information about the compounds is available in a standard file format. For each cell line in the database, gene expression data from Affymetrix chips<sup>9</sup> and cDNA chips<sup>10</sup> is given. The overall dataset is constantly updated by the NCI and complemented by additional information.

For each chemical compound and tumor cell line, the database specifies an indicator of the growth inhibition of the compound on the cell line, the  $GI_{50}$  value. The  $GI_{50}$  is based on the concept of test through control ( $T/C$ , see figure 1(a)). Consider the growth of a tumor cell line until a certain point in time in an untreated control. The measured intensity changes from  $C_0$  to  $C$ . In contrast, the measured intensity in a treated cell line changes from  $T_0$  to only  $T$ . Given the measured intensities  $C_0$ ,  $C$ ,  $T$  and  $T_0$ , we can

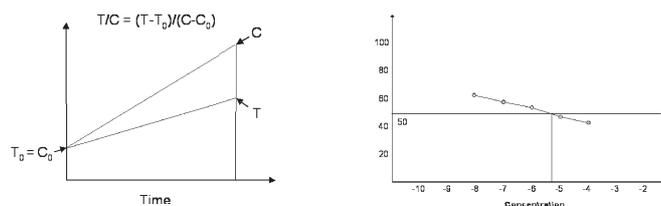


Figure 1. Left side (a): definition of  $T/C$  in general, right side (b): definition of  $GI_{50}$  from five measurements

calculate the growth (increase in measured intensity) relative to the control as  $T/C = (T - T_0)/(C - C_0)$ . If the  $T/C$  takes the value 0, then the growth of the tumor cell line has stopped completely. If it takes the value 1, the treatment did not make any difference compared to the untreated cell line. Thus, one (or in relative terms, 100%), implies low activity, and zero (or 0%) implies high activity<sup>a</sup>.

Now, in general, the  $GI_{50}$  value is defined as the concentration of a compound for which the growth is reduced to 50% compared to the untreated control, that is for which the  $T/C$  is 50%. Since it would be too costly and time-consuming to determine this point exactly, it is usually determined by *interpolation*. For the given dataset, the first step in this process is the measurement of the  $T/C$  value for five concentrations, from the lowest concentration  $10^{-8}$  M to the highest concentration  $10^{-4}$  M (see Fig. 1(b)). Then, we connect all pairs of neighboring points of measurement by straight lines. Next, we determine the point where one of these lines intersects the horizontal line at  $T/C = 50\%$ . The value on the x-axis (the concentration) of this point is then defined as the  $GI_{50}$ .

Unfortunately, there are many complications in this process. First, the procedure might not be successful, because the connected lines might be all below or above  $T/C = 50\%$ . In this case, the procedure usually *extrapolates* to a range outside the interval between  $10^{-8}$  M and  $10^{-4}$  M. If the intersection is close to the interval boundaries, then the value is taken as it is. Otherwise, the extrapolation is considered too unreliable, in which case the value is rounded to, e.g.,  $10^{-4}$  exactly. Therefore, the rounded values in the dataset are indicators of unreliable extrapolations. For the purpose of this paper, we dropped rounded values altogether, since we might otherwise

<sup>a</sup>The explanation is slightly simplified for ease of presentation.

not predict growth inhibition, but rather other, more coarse effects such as solubility. Since some compounds were tested with one cell line at various concentrations we selected the  $GI_{50}$  value most distant from the upper and closest to the lower concentration border.

## 2.2. Methods

The main tools for the analysis of the data are the graph mining system *FreeTreeMiner*<sup>12</sup> and the regression rule learning system *Cubist*<sup>11</sup>. Due to space constraints we can not give a detailed description of the two systems and refer to the references for details.

Most learning systems expect the input data in an attribute-value representation, i.e. a table. Thus, we need a preprocessing step to extract features from the compound's structure. In our study we use frequently occurring substructures to characterize each molecule. Each attribute represents a substructure; it is set to 1, if the molecule contains the substructure and 0 otherwise. Of course, it is way to expensive to use all possibly occurring substructures. Instead, we use a graph mining tool to identify those substructures that occur frequently in the database.

In its most general form, graph mining is concerned with finding frequently occurring subgraphs in a database of graphs. More formally, for a database  $D$  of graphs let  $f(s, D)$  denote the number of graphs containing the subgraph  $s$ . Then the goal of a graph mining system is to generate the set of all (connected) subgraphs that occur more often than a predefined threshold  $t$ :  $S = \{s | f(s, D) \geq t\}$ . As we represent a compound as a molecular graph, we can apply any general graph mining tool for the feature generation step. For our study we use *FreeTreeMiner*<sup>12</sup> to extract all acyclic subgraphs from the molecule database, whose relative frequency is greater than 3%<sup>b</sup>. The restriction on acyclic substructures is made mainly for efficiency purposes. However, unpublished experiments on standard SAR datasets shows no significant difference in performance between various graph mining approaches<sup>13</sup>. In the experiments we find that 5015 acyclic graphs occur in more than 3% of the database's compounds. Thus, we represent each compound by a list of Boolean values, where each value specifies whether the corresponding substructure occurs in the compound.

---

<sup>b</sup>Preliminary experiments indicate that a smaller threshold hardly makes a difference as we use only the most significant substructures (see below) and those tend to remain the same if one decreases the threshold.

Of course, many of the generated attributes might not contain any significant information about the  $GI_{50}$  value. Thus, we perform a second step to identify the meaningful attributes. We calculate a Wilcoxon rank sum statistic for each substructure with regard to the  $GI_{50}$  value and keep only those attributes that exceed a minimum significance level. By adjusting the significance threshold one can easily control the number of attributes in the dataset for performance tuning and overfitting avoidance. To calculate the statistic, we split the list of all compounds in two parts: those compounds which contain the substructure and those which do not.

We sort both lists according to the  $GI_{50}$  value for each cell line and compute the Wilcoxon rank sum statistic between the two lists. The resulting  $p$ -value is an indicator on how certain one can be that the compounds containing a particular substructure exhibit a different distribution of  $GI_{50}$  values than those compounds that do not contain the substructure. We calculate those  $p$ -values for each attribute and each cell line and use the minimum  $p$ -value over all cell lines as the overall  $p$ -value of the substructure. Thus, we assume a substructure is significant if it is significant for at least one cell line. We then sort the substructures according to these aggregate  $p$ -values to obtain the desired ranking of the generated features.

We chose the commercial tool Cubist for inducing a model that predicts the  $GI_{50}$  from the preprocessed data. Cubist generates regression rules with tests for attribute values in the body (antecedent) and linear models in the head (consequent). Table 2 contains typical rules generated by Cubist. Cubist implicitly selects features (for inclusion in the rule body or head), but is not designed to detect feature interactions. It is known to perform well in terms of predictive accuracy and to scale nicely to large datasets.

### 3. Experimental Set-Up and Data Flow

The goal of our experiments is to show that the inclusion of biological information makes a measurable difference in the predictive accuracy of the models. Note that it is hard to prove that including biological information is not useful at all, since we can only make statements about the particular approach we are following here. The results indicate that one can improve predictive accuracy if the problem representation is chosen carefully. The null hypothesis we want to put to the test in the following is that the inclusion of biological information does not improve performance at all.

We follow the hold-out procedure for evaluating the performance of the learned models: Two thirds of the examples are used for training the

	$g_1$	$g_2$	$g_3$	$g_4$	...	$g_m$
$l_1$					...	
$l_2$					...	
$l_3$					...	
...	...	...	...	...	...	...
$l_{60}$					...	

MolID	Structure
$C_1$	
$C_2$	
$C_3$	
$C_4$	
$C_5$	
...	...
$C_{37,000}$	

MolID	CellID	$-\log GI_{50}$
$c_1$	$l_1$	4.0
...	...	...
$C_1$	$l_{60}$	5.8
$C_2$	$l_1$	...
...	...	...
$C_2$	$l_{60}$	4.5
...	...	...
$C_{37,000}$	$l_1$	6.1
...	...	...
$C_{37,000}$	$l_{60}$	4.0

Figure 2. Input data for our approach. On the left, the table CellLines contains gene expression information for thousands of genes and around 60 cell lines. The Compounds table in the center gives 3D structure information for more than 37,000 chemical compounds. On the right, table  $GI_{50}$  contains the negative logarithm of the  $GI_{50}$  for combinations of compounds and cell lines.

model and one third is set aside to evaluate it. Since the dataset is very large, there is no need to perform the more time-consuming cross-validation procedure. Fig. 2 shows the input data for our approach. We are given a (target) table of  $-\log GI_{50}$  values for combinations of molecules and cell lines. Thus, we have to deal with a multi-relational learning problem. The naïve approach would be to join the three tables and then predict the  $GI_{50}$  from the chemical and biological information. However, joining the full three tables would result in a table of approximately 40 GB of data, which makes this approach impractical. Therefore, we perform feature selection to reduce the dimensionality of the gene expression and molecular structure table before joining the tables<sup>c</sup>. Since we still join the tables, we accept a high degree of redundancy in the data as gene expression and compound information is duplicated many times in memory. This, however, only affects memory consumption and not predictive accuracy.

As explained above, we choose those substructures whose Wilcoxon  $p$ -values on the  $GI_{50}$  values exceeds a certain threshold. Feature selection of the gene expression data is done in a class-blind manner. For the first batch of experiments, we simply sort the genes according to their variance over the cell lines, and add the expression values in this order as features to the data. Thus, inclusion of genes can be parameterized in exactly the same way as the inclusion of substructures. For instance, we might run experiments with the first 100 genes and the first 500 substructures.

<sup>c</sup>Feature selection is necessary anyway, given the high dimensionality and the fact that the performance of most machine learning algorithms tends to degrade with increasing numbers of features.

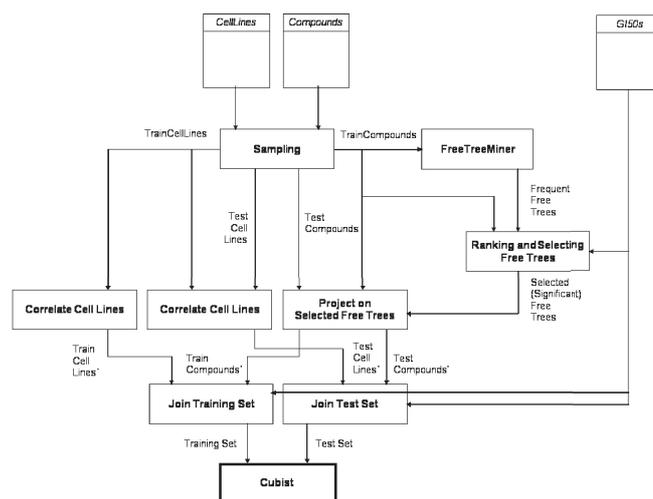


Figure 3. Data flow in the second batch of experiments (see also appendix A)

For the second batch of experiments, we do not use individual gene expression values as attributes. Instead, each attribute represents a reference cell line. For prediction, the new unseen cell line is compared to each reference cell line by calculating a correlation coefficient. This coefficient is then used as feature value. We conduct two experiments: In the first full-fledged variant, we use all 21 training cell lines as reference cell lines, and compute the Pearson correlation coefficient with these cell lines as features. In the second reduced variant, we use only 7 selected, representative cell lines from the training set as reference features. The reference cell lines are chosen to represent all 7 tissues: A549/ATCC for lung, SW-620 for colon, OVCAR-8 for ovary, SF-295 for central nervous system, MOLT-4 for blood, SK-MEL-28 for skin and UO-31 for kidney. The overall data flow for the second batch of experiments is shown in Fig. 3. A complete description of the data flow in both batches of experiments is given in appendix A.

#### 4. Experimental Results

Given the above experimental set-up, we first state the results quantitatively (see table 1) and then qualitatively, that is, we present parts of the best model found (see table 2).

We test the influence of biological information describing the respective

cell lines – represented in different forms – on prediction accuracy for various sets of substructures. The results are shown in table 1. As outlined in section 2.2 the sets of 100, 500 and 1000 substructures are selected according to their significance. As also outlined the biological information is provided in three different representations: Either as single genes, selected from the Scherf dataset<sup>10</sup> according to the variance over all cell lines (“single genes”), or as Pearson correlation coefficients with all training cell lines (“cell lines”) or, further condensed, as correlation with only one cell line per originating tissue (“tissue”). We also obtain reference results from substructure occurrence information without any biological information (“none”). We would now like to accept or reject the null hypothesis that the addition of biological data reduces the prediction error. To do so we perform a paired t-test on the test set example predictions. The table denotes the t-statistic and the corresponding  $p$ -value.

The results indicate that the null hypothesis can not be accepted or rejected regardless of the chosen set-up. Instead, the improvement or deterioration of predictive accuracy depends very much on dataset size and the representation of the biological information. First of all, even if one does not include any biological information, the performance depends on the number of substructure attributes: it is best for 500 substructures (error: 0.5701), slightly worse for 100 substructures (0.5744) and clearly worse for 1000 substructures (0.5897). This indicates that overfitting is an important issue for the set-up with 1000 substructures. Consequently, adding biological information in this case only increases the prediction error further to 0.5969 and 0.6054.

With 100 and 500 substructure features, adding biological information as correlation coefficients generally reduces the error. However, overfitting is an issue as well: using 7 representative cell lines performs better than using all 21 training cell lines in both cases, in the latter case even by a large margin (0.5694 vs. 0.5596). If one uses single genes as features, the performance depends very strongly on the number of features. There is an improvement for 500 genes (0.5637), but significant deterioration for 100 or 1000 gene attributes. Apparently, overfitting is again an issue. Overall, using selected reference cell lines to represent biological information seems to yield the best results.

A few of the model’s prediction rules produced by the learner are given in table 2. Rule 2 covers 2322 examples and states that for a substance where the substructures  $c(:c(:c(:c(:c))))(:c(:c(:c(:n(:c)))))$  and  $c(:c(:c(:c(-0))))(:c(-C(-C(-C(-C)))))(:c)$  (in SMARTS format, see

Table 1. Quantitative results from applying Cubist to the NCI data. We present results for varying numbers of compound features (100, 500, 1000): In each case the first row gives the reference result without the use of biological information. The remaining row show results for varying representations of the biological information. The third column states the mean absolute error. The fourth column gives the value of the t statistic from a paired t-test on the test instances, followed by the associated  $p$ -value. The sixth column states whether the outcome is a significant win or loss when compared to the reference result without biological information. The final column shows the runtime of Cubist on a Pentium IV CPU with 2.8 GHz.

# Sub-structures	Biological Information	Mean Abs. Error	t-Stat	$p$ -value	sign. Win/Loss	CPU Time
100	none	0.5744	-	-		79.1 s
100	cell lines	0.5741	-1.83	$p \approx 0.034$	+	162.1 s
100	tissue	0.5735	-7.32	$p < 0.001$	+	116.5 s
500	none	0.5701	-	-		597.5 s
500	single genes (100)	0.6319	78.49	$p < 0.001$	-	1048.2 s
500	single genes (500)	0.5637	-34.41	$p < 0.001$	+	4059.9 s
500	single genes (1000)	0.5826	33.83	$p < 0.001$	-	10040.0 s
500	cell lines	0.5694	-5.91	$p < 0.001$	+	760.4 s
500	tissue	0.5596	-51.13	$p < 0.001$	+	674.2 s
1000	none	0.5897	-	-		1186.2 s
1000	cell lines	0.5969	25.45	$p < 0.001$	-	1334.7 s
1000	tissue	0.6054	40.03	$p < 0.001$	-	1157.6 s

<http://www.daylight.com/dayhtml/doc/theory/theory.smarts.html> for a description) are present the  $GI_{50}$  value for a cell line is 4.6566 plus 0.29 times the correlation of this cell line with cell line SK-MEL-28 plus 0.17 times the correlation with cell line UO-31. Rule 9 works similarly for 7325 examples. The  $GI_{50}$  value for a substance/cell line combination can be predicted as 4.8483 plus 0.19 times the correlation with cell line SK-MEL-28 if the substance contains the substructures  $c(:c(:c(:c(:c(:c(:c)))))))-(:c(:c(:n(:c(:c(:c))))))$  while the substructures  $c(-C(-C))-(:c(-C(-C)(=O))(:c))$  and  $c(:c(:c(-O))(:c(:c(-C(-C))(:c))))$  are missing. Rules can also uncover more complex contributions from the biological features. For instance, in rule 23 (covering 4911 examples) in the presence of the two substructures,  $C(-C(-C)(-C))(-C(-C)(-O(-C)))$  and  $C(-C(-C(-C))(-O(-C(-C(-C))))$ , the  $GI_{50}$  prediction depends on the correlations with cell lines UO-31 (-0.43 times), SK-MEL-28 (0.38 times), OVCAR-8 (0.35 times), SW-620 (0.4 times) and MOLT4 (0.23 times), which makes use of five of the seven biological features present in the data.

The rules presented here clearly show that the use of appropriately represented biological information can enhance prediction power and complete structure-based information.

Table 2. Sample rules from Cubist. The rules are part of the model for 500 substructures and 7 selected tissues. On the left-hand side of the rules (the antecedent), we have tests for the occurrence or non-occurrence of substructures (in SMARTS format). On the right-hand side, we have linear models on variables representing the correlation with the reference cell lines.

<p>Rule 2: [2322 cases, mean 4.8189, range 3.456 to 10, est err 0.4961]</p> <p>if</p> <p><math>c(:c(:c(:c(:c))))(:c(:c(:c(:n(:c)))))) = 1</math></p> <p><math>c(:c(:c(:c(-0))))(:c(-C(-C(-C(-C))))(:c)) = 1</math></p> <p>then</p> <p><math>GI_{50} = 4.6566 + 0.29 \text{ SK-MEL-28} + 0.17 \text{ U0-31}</math></p>
<p>Rule 9: [7325 cases, mean 4.9445, range 2.903 to 10, est err 0.5976]</p> <p>if</p> <p><math>c(-C(-C))(:c(-C(-C)(=0))(:c)) = 0</math></p> <p><math>c(:c(:c(:c(:c(:c(:c(:c))))))(:c(:c(:n(:c(:c(:c)))))) = 1</math></p> <p><math>c(:c(:c(-0))))(:c(:c(-C(-C))(:c))) = 0</math></p> <p>then</p> <p><math>GI_{50} = 4.8483 + 0.19 \text{ SK-MEL-28}</math></p>
<p>Rule 23: [4911 cases, mean 6.2986, range -1.02 to 13, est err 1.3271]</p> <p>if</p> <p><math>C(-C(-C)(-C))(-C(-C)(-0(-C))) = 1</math></p> <p><math>C(-C(-C(-C)))(-0(-C(-C(-C)))) = 1</math></p> <p>then</p> <p><math>GI_{50} = 5.9707 - 0.43 \text{ U0-31} + 0.38 \text{ SK-MEL-28} + 0.35 \text{ OVCR-8} + 0.4 \text{ SW-620}</math>  <math>+ 0.23 \text{ MOLT-4}</math></p>

## 5. Conclusion and Outlook

In this paper, we investigated whether the inclusion of gene expression data can improve a predictive model for growth inhibition learned from the NCI DTP human tumor cell line screening data. Experiments showed that simply relying on individual gene expressions does not necessarily improve, but might degrade performance in predictive modeling of growth inhibition. To show statistically significant improvements over using chemical information alone, a suitable representation of the biological information has to be found.

In future work, we plan to take advantage of the rich structure in the various parts of the input data. For instance, there are classes of substances sharing the same mechanism of action or physico-chemical properties or groups of genes (functional modules) that belong together. Our approach did not yet address these interrelationships. The goal should be to find these groups automatically in the data: Learning mechanisms should be able to recognize and handle subgroups in sets of attributes as well as examples and ultimately use them for prediction.

## Appendix A: Data Flow in the Study

For the first batch of experiments (individual genes as features), the parameters to obtain varying numbers of features are *NrFreeTrees* and *NrGenes*:

- (1) Sampling
  - (a)  $(\sqrt{2} - 1) \cdot 100\%$  of the cell lines are randomly sampled (without replacement) for testing, the rest is used for training.
  - (b)  $(\sqrt{2} - 1) \cdot 100\%$  of the compounds are randomly sampled (without replacement) for testing, the rest is for training.
- (2) Find frequent substructures in training compounds: apply FreeTreeMiner to find frequently occurring substructures in the molecular graphs of the training compounds.
- (3) Rank and select significant frequent substructures
  - (a) Rank the frequent substructures according to the minimum  $p$ -value over all the training cell lines they were tested on. Note that not all compounds were tested on all cell lines.
  - (b) Select the first *NrFreeTrees* from the sorted list of substructures.
- (4) Reformulate compounds in terms of substructures
  - (a) Reformulate the training compounds in terms of the selected substructures. A substructural feature is set to one if the substructure is contained in the molecular graph, and set to zero otherwise.
  - (b) Analogously, reformulate the test compounds in terms of the selected substructures.
- (5) Rank and select genes
  - (a) Rank all genes according to their highest variance on the training cell lines.
  - (b) Select the first *NrGenes* from the list of genes.
- (6) Project on selected genes
  - (a) Project the training cell lines onto the selected genes. In other words, we keep only the *NrGenes* genes with the highest variance to describe the cell lines.
  - (b) Project the test cell lines onto the selected genes.
- (7) Join the training compounds, the training cell lines, and the  $GI_{50}$  values to obtain the training set. Note that not all compounds were tested on all cell lines. Therefore, the join involves a look-up in the  $GI_{50}$ s table for each pair of a compound and a cell line.
- (8) Join the test compounds, the test cell lines, and the  $GI_{50}$  values to obtain the test set. This is the same procedure as for the training set. In this way, the training set contains approximately two thirds of the instances, and the test set approximately one third. Also notice that in this way no compound or cell line in the test set is used for training.
- (9) Train Cubist on the training set and test it on the test set.

The data flow in the second batch of experiments (correlations with reference cell lines as features) differs slightly from the one before (see also Fig. 3), in steps 5 to 9. The only parameter is the number of substructures (*NrFreeTrees*):

- (5) Correlate training cell lines: compute a matrix with the Pearson correlation coefficient between all pairs of cell lines in the training cell lines. Thus, the correlation between a cell line and another is used as a feature to describe the former.
- (6) Correlate test cell lines with training cell lines: a matrix containing the Pearson correlation coefficient between each test cell line and each training cell line is computed. The correlation of a test cell line with a training cell line is used to describe the former.
- (7) Join the training compounds, the training cell line correlations, and the  $GI_{50}$  values to obtain the training set.
- (8) Join the test compounds, the test cell line correlations, and the  $GI_{50}$  values to obtain the test set.
- (9) Train Cubist on the training set and test it on the test set.

## References

1. M.R. Boyd, in : A.B. Teicher (ed.), *Cancer Drug Discovery and Development, Vol.2; Drug Development; Preclinical Screening, Clinical Trial and Approval*, Humana Press, 23-43, (1997).
2. P.E. Blower, C. Yang, M.A. Fligner, J.S. Verducci, L. Yu, S. Richman, J.N. Weinstein, *Pharmacogenomics J.* **2**(4), 259-271 (2002).
3. X. Fang, L. Shao, H. Zhang and S. Wang, *J. Chem. Inf. Comput. Sci.* **44**, 249-257 (2004).
4. A.A. Rabow, R.H. Shoemaker, E.A. Sausville and D.G. Covell, *J. Med. Chem.* **45**, 818-840 (2002).
5. L.M. Shi, Y. Fang, J.K. Lee, M. Waltham, D.T. Andrews, U. Scherf, K.D. Paul and J.N. Weinstein, *J. Chem. Inf. Comput. Sci.* **40**, 367-379 (2000).
6. A. Wallqvist, A.A. Rabow, R.S. Shoemaker, E.S. Sausville and D.G. Covell, *Bioinf.* **19**(17), 2212-2224 (2003).
7. J.N. Weinstein, Y. Pommier, *C.R.Biologies* **326**, 909-920 (2003).
8. D.T. Ross, U. Scherf, M.B. Eisen, C.M. Perou, C. Rees, P. Spellman, V. Iyer, S.S. Jeffrey, M. Van de Rijn, M. Waltham, A. Pergamenschikov, J.C.F. Lee, D. Lashkari, D. Schalon, T.G. Myers, J.N. Weinstein, D. Botstein, P.O. Brown, *Nature Genetics* **24**, 227-235 (2000).
9. J.E. Staunton, D.K. Slonim, H.A. Collier, P. Tamayo, M.J. Angelo, J. Park, U. Scherf, J.K. Lee, W.O. Reinhold, J.N. Weinstein, J.P. Mesirov, E.S. Lander, T.R. Golub, *PNAS* **98**(19), 10787-10792 (2001).
10. U. Scherf, D.T. Ross, M. Waltham, L.H. Smith, J.K. Lee, L. Tanabe, K.W. Kohn, W.C. Reinhold, T.G. Myers, D.T. Andrews, D.A. Scudiero, M.B. Eisen, E.A. Sausville, Y. Pommier, D. Botstein, P.O. Brown, J.N. Weinstein, *Nature Genetics* **24**, 236-244 (2000).
11. RuleQuest Research (2005). <http://www.rulequest.com/cubist-info.html>
12. U. Rückert, S. Kramer, *Proc. ACM Symp. on Appl. Comp. (SAC 2004)*, 564-570 (2004).
13. S. Sommer, *Lazy Structure-Activity Relationships*, Diploma Thesis, TU München (2005).