

Bruce, Craig L. (2010) Classification and interpretation in quantitative structure-activity relationships. PhD thesis, University of Nottingham.

**Access from the University of Nottingham repository:**

<http://eprints.nottingham.ac.uk/11666/1/thesis-final.pdf>

**Copyright and reuse:**

The Nottingham ePrints service makes this work by researchers of the University of Nottingham available open access under the following conditions.

This article is made available under the University of Nottingham End User licence and may be reused according to the conditions of the licence. For more details see:  
[http://eprints.nottingham.ac.uk/end\\_user\\_agreement.pdf](http://eprints.nottingham.ac.uk/end_user_agreement.pdf)

**A note on versions:**

The version presented here may differ from the published version or from the version of record. If you wish to cite this item you are advised to consult the publisher's version. Please see the repository url above for details on accessing the published version and note that access may require a subscription.

For more information, please contact [eprints@nottingham.ac.uk](mailto:eprints@nottingham.ac.uk)

CLASSIFICATION AND INTERPRETATION  
IN QUANTITATIVE STRUCTURE-ACTIVITY  
RELATIONSHIPS

Craig L. Bruce, MChem.

Thesis submitted to the University of Nottingham  
for the degree of Doctor of Philosophy

November 2010

# Abstract

A good QSAR model comprises several components. Predictive accuracy is paramount, but it is not the only important aspect. In addition, one should apply robust and appropriate statistical tests to the models to assess their significance or the significance of any apparent improvements. The real impact of a QSAR, however, perhaps lies in its chemical insight and interpretation, an aspect which is often overlooked.

This thesis covers three main topics: a comparison of contemporary classifiers, interpretability of random forests and usage of interpretable descriptors. The selection of data mining technique and descriptors entirely determine the available interpretation. Using interpretable approaches we have demonstrated their success on a variety of data sets.

By using robust multiple comparison statistics with eight data sets we demonstrate that a random forest has comparable predictive accuracies to the de facto standard, support vector machine. A random forest is inherently more interpretable than support vector machine, due to the underlying tree construction. We can extract some chemical insight from the random forest. However, with additional tools further insight would be available. A decision tree is easier to interpret than a random forest. Therefore, to obtain useful interpretation from a random forest we have employed a selection of tools. This includes alternative representations of the trees using SMILES and SMARTS. Using existing methods we can compare and cluster the trees in this representation. Descriptor analysis and importance can

be measured at the tree and forest level. Pathways in the trees can be compared and frequently occurring subgraphs identified. These tools have been built around the Weka machine learning workbench and are designed to allow further additions of new functionality.

The interpretability of a model is dependent on the model and the descriptors. They must describe something meaningful. To this end we have used the TMAcc descriptors in the Solubility Challenge and literature data sets. We report how our retrospective analysis confirms existing knowledge and how we identify novel C-domain inhibition of ACE.

In order to test our hypotheses we extended and developed existing software forming two applications. The Nottingham Cheminformatics Workbench (NCW) will generate TMAcc descriptors and allows the user to build and analyse models, including visualising the chemical interpretation. Forest Based Interpretation (FBI) provides various tools for interpreting a random forest model. Both applications are written in Java with full documentation and simple installations wizards are available for Windows, Linux and Mac.

# Publications

1. Spowage, B. M.; **Bruce, C. L.**; Hirst, J. D. Interpretable Correlation Descriptors for Quantitative Structure-Activity Relationships. *J. Cheminf.* **2009**, 1:22.
2. **Bruce, C. L.**; Melville, J. L.; Pickett, S. D.; Hirst, J. D. Contemporary QSAR Classifiers Compared. *J. Chem. Inf. Model.* **2007**, 47, 219-227.

# Acknowledgements

Foremost thanks go to my supervisor, Jonathan Hirst. His guidance and patience have been paramount in completing this work. I am grateful to GlaxoSmithKline for funding and the many useful discussions, particularly my industrial supervisor, Stephen Pickett, and his colleagues Chris Luscombe and Gavin Harper.

Thanks to both current and former members of the Hirst Group, especially James Melville and Benjamin Bulheller. I thank Clare-Louise Evans and Haydn Williams for providing a productive, yet relaxed, office environment.

I am grateful to the University of Nottingham for providing access to the High Performance Computing facility and the School of Chemistry for the Computational Chemistry facilities. In addition, thanks to the EPSRC for funding. I would like to acknowledge the academic licenses made available from both OpenEye and ChemAxon, which have allowed this work to progress much further without the need to reinvent the wheel, so to speak.

My current employer has encouraged and given me the opportunity to finish this work, even after I claimed it would be finished within a month. I thank Sandra McLaughlin and Andrew Grant for their extended patience given my huge underestimate of time required.

Finally, I am very grateful for the support from Michael McCusker and my family to see this work through to the end. It is to them I dedicate this completed thesis.

# Contents

Abstract . . . . .	i
Publications . . . . .	iii
Acknowledgements . . . . .	iv
List of Figures . . . . .	vii
List of Tables . . . . .	x
List of Abbreviations . . . . .	xii
<b>1 Introduction</b>	<b>1</b>
1.1 Industrial overview . . . . .	1
1.2 Cheminformatics . . . . .	3
1.2.1 Chemical data storage . . . . .	4
1.2.2 Substructure searching . . . . .	10
1.2.3 Similarity searching . . . . .	11
1.2.4 Clustering . . . . .	12
1.2.5 Docking . . . . .	13
1.3 Quantitative Structure-Activity Relationships . . . . .	14
1.4 Pragmatic programming . . . . .	19
1.5 High Performance Computing . . . . .	21
1.6 Summary . . . . .	23
<b>2 Methods</b>	<b>25</b>
2.1 Learning classifiers . . . . .	25
2.1.1 Decision tree . . . . .	26

2.1.2	Ensemble methods . . . . .	27
2.1.3	Bagging . . . . .	28
2.1.4	Boosting . . . . .	29
2.1.5	Stacking . . . . .	30
2.1.6	Random forest . . . . .	30
2.1.7	Support Vector Machine . . . . .	32
2.1.8	Partial Least Squares . . . . .	33
2.2	Model statistics . . . . .	34
2.2.1	Classification . . . . .	34
2.2.2	Regression . . . . .	36
2.2.3	Cross-validation . . . . .	38
2.3	Nonparametric Multiple-Comparison Statistical Tests . . . . .	39
2.4	Topological maximum cross correlation descriptors . . . . .	41
<b>3</b>	<b>Contemporary QSAR classifiers compared</b>	<b>49</b>
3.1	Abstract . . . . .	49
3.2	Introduction . . . . .	50
3.3	Methods . . . . .	52
3.4	Results and discussion . . . . .	56
3.5	Conclusions . . . . .	69
<b>4</b>	<b>Random forest: an interpretable classifier</b>	<b>72</b>
4.1	Introduction . . . . .	72
4.2	Methods . . . . .	73
4.3	Results and discussion . . . . .	79
4.3.1	Larger and skewed data . . . . .	79
4.3.2	Interpretation . . . . .	81
4.4	Conclusions . . . . .	88
<b>5</b>	<b>TMACC: interpretable descriptors</b>	<b>91</b>
5.1	Introduction . . . . .	91



5.2	Methods . . . . .	93
5.3	Results and discussion . . . . .	96
5.3.1	Angiotensin converting enzyme . . . . .	96
5.3.2	Solubility challenge . . . . .	103
5.4	Conclusions . . . . .	113
<b>6</b>	<b>Conclusions</b>	<b>115</b>
	<b>References</b>	<b>121</b>
<b>A</b>	<b>Appendix</b>	<b>143</b>
A.1	Solubility challenge structures . . . . .	143

# List of Figures

1.1	A sample graph depicting 12 nodes and 11 edges, representing propan-1-ol. . . . .	5
1.2	InChI and InChIKey for Lipitor. . . . .	8
1.3	Docking example. . . . .	14
1.4	A plot of the numbers of pairs of brooding storks and newborn babies in West Germany from 1960 to 1985. . . . .	17
2.1	An example decision tree modelling Lipinski-like rules. . . . .	26
2.2	Overview of bagging. . . . .	29
2.3	Overview of a random forest. . . . .	31
2.4	Separating hyperplane of the Support Vector Machine that maximizes the margin between two sets of perfectly separable objects. . . . .	33
2.5	Aspirin. TMAcc descriptors are based on topological distances. . . . .	44
2.6	A molecule from the ACE dataset in NCW, after the each atom has been assigned an activity contribution by colour . . . . .	47
3.1	Robustness with increasing number of trees (on 2.5D descriptors). . . . .	58
3.2	Robustness with increasing number of trees (on linear fragment descriptors). . . . .	63

3.3	Decision trees classifying activity of the ACE data set generated by (a) the J48 algorithm with pruning and (b) the random tree algorithm without pruning. Descriptors definitions are in Table 3.6 . . . . .	65
3.4	Percentage of model explained by unique descriptors for the DHFR data set. . . . .	67
4.1	Summary of descriptor frequency across a forest using the ACE data set . . . . .	83
4.2	Summary of descriptor frequency across tree 10 using the ACE data set . . . . .	85
4.3	Weka tree visualiser with added navigation panel to view trees in a forest . . . . .	86
4.4	A tree encoded as SMILES. Oxygen (O) is the root node. . .	87
4.5	A tree encoded as SMARTS. The wildcard atom type (A) is the root node. . . . .	87
4.6	Summary of the majority vote for a forest built using the ACE data set . . . . .	88
4.7	Molecule performance for a forest built using the ACE data set	89
5.1	ACE inhibitor features investigated. . . . .	97
5.2	TMACC interpretation of ACE inhibitors. A. Captopril. B. Enalaprilat. C. Lisinopril. . . . .	98
5.3	Conserved ACE residues that interact with lisinopril. A) tACE active site (green). <sup>172</sup> B) The N-domain active site of sACE (purple). <sup>173</sup> . . . . .	101
5.4	Comparison of the S1' sub-site residues which bind the lysyl group of lisinopril. A) tACE (green) <sup>172</sup> and B) the N-terminal domain of sACE (purple). <sup>173</sup> . . . . .	103

5.5	Predicted versus observed solubility for the training data ( $r^2 = 0.79$ ).	106
5.6	Predicted versus observed solubility for the cross-validated data.	107
5.7	Predicted versus observed solubility for the 32 molecule set.	107
5.8	Predicted versus observed solubility for the 28 molecule set.	108
5.9	Predicted versus observed solubility for the 24 molecule set.	108
5.10	TMACC colour scheme	108
5.11	<b>A</b> amiodarone, <b>B</b> diazoxide, <b>C</b> hydrochlorothiazide & <b>D</b> pyrimethamine.	110
5.12	<b>A</b> ciprofloxacin, <b>B</b> danofloxacin, <b>C</b> enrofloxacin & <b>D</b> sparfloxacin.	111

# List of Tables

1.1	Examples of SMILES . . . . .	6
1.2	Examples of SMARTS . . . . .	9
2.1	A general confusion matrix . . . . .	35
2.2	Colour codes used in TMAcc interpretation . . . . .	47
3.1	Summary of QSAR Data Sets . . . . .	56
3.2	Percentage of Correctly Classified Molecules for Different Classifiers on 2.5D Descriptor Data Sets . . . . .	57
3.3	Mean percentage of correctly classified molecules for different parameters of SVMs on 2.5D descriptor datasets. . . . .	60
3.4	Percentage of Correctly Classified Molecules for Different Classifiers on Linear Fragment Descriptor Data Sets . . . . .	61
3.5	Mean percentage of correctly classified molecules for different parameters of SVMs on linear fragment descriptor datasets. . . . .	61
3.6	Descriptor definitions from the decision trees in Figure 3.3 . . . . .	66
4.1	Composition of GSK data set and distribution of classes . . . . .	74
4.2	Default cost matrix. . . . .	76
4.3	Confusion matrix for the default random forest reporting the test set results. . . . .	79
4.4	Cost matrix for the cost sensitive random forest. . . . .	79
4.5	Confusion matrix for the cost sensitive random forest reporting the test set results. . . . .	79

4.6	Confusion matrix for the oversampled random forest reporting the test set results. . . . .	80
4.7	Cost matrix for the cost sensitive and oversampled random forest. . . . .	81
4.8	Confusion matrix for the cost sensitive and oversampled random forest reporting the test set results. . . . .	81
5.1	Frequency of activity of ACE inhibitor features as determined by the TMAcc interpretation. . . . .	100
5.2	Conserved ACE residues important for inhibitor interactions	102
5.3	Weka classifiers with multiple TMAcc variants. . . . .	104
5.4	Number of attributes in different variants of TMAcc. Number of $\log S$ components refers to those included. . . . .	104
5.5	Errors for the cross-validation for the three techniques. . . .	104
5.6	Final two techniques with cutoff of maximum topological distance . . . . .	105
5.7	GridSearch for $C$ , $\gamma$ and $\epsilon$ . . . . .	105
5.8	Test set compounds and their observed and predicted $\log S$ values. . . . .	109
5.9	Results of solubility challenge, ranks against the other 99 entrants. . . . .	112
5.10	Results of solubility challenge, against performance markers. The best and median results are also shown for comparison.	113
A.1	Solubility training set compounds . . . . .	143
A.2	Solubility test set compounds . . . . .	164

# List of Abbreviations

ACE	Angiotensin converting enzyme
AChE	Acetyl-cholinesterase
API	Application programming interface
BZR	Benzodiazepine receptor
COX2	Cyclooxygenase-2
CPU	Central processing unit
DHFR	Dihydrofolate reductase
FBI	Forest based interpretation
GPB	Glycogen phosphorylase b
GPU	Graphics processing unit
HPC	High performance computing
HQSAR	Hologram QSAR
InChI	International Chemical Identifier
LOO	Leave one out
MAE	Mean absolute error
MSE	Mean square error

NCW	Nottingham cheminformatics workbench
PCR	Principal component regression
PLS	Partial least squares
QSAR	Quantitative structure-activity relationship
QSPR	Quantitative structure-property relationship
RF	Random forest
RMSE	Root mean-squared error
SMARTS	Smiles arbitrary target specification
SMILES	Simplified molecular input line entry specification
SVM	Support vector machine
SVR	Support vector regression
THER	Thermolysin
THR	Thrombin
TMACC	Topological maximum cross correlation



# Chapter 1

## Introduction

### 1.1 Industrial overview

There has been an increasing push within the pharmaceutical industry to accelerate the output of new drugs, as many of the blockbusters approach the end of their patents. The so called patent cliff is going to impact all big pharma at the cost of billions of USD per annum. In addition, there has been a long term drive to decrease the length of drug discovery. The time required to bring a drug to market takes up a large proportion of the patent lifespan. Improvements to efficiency and throughput can be applied to every step of the drug discovery process. The drug discovery process is a long chain of research and development. A new drug will take anything from 12-15 years to reach the market. This thesis covers techniques normally used in lead generation and lead optimisation, both very early stage. Once a candidate compound has been validated using in silico methods, experimental data must support the hypothesis. Many filters and models exist to check for amongst other properties, bioavailability and toxicology. It is far better to drop a compound early rather than late. Each month in development accrues more expense which ultimately must be recouped by a successful drug, before itself making a profit. Candidate drugs enter animal and human trials at great cost. Each subsequent phase of testing

adds more rapidly increasing cost. In vitro studies are carried out first in test tubes before moving to in vivo studies in animals. The aim of these studies is to assess the response in living models. The dosage is also measured. Human trials occur in several phases. Phase I is a small cohort of healthy volunteers. The candidate drug and placebo are measured to determine the effectiveness and safety in humans. Further dosage studies are conducted based on the animal models as they are only a guide. There can be substantially different responses between both. Phase II continues to measure safety and efficacy. Patients are included for the first time in Phase II, along with further volunteers, comprising a larger test population. Phase III trials employ a larger group to assess a wider variety of patients. The phase III trials typically continue while regulatory approval is applied for. A candidate drug can be withdrawn at any stage if the trials highlight undesirable toxicity, side effects or if the drug simply does not work. Each country requires a separate license making global distribution complicated, lengthy and expensive. In addition different clinical studies may be required to comply with local law. Clinical studies continue after launch to ensure no unforeseen issues arise as a wider population administers the drug. In some cases side effects may take years to manifest and hence clinical trials simply cannot check for these. A recent high profile case of side effects was Merck's anti-inflammatory drug Vioxx. It led to an increased likelihood of heart attack and strokes. Merck voluntarily withdrew Vioxx after subsequent studies confirmed the link. During the clinical trials process R&D will find a suitable synthesis for mass production and formulation for the chosen delivery method. Only a handful of candidate drugs ever reach the market. Previously pharmaceuticals thrived on multiple billion dollar drugs. Most of these reach patent expiry by 2015, leaving a huge gap in revenues. Few new drugs have reached the same profitability. In recent years it is now harder to register a drug and more expensive. The funding bodies

and insurance companies demand lower prices for patented drugs and when cheaper generics are an option they are often taken. Pharma has reacted to the changing marketplace by reshaping R&D and expanding into new therapeutic areas, e.g. personalised medicines. Companies will have more drugs targeted at smaller patient populations, while less profitable, it removes the dependence on a few top earners. Biotechnology companies were hailed as the answer, many of which have now been bought by the pharma giants. Pharma themselves have continued to merge as well forming even bigger multinationals, which are now ready for restructuring and streamlining of costs. All efforts are to produce more successful drugs for a fraction of the cost.

Patents are harder to obtain and are more frequently being challenged. The cost to bring a successful drug to market now is 800 million to 1.3 billion dollars. Late stage attrition is a strong contributor to this high cost. Once a candidate enters clinical trials the investment costs jump immediately. Even once off patent there used to be little competition from the generics. Now it is substantial. In addition, the various agencies across the globe drive for the cheapest price. The cost and duration of development is becoming increasingly prohibitive. All aspects of development can and should be reviewed to improve them. The whole field of cheminformatics is essentially aimed at aiding the early stages of drug discovery. Unlike other disciplines in chemistry, cheminformatics is very closed aligned to the pharmaceutical and agrochemical industries.

## 1.2 Cheminformatics

While not the most well-known part of chemistry, it plays an important part in the delivery of *in silico* techniques. Cheminformatics was originally defined by Brown in 1998.<sup>1</sup> However, the subject has been in existence far longer. Markush structures were first used in patents from 1924 for describ-

ing multiple substituents. Wiswesser line notation, the first line notation to describe complex molecules was created in 1949.<sup>2</sup> The American Chemical Society created the Journal of Chemical Documentation in 1961 which has now morphed into the Journal of Chemical Information and Modeling. It is no longer the only journal dedicated to cheminformatics.

Pivotal to cheminformatics' development has been the growth and capabilities of the computer, core to any cheminformatics technique. Cheminformatics consists of several topics, which will be discussed briefly: chemical data storage, substructure searching, similarity searching, clustering, docking and QSAR to name a few. Most techniques are available as both 2D and 3D methods. 2D methods are primarily concerned with the topology of molecules. Conformers and stereochemistry are typically ignored unlike with 3D methods. 3D methods are typically more complex in order to model the extra data. Studies have found 2D methods can sometimes outperform 3D counterparts.<sup>3</sup> This may sound counterintuitive but simpler methodologies can yield better results with less computational effort.

### 1.2.1 Chemical data storage

The electronic storage of chemical data is crucial to computational techniques, yet even today there are many formats with advantages and disadvantages for all. While it is simple to store a chemical structure as an image file, this encodes limited chemical information in a challenging format. Machine readable files are necessary for tools to have access to the chemical information. A common storage method for chemical structures is using a molecular graph. A molecular graph uses graph theory from mathematics.<sup>4</sup> A graph is a representation of nodes and edges. In a molecule the nodes represent atoms and edges the bonds. The atoms and bonds in a molecule are not homogeneous. There will be several types of both, which must be captured. A limitation of graph theory is it only details the topology of a

structure, e.g. what nodes are connected to which edges. There is no spatial arrangement information. A sample graph is depicted in Figure 1.1.

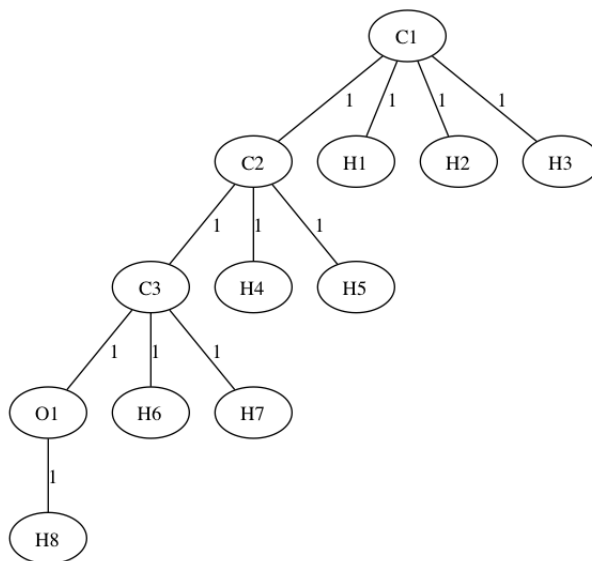


Figure 1.1: A sample graph depicting 12 nodes and 11 edges, representing propan-1-ol. The bond order is present on the edges.

Molecular graphs are the blueprints for the construction of SMILES (Simplified Molecular Input Line Entry Specification).<sup>5</sup> SMILES is a common and popular format, partly due to its concise and human readable nature. Each molecule is represented by a single line of text corresponding to the atoms and bonds in the molecular graph. Atoms are represented by their atomic symbol. Due to the high frequency of hydrogen and single bonds they are implicit. Double and triple bonds are encoded as an equals, =, and hash, #, symbol, respectively. Aromaticity is encoded by using lower case, *c*, for aromatic carbon and upper case, *C*, for aliphatic. Rings are denoted by numbering the opening and closing atoms of the ring with the same number. Multiple rings use sequentially increasing numbers to identify them. Branching chains are handled by encasing all branched atoms in parentheses. Table 1.1 depicts several sample SMILES.

A given molecule can have multiple, yet valid, SMILES strings. Starting at different atoms will result in a different path through the molecule. However, the molecule is identical. This can lead to duplicate SMILES in

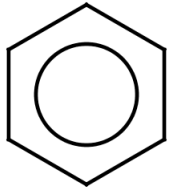
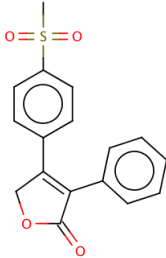
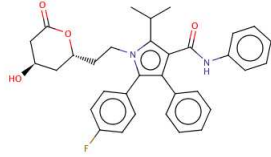
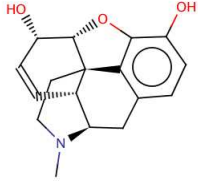
SMILES	Name	Depiction
<chem>c1ccccc1</chem>	Benzene	
<chem>CS(=O)(=O)c1ccc(cc1)C2=C(C(=O)OC2)c3ccccc3</chem>	Vioxx	
<chem>CC(C)c1c(C(=O)-Nc2ccccc2)c(c(c3ccc(F)-cc3)n1CC[C@@H]-4C[C@@H](O)CC(=O)-O4)c5ccccc5</chem>	Lipitor	
<chem>CN1CC[C@]23[C@H]4Oc5c3c-(C[C@@H]1[C@@H]-2C=C[C@@H]4O)ccc5O</chem>	Morphine	

Table 1.1: Examples of SMILES

a database, which is highly undesirable. To provide a unique representation the SMILES must be canonicalised. All variations of a single molecule should resolve to the same canonical SMILES.<sup>6</sup> There are now various algorithms for canonicalisation. Used consistently they provide unique SMILES.

Although the molecular graph does not contain any spatial data, SMILES can encode limited stereochemical information, such as chiral centres and cis-trans isomers. As SMILES do not contain spatial information they are not suitable storage for 3D methods. In docking the ligand and protein, 3D information is paramount to the technique. 3D information is stored using connection tables which contain Cartesian coordinates. Many formats use this method such as PDB (Protein Data Bank) and SDF (Structure-Data File). Typically the *xyz* coordinates of each atom are detailed along with each bond connections by atom ID.

Competing formats have arisen over time. One such example is the InChI (IUPAC International Chemical Identifier).<sup>7</sup> Designed by IUPAC with an aim to be freely available, computable by anyone and have a human readable quality, they encode more information than SMILES and address some of its weaknesses, e.g. the need for canonicalisation for uniqueness. The algorithm is a three step process: normalisation, canonicalisation and serialisation. Redundant information is removed, atoms are uniquely identified and the output written as a string. Additionally, an InChIKey (or hash) can be generated, which is not human readable and is a fixed 25 character string. It was introduced for practical reasons around web based searches as the full InChI can be overly verbose. The main InChI is composed of up to six layers of information comprising the core layer, charge, stereochemistry, isotopic, fixed hydrogens and reconnected layer. The core layer comprises three sublayers: chemical formula, connectivity and hydrogens. Only the chemical formula sublayer is required for a valid InChI. Each layer, and sublayer, is delimited by a slash, /. The InChIKey is based

on a hash algorithm. The first 14 characters determine the connectivity, while all additional information is encoded in the eight characters after a hyphen. The final two characters encode the InChI version and a checksum. The InChI and InChIKey for Lipitor is shown in Figure 1.2. While many vendors have added InChI support to their applications they have yet to receive widespread use over SMILES. SMILES have been the dominant 2D data format for many years. An OpenSMILES specification is being drawn up to address the original shortcomings.<sup>8</sup>

```
InChI=1S/C33H35FN2O5/c1-21(2)31-30(33(41)35-25-11-7-4-8-12-25)29(22-9-5-3-6-10-22)32(23-13-15-24(34)16-14-23)-36(31)18-17-26(37)19-27(38)20-28(39)40/h3-16,21,26-27,37-38H,17-20H2,1-2H3,(H,35,41)(H,39,40)/t26-,27-/m1/s1  
OUCSEDFVYPBLLF-KAYWLYCHSA-N
```

Figure 1.2: InChI and InChIKey for Lipitor.

Complementing SMILES are SMARTS, SMiles ARbitrary Target Specification,<sup>9</sup> a query language to search compound collections. Similar notation to SMILES is used, such as bond notation. Additional syntax is required to capture regular expression patterns. Carbon can be matched with its atomic number, [#6], aliphatic or aromatic carbon, [C,c] or the atom wildcard, \*. Connectivity can be specified using [CX4], where the carbon must have four bonds. Carboxylic acid is represented as [CX3](=O)[OX2H1], a carbon with three bonds, connected via a branched double bond to an oxygen and to another oxygen. The second oxygen has two bonds, one of which connects to a single hydrogen. Logical operators of and, ;, and or, ,, are available to form patterns such as a primary amide: [N;H3;+][C;X4], charge represented by + or -. SMARTS therefore represent a powerful yet flexible query language in which to encode chemical queries. SMARTS are routinely used in substructure searching, fragment based approaches, scaffold building and combinatorial chemistry. SMARTS are more complicated than SMILES, but encode more information, and hence are typically more verbose. Table 1.2 depicts various SMARTS examples.



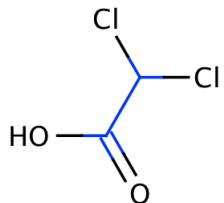
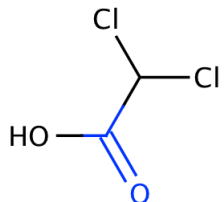
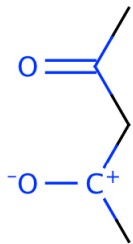
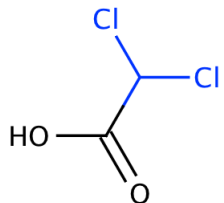
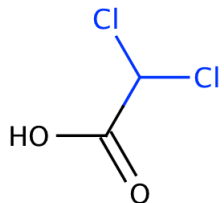
SMARTS	Name	Depiction
		
[CX4]	Alkyl carbon	
[CX3]=[OX1]	Carbonyl group	
[\$([CX3]=[OX1]),-\$([CX3+]-[OX1-])]	Carbonyl group, either resonance form	
[#6][F,Cl,Br,I]	Carbon attached to a halide	

Table 1.2: Examples of SMARTS

The plethora of chemical formats now available can pose a barrier, especially as commercial vendors typically introduce their own as well (e.g. `oeb` from OpenEye and `moe` from the Computing Computing Group). Various conversion tools exist. The best known is open Babel. However, some formats are less formalised than others leading to incorrect conversions.

### 1.2.2 Substructure searching

Often a database of compounds needs to be searched. Using either SMILES or SMARTS, depending how precise the query is, this can be quickly achieved. Many search methods already exist within graph theory. Substructure searching is essentially comparing two graphs to see if the query graph is represented in the other. This is known as subgraph isomorphism. Established subgraph searching methods perform poorly in large chemical databases, as they are exhaustive searches. A two step process is now used, in which the first step removes the majority of query compounds, leaving a minority for the exhaustive subgraph matching. A binary representation of the molecule is used, as binary calculations can be performed very efficiently. A chemical dictionary is used to represent the structural features present in a molecule. If a given feature is present a 1 is entered to the bitstring, otherwise a 0. In order for the molecule to pass onto subgraph matching the bitstrings must match. Therefore, as soon as differences appear in the bitstring the compound is discarded, as it will not match. Using the reduced database the subgraph isomorphism search is executed. It belongs to a class of problems known as *NP*-complete. *NP*-complete problems are characterised by an exponential relationship between the amount of time required and the size of the problem. This is because they are an exhaustive or brute-force approach. Therefore, it is prudent to avoid them when possible.

### 1.2.3 Similarity searching

Substructure searching is useful but often we do not have a complete query or want to find an alternative structure. Similarity searching is based on the similar property principle that states similar structure often leads to similar properties.<sup>10</sup> Therefore, the ability to find similar compounds is of interest. SMARTS could be constructed to become increasingly fuzzy, but to capture all possibilities is non-trivial. Sometimes only a small fragment of interest is known and one needs to find similar compounds. When dealing with 3D structure, similar compounds are more relevant than substructure matches, as the conformation becomes increasingly important. 2D similarity is computed by using the bitstrings used for substructure searching.

Molecular fingerprints are the bitstring representation of molecules using binary values.<sup>11</sup> Two flavours exist, the use of a fragment dictionary and hashed fingerprints. Fragment dictionaries use a predetermined list of structural features to represent each bit. Thus, you can map back to the structural element from the bitstring. This can make interpretation more accessible. Using hashed methods a predefined dictionary is not required, an advantage as any fragment present will be encoded. Using a dictionary you control what fragments are available and therefore can bias the fingerprint. Hashed fingerprints are unique per dataset and not readily interpretable in the same manner as a dictionary method. Fingerprints are now a popular basis for descriptors, even though their conception was never intended for this. 2D fingerprints were originally developed to accelerate substructure searching algorithm performance.<sup>11</sup> There is no intrinsic reason why they should perform well as descriptors. The good performance is likely because the molecule's properties and biological activity are dependent on the features encoded by the fingerprint.

A similarity coefficient is required to compare the bitstring representation of molecules. There are numerous similarity coefficients available. One

of the most common is the Tanimoto (or Jaccard) coefficient. It can be expressed as  $S_{AB}$ ,

$$S_{AB} = \frac{c}{a + b - c} \quad (1.1)$$

where  $a$  are the bits sets in that target structure,  $b$  the bits set in the database structure and  $c$  the bits set common in both structures. The result is a value between zero and one, where zero is no similarity and one an exact match. Results can then be sorted or restricted based on this measure. In addition to the other coefficients used in other disciplines similarity can also be calculated via compression.<sup>12</sup>

### 1.2.4 Clustering

Cluster analysis is useful for large data sets. Clustering aims to group similar compounds together. The similarity of the group members could be activity, therapeutic target or mode of action depending on the descriptors available. Typically clusters are made based on distance measures between other members of the data set. Most methods are also non-overlapping; each member belongs to only one cluster. Two types of methods are common: hierarchical and non-hierarchical. Hierarchical methods compare all members to each other. They create clusters of decreasing size, with each smaller cluster being a subset of the larger cluster. They are much like a decision tree in this respect, especially as a dendrogram is used for visualisation. Unlike a decision tree the clusters start as single compounds and grow as members are reorganised from the bottom up. Ward's method is based on distance from one member to all others with the aim of minimising variance, without a dendrogram.<sup>13</sup> The user often needs to pick the number of required clusters. This can be done manually retrospectively or cluster level selection methods now exist to determine a balance of cluster number and tightness of the clusters. The Jaccard statistic is used to compare

cluster groupings. Non-hierarchical clustering is normally done using the Jarvis-Patrick method.<sup>14</sup> The data set is only read once in this method. The first compound forms the first cluster. The second joins if a criterion is met otherwise it forms a new cluster. Hierarchical clustering allows multiple comparisons of all data points. A drawback of this method is the presence of singletons, clusters with one member. By altering the rules for joining or making a new cluster the number of singletons can be reduced.

### 1.2.5 Docking

Docking is typically used to model how a given set of ligands would interact with a protein. Many drugs have protein targets and understanding their interaction is key to designing a potent compound. This is a 3D experiment where the conformation of the ligand in the protein pocket, or binding site, is explored to find the most energetically favourable pose. Protein structures are not fixed. The presence of a ligand will by design alter the conformation of the protein. This may be key in allowing the ligand to bind. Accurate protein behaviour is important to modelling realistic binding. The solvent of the system should also be taken into account as gas-phase simulations are not representative of a biological system. Ideally an experiment will start with a known crystal structure to which a theoretical ligand will be bound. A sample docking of the actual structure and best pose is shown in Figure 1.3. Initial methods assumed rigid-body structures, which is not ideal. More modern techniques allow flexible docking of both protein and ligand giving a more accurate representation. Flexible docking is far more computationally expensive, especially when a large set of ligands is used. Docking is a challenging technique as there are many hurdles. The crystal structure of the protein is the interpretation of the original crystallographer. Protein structures devised from homology modelling are even more subjective. If the protein structure is incorrect expecting reasonable binding energies is

unrealistic. The binding energy represents the non-covalent interactions between the ligand and protein. A force field approach would use van der Waals and electrostatic interactions between all atoms of both molecules to predict the binding energy.

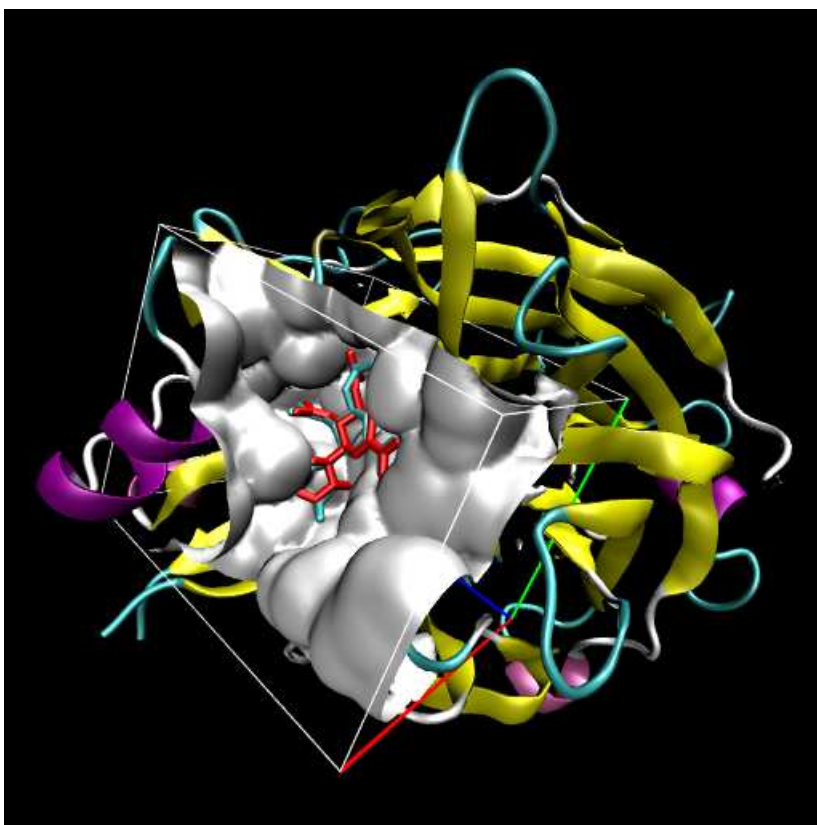


Figure 1.3: Docking example. Only the atoms in the box are used for calculation purposes. The red molecule is the crystal structure and blue the best docked pose.

### 1.3 Quantitative Structure-Activity Relationships

QSAR (Quantitative Structure-Activity Relationship) is the focus of this thesis. It was first used in the seminal work by Hansch<sup>15,16</sup>. Hansch devised an equation relating descriptors of electronic properties and hydrophobicity to biological activity.

$$\log\left(\frac{1}{C}\right) = k_1 \log P + k_2 \sigma + k_3 \quad (1.2)$$

where  $C$  is the concentration of compound needed to produce a standard response in a given time.  $\log P$  is the octanol-water partition coefficient and  $\sigma$  is the Hammett substitution parameter. Hansch also proposed that activity was parabolically dependent on  $\log P$ :

$$\log\left(\frac{1}{C}\right) = -k_1 (\log P)^2 + k_2 (\log P) + k_3 \sigma + k_4 \quad (1.3)$$

The reasoning for parabolic dependence on  $\log P$  was the compound hydrophobicity should not be so low as not to cross the cell membrane, or so high that once in the membrane it remains in situ. Electronic parameters are important in determining activity. The Hammett parameters come from Equations 1.4 and 1.5. The equations quantify related compound reaction rates and positions of equilibrium.

$$\log\left(\frac{k}{k_0}\right) = \rho\sigma \quad (1.4)$$

$$\log\left(\frac{K}{K_0}\right) = \rho\sigma \quad (1.5)$$

where  $k$  is the rate and  $K$  is the equilibrium constant for a particular substituent relative to a reference compound (typically hydrogen, corresponds to  $k_0$  and  $K_0$ ). Hammett used the hydrolysis of benzoate esters to measure reaction rates and ionisation constants of substituted benzoic acids for equilibria. The parameter  $\sigma$  is determined by the nature of the substituent and whether it is meta or para to a group on the aromatic ring. The reaction constant  $\rho$  is fixed for a particular process. Since Hammett's original work<sup>17</sup> there have been various advances.<sup>18</sup>

Modern QSAR is more ambitious in the number and variety of descrip-

tors considered. There are three components to a QSAR model: the data, how one represents the data and the statistical technique chosen to find a relationship between them. All three affect the overall model produced. All models should be thoroughly validated to ensure they are predictive. While the classic QSAR dataset is small (about 30 compounds) it is now widely acknowledged that this can be too small.<sup>19</sup> One cannot expect to find a relationship from so few datapoints. 60 compounds has been suggested as the minimum size for a dataset.<sup>20</sup>

Tens of thousands of descriptors can be readily generated in silico. On first inspection one may think more descriptors means an improved model. However, in reality the noise and cross-correlation of the descriptors can confuse the learning algorithm. Better performance can be achieved with a smaller number of descriptors. Indeed techniques exist to perform attribute selection before building the primary model as increased descriptors can impact the model generation significantly. Some machine learning techniques inherently perform this step.

Popular algorithms for QSAR are decision trees, neural networks, genetic algorithms, support vector machine (SVM), partial least squares (PLS) and ensemble methods, e.g. random forests. Further details of these algorithms is detailed in Chapter 2. The literature is full of examples of all these algorithms on various targets and various comparisons. Although no modification to the algorithm is required, many algorithms have had some modification, for example making neural networks more interpretable<sup>21,22</sup>, constructing chemical kernels<sup>23,24</sup> for SVM and multiple variations on PLS<sup>25,26</sup>.

Recently, several articles have questioned the usefulness of QSAR, for example, when excellent models in terms of  $q^2$  (defined in Chapter 2) can be found between number of brooding storks and newborn babies (Figure 1.4).<sup>27</sup> How can we expect QSAR to find meaningful chemical relationships when, it seems, anything can correlate? The key is not changing the tools,



but ensuring they are used correctly. Larger data sets and interpretable descriptors, are just two areas for improvement. The use of multiple statistics to measure the model, not just  $q^2$  is important, as a single statistic can be misleading. Rigorous statistics should be used to compare techniques. The community must push best practices to enable further advances in the field.

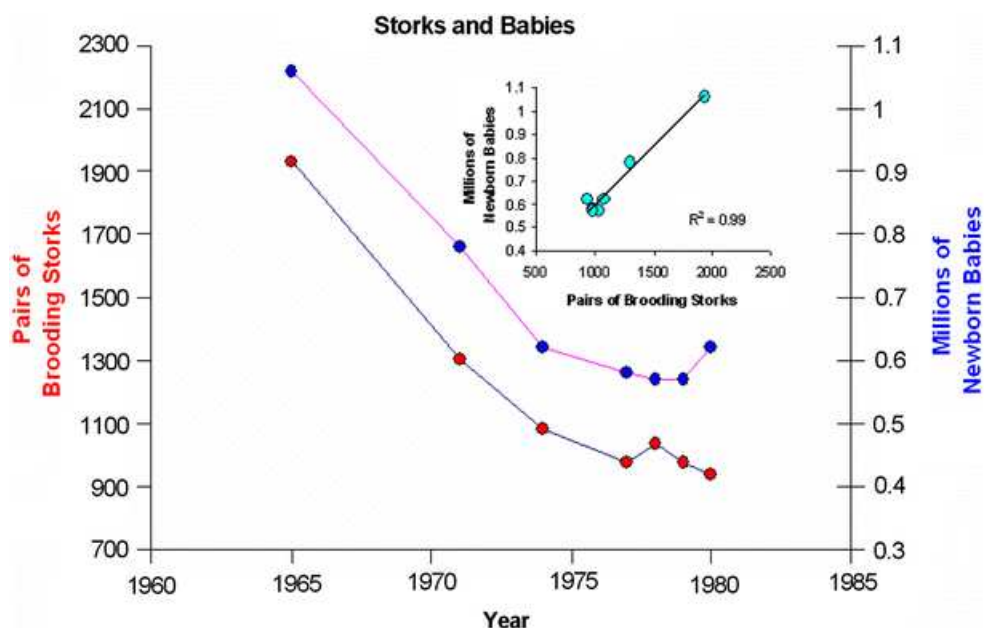


Figure 1.4: A plot of the numbers of pairs of brooding storks and newborn babies in West Germany from 1960 to 1985. Representation of the data as a correlation plot (Inset). With permission from Springer Science+Business Media: *Journal of Computer-Aided Molecular Design*, QSAR: dead or alive?, 22, 2008, 82, Arthur M. Doweyko, Figure 1, ©Springer Science+Business Media.

QSAR is analogous to QSPR (Quantitative structure-property prediction). We show a QSPR example later when we predict solubility as part of the solubility challenge organised by the Journal of Chemical Information and Modeling. Solubility is an important property of a drug, but hard to estimate both experimentally and computationally. It is the ability of a substance to dissolve in a solvent. Drugs need to be water soluble in order to be orally bioavailable, which is the preferred method of administration. Drugs which are not water soluble cannot be tested in biological assays, have poor pharmacological profiles and tend to precipitate in storage.<sup>28</sup> Solubility is one of the contributing factors to the high attrition rates in drug discovery,

which must be reduced. Current computational models for solubility can have an error of an order of magnitude. This is compounded by a lack of reliable and reproducible experiment data.

Drug discovery is a multi-variate problem. For example, while one can create a model for activity, it is useless if the compounds do not have a suitable solubility. Even once these are overcome, other obstacles will likely challenge the path to successful registration. Many factors contribute to solubility, making prediction challenging. These include lipophilicity, number of hydrogen bonds formed in solvent, the ability to form intramolecular hydrogen bonds, the ionisation states of functional groups and the properties in crystal form.<sup>29</sup>

The field, in general, has seen numerous advances over the last few decades especially in QSAR, mainly from 2D all the way to 6D. Admittedly 2D and 3D are the most commonly used. 2D QSAR uses descriptors based on the 2D topology of the molecule. This can include 3D values such as volume or surface area. However, these values are typically calculated without multiple conformers and possibly without 3D coordinates if SMILES are the input data format. 2D QSAR uses machine learning algorithms to find a relationship between the descriptors and activity. 3D QSAR has two popular flavours, Comparative Molecular Similarity Indices Analysis<sup>30</sup> (CoMSIA) and Comparative Molecular Field Analysis (CoMFA).<sup>31</sup> CoMFA attempts to find a correlation between activity and 3D shape, electrostatics and hydrogen bonding. The biologically active conformation for each molecule is required. Each conformation has molecular fields generated. The fields are calculated with the molecule in a lattice, thus allowing comparison to all other molecules. Typically electrostatic and steric probes are used at each defined point within the lattice. PLS is used to analyse the data generated from the lattice. The coefficients obtained from PLS allow 3D contour plots to be generated on the lattice. The contours indicate regions where charged

groups or steric bulk can affect activity (both in a positive and negative fashion). CoMSIA was developed after CoMFA and addresses some of its drawbacks.

The fourth “dimension” in the paradigm is sampling and includes the sampling of conformation, alignment, pharmacophore sites and entropy.<sup>32</sup> The composite information coming from each of these sampled property sets is embedded in the resulting QSAR model. Most of the other *n*D-QSAR methods consider each of these properties, which are sampled as individual dimensions in their QSAR studies. Hence one would get a 5D QSAR<sup>33</sup> method if conformational sampling is included, and a 6D QSAR<sup>34</sup> approach if both conformational and alignment samplings are considered.

## 1.4 Pragmatic programming

Computational tools have advanced both commercially and from open source projects. A range of programs are now available to assist from various commercial suppliers. One of the most useful features is access to an application programming interface (API), as invariably one always wants to do something slightly outside the scope of a program. Weka<sup>20</sup> is a machine learning workbench; Marvin<sup>35</sup> allows for the sketching and visualisation of molecules. Both programs are written in Java enabling them to work cross-platform. NCW and FBI, the two packages which result from this thesis were possible through the availability of the API for Weka and Marvin. Without the API a huge amount of additional coding would of been required. In addition, these tools have already been validated by the community. NCW and FBI are discussed in Chapters 5 and 4 in more detail.

Weka is an open source application that allows us to modify the source if necessary or extend the API. Marvin is not open source but the API is mature and was capable of accomplishing our tasks. The modern day computational chemist or cheminformatician requires a strong grounding in

computer science to benefit from all the tools available. Indeed, cheminformatics has seen an explosion of software packages and web services. This is only set to rise as Research Councils encourage open source publication of work they have funded. Some programs have gone on to form commercial spin outs from universities. Some programs funded over the long term have become very successful, Weka is an example, first introduced in 1993. From the cheminformatics world KNIME<sup>36</sup> is becoming increasingly popular as an open source alternative to Pipeline Pilot.<sup>37</sup> It optionally runs Weka and uses commercial plug-ins from Schrödinger<sup>38</sup> who partly fund its ongoing development.

Producing programs as a result of a research project can be problematic, as once complete they often are left unmaintained. Many such projects were seen during this research. This was one reason for the introduction of NCW to encapsulate the in-house code built up over the years into a single maintainable package. The pragmatic programming approach also aids this.<sup>39</sup> There are three principles to follow: version control, testing and continuous integration. First, a collection of source files is no good to anyone; documentation, compilation instructions and any dependencies are required. Version control is a repository of source code (or any file) which keeps a record of all the changes to a project. The repository's ability to compare and revert to older revisions is very useful, as well as the simple fact that all the required files are in one place. Popular version control programs are subversion<sup>40,41</sup> and git<sup>42</sup>; both follow a different methodology. Second, how can one know what code is supposed to do? Documentation is normally thin, rarely extensive. Unit tests provide the programmer with some confidence in the code, but third parties can view unit tests as sample code usage. Unit tests enable code verification and validation. In Java the unit test is written in a separate source file and tests public methods within the class. Tests in Java are known as JUnit tests. Third, contin-

uous integration allows repetitive deployment tasks to be automated and performed by a schedule and on demand. CruiseControl is a popular open source choice.<sup>43</sup> Once configured it detects any change to one's source control repository then checks out the latest code, builds it, running any tests as well. Not only does this save the developer time performing extra steps (which often would not be carried out), it can highlight errors very quickly; from test failures to missing dependencies on the build machine. Compilation is always simple on the developer's machine. The clean build machine is used to highlight hidden dependences. Both NCW and FBI follow these three pillars of software development. In addition, CruiseControl can automate packaged installers using IzPack. IzPack is a Java based GUI wizard installer.<sup>44</sup> Once CruiseControl has compiled the basic source IzPack creates an installer. CruiseControl also packages it ready for easy deployment on Windows, Mac and Linux. After each source change everything is deleted and built from scratch, ideally ending with one's deployment files ready to go, or an email highlighting the error and modification to the source since the last successful build. This is far quicker than a developer could do, saving time and allowing others to access the compiled code easily.

## 1.5 High Performance Computing

To generate large numbers of QSAR models more than a single workstation is required. For a combination of reasons including data set sizes and algorithm complexity, model generation time is increasing. The use of High Performance Computing (HPC) is prevalent within science already. Within Chemistry, molecular dynamics and *ab initio* calculations are common tasks for HPC. HPC clusters can readily generate large numbers of QSAR models. To take advantage of computational advances algorithms are being written with parallel coding. The use of parallel programming standards can reduce computation on multi-cores computers which are the de facto

standard now. Previously, HPC has utilised parallel computation across physical computers. This requires rewriting your software and having a cluster with suitable parallel connectivity. The latest parallel advances relate to the same hardware, not clusters. However, merging both forms of parallelism is possible and advantageous. Universities and industry alike both have HPC facilities available for researchers. HPC is dominated by the Linux operating system. Most HPC codes were never written for Windows. Scientific computing is firmly a task for Linux. Queuing systems are available for HPC as many users will want access to the compute resource. Sun Grid Engine is a well known example. It enables users to submit jobs to the queue and it will process them according to priority, hardware requirements and resources available. Traditional HPC is carried out on large, purpose built, homogeneous clusters. This typically represents a large investment from the institution, but has defined outcomes (in terms of compute ability). More recently a different model, Grid computing, has become popular. Grid computing varies from HPC as it does not rely on dedicated resources. The key concept of Grid computing is a dynamic pool of resource that is constantly changing for various reasons. With this set up queues are handled differently and rules are in place for when non dedicated resource is being used. The appeal of Grid computing is the ease at which the pool size can grow by utilising existing hardware. The standard workstation today is more powerful than dedicated HPC cluster nodes only a few years ago. However, after 5pm these workstations sit idle, wasting electricity and CPU cycles. Grid computing enables you to tap into this resource. From a cost perspective this is an attractive route to increase total compute resources. Condor is a popular program to manage a grid clusters. Grids are not bound by geographic location, unlike HPC which tend to represent a physical set of hardware. BOINC, another grid scheduler, runs many public research projects on volunteer computers across the

world. Using Condor one can achieve similar global pools.

Computational hardware and software develop at astonishing rates, and continue to do so. In order to exploit these advances our scientific software must make use of these developments. There is a cost with parallel or similarly advanced coding paradigms - they are increasingly complex to write. One requires a firm understanding of the base language before attempting to parallelise code. In this respect one needs to be a more accomplished programmer to fully leverage the resources available. Although challenging, the return is impressive. GPUs (Graphical Processing Units), another form of parallelism on GPUs opposed to CPUs can achieve speed improvements of one hundred fold.

## 1.6 Summary

QSAR needs to remain in favour with cheminformaticians. This can be achieved by using appropriate statistics that correctly represent the data and exploiting the chemical insight within the model. Suitable statistics is a matter of good practice and not using any one measure to determine how useful a model is. The presence of readily available interpretation would be desired by any modeller. Robust statistics and reliable interpretation will lead to greater confidence in models. This is not to say models are perfect predictions, they are not. However, if the one has chemical insight available and a measure of confidence then an informed decision can be made based on the data.

In the following chapters we will test various hypotheses. We will assess, by means of multiple comparison statistics, if random forests offer competitive predictive ability to support vector machine using multiple data sets. In order to assess multiple classifiers, data sets and parameters in a timely fashion high performance computing schedulers will be employed. It is already known that random forests are interpretable. However, the practicality of

accessing this interpretation has received less attention. The volume of 100 trees makes manual interpretation unattractive. We will investigate tools to assist in dealing with this number of trees. The TMACC descriptors have demonstrated predictive ability. They are interpretable but this has not been validated in detail. We will retrospectively test data sets to see if the TMACC interpretation matches that reported in the literature.



# Chapter 2

## Methods

### 2.1 Learning classifiers

While QSAR has a relatively long history for a computational method, modern QSAR borrows directly from the machine learning techniques of data mining. Neural networks and SVM are two such examples. The goals of QSAR and data mining do not overlap entirely. While both are concerned with generating a model to explain an unknown relationship, only QSAR has the secondary goal of model interpretation. The inner workings of a model based on chemical insight are invaluable, more so than mining credit card spending habits, for example. The ability to interpret is specific to the model. Some are straightforward; others are not. This would not be a primary concern when a new technique is designed. However, for use in drug discovery it is tremendously useful. There has been lots of work interpreting learning classifiers, e.g. neural networks, that were not interpretable originally.<sup>21</sup> It is arguable that interpretation is more important than improvements in prediction accuracy. Next, the various learning classifiers used in this thesis will be discussed.

### 2.1.1 Decision tree

The decision tree,<sup>45</sup> also known as recursive partitioning, is essentially a collection of decision stumps. Each stump is a split of the instances available according to the attribute with the greatest purity. The purity for a given attribute is the fraction of correctly classified instances. In the full tree each split results in two subsets of instances. Both of these are now split again on the purest attribute. This continues recursively, until a stopping criterion is met, typically a minimum number of instances. The splits are known as branches, while the collection of classified instances is a leaf. The tree is a collection of both. Afterwards a pruning algorithm is applied, which reduces the tree to the core components. The result is typically smaller than the original tree, which helps to reduce overfitting. Duplicate branches and/or leaves can be removed during this process and the tree can have a more lop-sided appearance. A decision tree modelling Lipinski-like rules<sup>46</sup> can be seen in Figure 2.1.

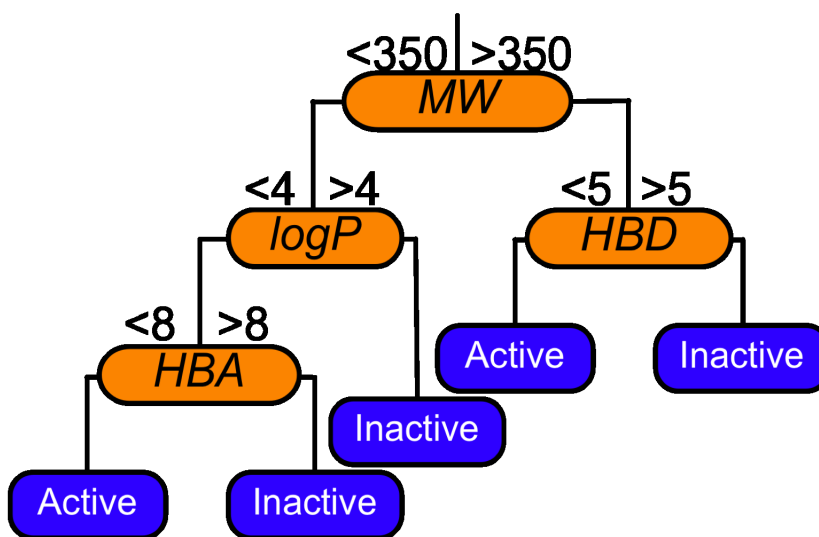


Figure 2.1: An example decision tree modelling Lipinski-like rules. Molecular weight (MW) should not exceed 350. The octanol-water partition coefficient ( $\log P$ ) should be below four. Hydrogen bond donors (HBD) should not exceed five. Hydrogen bond acceptors (HBA) should not exceed eight. Note the classifications of active or inactive refer to meeting Lipinski rules, not molecular activity.

Once a tree is built it can be used on external data to produce a classification and a set of rules determined by the path travelled through the tree. More than two splits per branch is possible. In Weka<sup>20</sup> only two splits per branch are allowed. However, multi-classification problems can be solved. Due to the construction of the tree, the time to generate it can be predicted in advance.

### **2.1.2 Ensemble methods**

To improve the performance of single classifiers, ensemble or meta approaches have been developed. Of note are bagging, see section 2.1.3, boosting, see section 2.1.4 and stacking, see section 2.1.5. The premise behind the technique is that 100 experts will, on average, provide a better answer than a single expert. If there are disagreements the majority and other methods can be used to determine the overall ensemble answer. There has been much research into how these techniques work.<sup>47-49</sup> Some results have proved difficult to explain, for example adding random variance to bagging will improve the performance. Boosting has been the subject of extensive analysis and not until its close relation to additive learning became apparent was it understood.<sup>50</sup> Bagging and boosting are concerned with using the same classifiers for the whole ensemble, whereas stacking can mix any number of different classifiers together. Both approaches work well, as one would expect by repeating the same technique multiple times or combining multiple results together. It should be noted that each replica is unique in some fashion. In bagging different data is used for each classifier. Stacking is also appealing, as there is no single silver bullet for the optimal classifier. It is widely accepted that one must try a selection of classifiers. However, as a rule of thumb certain techniques such as SVM and random forest generally perform well. There is always the exception, as demonstrated in Chapter 3, where a single decision tree outperforms several more advanced techniques,

including SVM and random forest.

The prediction error of an ensemble is related to the error of the individual classifiers:

$$MSE_{Ensemble} = \frac{1}{N} \overline{MSE} \quad (2.1)$$

where  $\overline{MSE}$  is the average mean squared error,  $MSE$  of individual members and  $N$  is the number of members in the ensemble. As the number of members increases the theoretical error is smaller than that of a single member.<sup>51</sup> Increasing  $N$  indefinitely will not yield constantly improving accuracy; the improvement may become very small. In addition increasing the ensemble size will increase compute time, which may be undesirable. The error of an individual member can be expressed in bias and variance:

$$MSE = Variance(\hat{\theta}) + Bias^2(\hat{\theta}) \quad (2.2)$$

where  $\hat{\theta}$  is a estimator of the quantity  $\theta$ . Model bias decreases as model variance increases. This would seem reasonable. As the model becomes more complex, the bias towards any single instance will decrease. There is a trade-off between these two functions. Ideally, both should be low. However, as one adds data to reduce the bias, the variance will increase. The ideal model will balance both functions, but still maintain good predictive power and avoid overfitting. Ensembles achieve predictive improvement by reducing the variance in their members and leaving the bias unaltered. By taking advantage of the bias and variance trade-off the ensemble can obtain a lower prediction error than any single member.

### 2.1.3 Bagging

While trees offer relative straightforward interpretation, they are less accurate than state-of-the-art techniques, such as SVM. Ensemble techniques

such as bagging offer improvements over a single classifier. Bagging is a simple, yet effective approach. One creates  $n$  bags of the original data set by sampling with replacement, thus allowing the same instance into the same bag. With each of the  $n$  unique bags, one builds a model using any chosen classifier, e.g. a decision tree. For each instance, there are  $n$  predictions and a majority vote decides the overall classification. One drawback of meta techniques is the increase in model generation time. This example takes  $n$  times longer than a single tree, but will build a more predictive model. This is an example of Occam's razor,<sup>52</sup> where one must balance performance with accuracy. Bagging lends itself to parallelism via the many methods now available in computer science, such as threading and grid computing. Bagging is depicted in Figure 2.2.

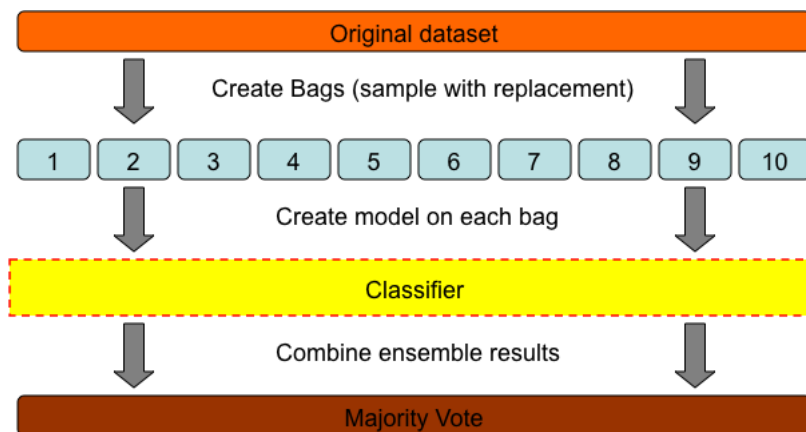


Figure 2.2: Overview of bagging. The training data set is used to generate ten re-sampled data sets known as bags. Each will be unique. A classifier is built for each bag. Different data ensures different models. A majority vote across all classifiers determines the ensemble prediction. Ten bags are for depiction purposes. Breiman used 50 in his original study, but found a lower number, could be optimal.<sup>53</sup>

### 2.1.4 Boosting

Boosting is a technique developed after bagging. It is widely reported to outperform bagging.<sup>54</sup> It works differently to bagging. A model is built using a chosen classifier. Each instance is assigned a weight depending on

how hard it was to classify correctly. Subsequent iterations involve improving the existing model by focussing on the instances poorly predicted in the previous iteration. With each iteration the model improves across the data set. There are various flavours of boosting, but all follow this general premise. Freund and Schapire were the authors of the original work known as AdaBoost.<sup>55-57</sup> Weak learners are used in boosting.<sup>58</sup> Weak learners are simple learning methods. The simplest decision tree is a decision stump, just one split. Boosting is very effective at improving the performance of decision stumps. Boosting does not work when the base learning is already successful at predicting the data as there is little or no error to optimise. Many comparison studies have focussed on bagging and boosting. Work by Dietterich shows that with little classification noise boosting outperforms bagging. However, when substantial noise is present bagging is superior.<sup>47</sup> Any classifier can be boosted, but it is not always feasible, as is the case with SVM. Diao et al. used only important instances, as determined by active learning, to be included in the training data.<sup>59</sup>

### **2.1.5 Stacking**

Stacking is another ensemble method.<sup>60,61</sup> Unlike bagging and boosting it is not restricted to using only one classifier for model building. Instead a selection of classifiers can be used in order to benefit from the different learning schemes. The overall classification reflects the combined predictions of classifiers. This often leads to better performance than a single classifier.

### **2.1.6 Random forest**

Random forest combines bagging and the random subspace method for decision forests.<sup>62</sup> The trees in a random forest differ to those previously described. First, only a random subset of attributes is available at each split point to determine purity, unlike all attributes in a typical tree. This can

be viewed as built-in feature selection, even though attributes available are randomly selected. Second, no pruning takes place. Third, in Weka's implementation there is no stopping criterion, leading to large, overgrown and overfitted trees. In later versions of Weka a maximum tree depth option was introduced allowing some degree of control over tree size. The result of these changes leads to a tree that is substantially larger than using the regular decision tree algorithm, even on the same data set. The tree is arguably overfit as the terminal leaves contain only a single instance. The lack of pruning leads to a very bushy structure. The size reduces the usefulness of the tree, as it is more complex to interpret.

Random forest is essentially bagging using decision trees with the modified trees. A random forest does outperform bagging with decision trees, see Chapter 3. The increased size of the trees in combination with the random availability of attributes is behind its improved predictive power. A forest construction is depicted in Figure 2.3. Multiple implementations of random forest are now available.<sup>63-66</sup> All are based on an ensemble of trees. Brieman's forest<sup>67</sup> is perhaps the most well known and used.

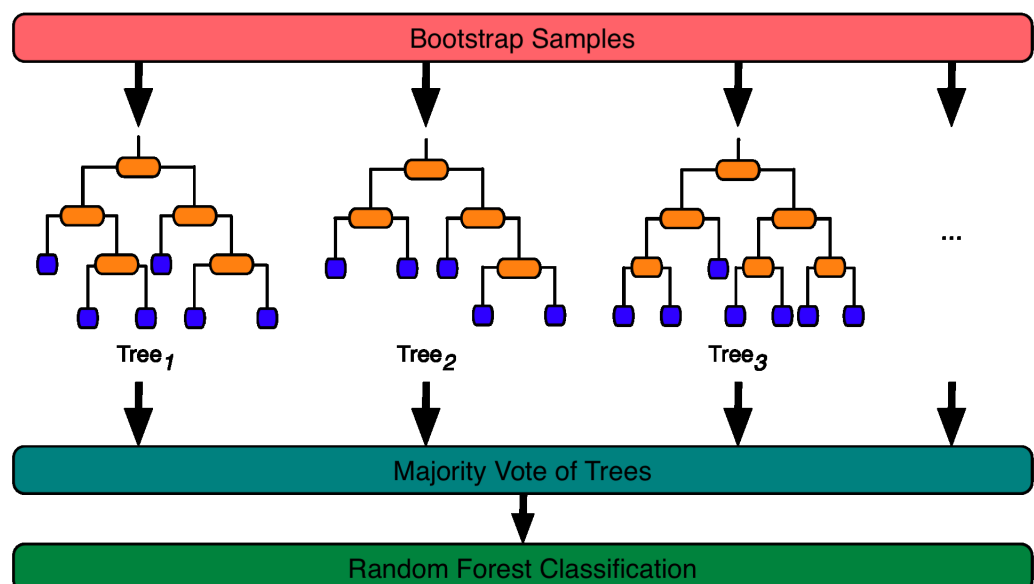


Figure 2.3: Overview of a random forest. Bootstrap samples are modelled using individual trees. The overall classification is based on a majority vote by the trees. The tree construction varies from standard decision trees and hence this is not bagging with decision trees.

### 2.1.7 Support Vector Machine

SVMs are not bioinspired, in contrast to trees or neural networks. While SVMs achieve excellent predictive power, they are not simple to interpret, and little work has been done in this area.<sup>68</sup> They are popular in a variety of disciplines as they perform well on various data sets. The drawback of this method is the model build time, due to the quadratic programming step of the algorithm for building a SVM. By nature they avoid local minima, thus aiding predictive power. Like most classifiers, researchers have modified SVMs to improve them. Most of this work has been focussed on the kernel, either creating new or modifying the common radial basis function (RBF) or Polynomial kernels.

Vapnik is credited with the original work on SVMs.<sup>69</sup> SVMs work effectively on both linear and non-linear problems. The SVM creates a hyperplane to split the data as accurately as possible, as shown in Figure 2.4. This is not a simple linear separation as by use of a kernel-trick the SVM transforms the feature space of descriptors. The linear hyperplane actually represents a nonlinear relationship of the data. This transformation is one of the problems of interpreting the model. Each kernel produces a different transformed feature space and thus several should be applied to find the optimal kernel. In addition to selecting the most appropriate kernel there are various other parameters that should be optimised. Some parameters affect the kernel; others do not. Non kernel specific parameters of importance are the complexity constant and  $\epsilon$ . The complexity constant controls the toleration of misclassified instances, the higher the value the fewer misclassified instances are permitted.  $\epsilon$  controls the round off value. Both these parameters can greatly affect the model generated. The SVM implementation in Weka uses sequential minimal optimization variant of SVM to reduce the time consuming quadratic programming step.<sup>70,71</sup>



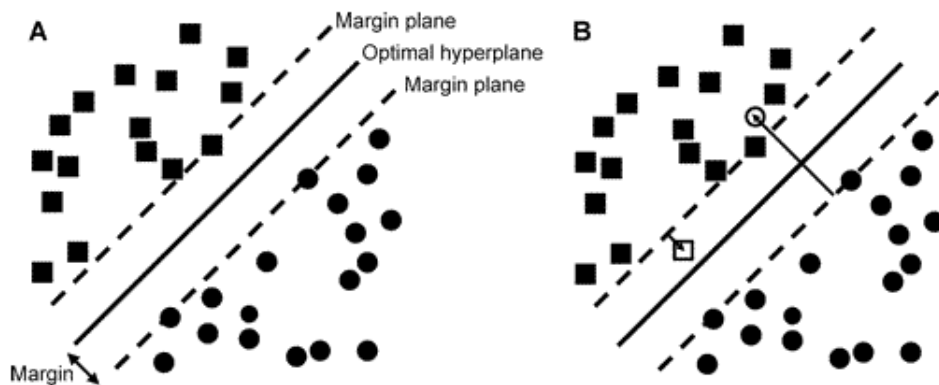


Figure 2.4: Separating hyperplane of the Support Vector Machine that maximizes the margin between two sets of perfectly separable objects, represented as circles and squares. (A) Optimal hyperplane that perfectly separates the two classes of objects. (B) Optimal soft margin hyperplane which tolerates some points (unfilled square and circle) on the wrong side of the appropriate margin plane. Reproduced with permission from Jorissen, R. N.; Gilson, M. K. Virtual Screening of Molecular Databases Using a Support Vector Machine. *J. Chem. Inf. Model.* **2005**, 45, 549-561 Copyright 2005 American Chemical Society

### 2.1.8 Partial Least Squares

PLS<sup>72</sup> is the more advanced version of Principal Component Regression (PCR). PCR is concerned with explaining the variation in the dependent variable. PLS improves PCR by taking both the dependent and independent variables into account to explain the variation. PLS is a popular technique in both cheminformatics and chemometrics. PLS produces an equation which explains the dependent variable in terms of latent variables. These latent variables are the combination of the independent variables with a weighting coefficient. The dependent variable,  $y$ , can be written as:

$$y = a_1t_1 + a_2t_2 + a_3t_3 + \dots a_nt_n \quad (2.3)$$

where  $a_n$  are coefficients of the latent variables,  $t_n$ .  $t_n$  is defined as

$$t_n = b_{i1}x_1 + b_{i2}x_2 + \dots b_{ip}x_p \quad (2.4)$$

where  $x_p$  are the independent variables.

Each latent variable is orthogonal to each other, providing maximum variation from the previous. The maximum number of latent variables is the lower of the number of variables or instances. Typically, one will only use a handful of the total latent variables. The coefficients,  $a_n$ , are of interest as they effectively weight or select the important features for the model. NCW uses these coefficients to help identify what partial activity each atom provides when using the TMACC descriptors.

## 2.2 Model statistics

There are various statistics to assess the predictive performance of a model. Classification and regression tasks require different statistics. As we deal with both, both are presented here. The general paradigm for creating a model is to select a training set to build the model upon. That model is then used to predict unseen data from a test set. For techniques which require parameter optimisation, one may tune the model on a third, unseen, validation set, therefore giving the model new data at each stage. The abundance of data can be a luxury not afforded to all. Cross-validation can be used to assess the model by splitting the data in multiple training and test sets. Even if sufficient data are available, the statistics produced from cross validation are commonly used to compare models.

### 2.2.1 Classification

The initial statistics from a classification experiment are the accuracy of the classifications. This leads to four numbers per class: true positive ( $TP$ ), false positive ( $FP$ ), true negative ( $TP$ ) and false negative ( $FN$ ). These four values are found in a confusion matrix. The number of classes determines the size of the matrix. A general confusion matrix is shown in Table 2.1. Knowing the incorrect predictions is perhaps more important than knowing

		Actual	
		Active	Inactive
Predicted	Active	TP	FP
	Inactive	FN	TN

Table 2.1: A general confusion matrix

what was correct. It is useful to see what instances proved a challenge for the model to classify. A single statistic of use is the percentage of correctly classified instances. However, this only reports the correct and incorrect percentages, it does not take into account the four possible results ( $TP$ ,  $FP$ ,  $TP$ ,  $FN$ ). This limitation does not apply to the Matthews correlation coefficient,<sup>73</sup> which takes all four into account, see Equation 2.5. Arguably the percentage of classified instances is an inferior statistic compared to the Matthews correlation coefficient.

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \quad (2.5)$$

The value represents a perfect prediction for +1, average random for 0 and an inverse prediction for -1. A limitation of the MCC is that it only applies to binary classification. For a greater number of classes one can use the Kappa coefficient. Fleiss' Kappa<sup>74</sup> not Cohen's Kappa<sup>75</sup> is used, as the latter is for two classes only.

$$\kappa = \frac{\bar{P} - \bar{P}_e}{1 - \bar{P}_e} \quad (2.6)$$

where  $1 - \bar{P}_e$  is the degree of agreement attainable above chance.  $\bar{P} - \bar{P}_e$  is the degree of agreement actually achieved above chance. Complete agreement gives  $\kappa$  of one. When there is no agreement  $\kappa$  is below zero.  $\bar{P}$  is the mean of  $P_i$  and  $\bar{P}_e$  requires  $P_j$ .  $P_j$ , the proportion of all assignments which were to the  $j$ -th category, is defined as:

$$p_j = \frac{1}{Nn} \sum_{i=1}^N n_{ij}, \quad 1 = \frac{1}{n} \sum_{j=1}^k n_{ij} \quad (2.7)$$

where  $N$  is the total subjects,  $n$  the number of ratings per subjects and  $k$  be the number of categories.  $n_{ij}$  represents the number of raters who assigned the  $i$ -th subject to the  $j$ -th category.  $P_i$ , is the extent to which raters agree for the  $i$ -th subject:

$$P_i = \frac{1}{n(n-1)} \sum_{j=1}^k n_{ij}(n_{ij} - 1) \quad (2.8)$$

$$= \frac{1}{n(n-1)} \sum_{j=1}^k (n_{ij}^2 - n_{ij}) \quad (2.9)$$

$$= \frac{1}{n(n-1)} \left[ \left( \sum_{j=1}^k n_{ij}^2 \right) - (n) \right] \quad (2.10)$$

We calculate  $\bar{P}$  and  $\bar{P}_e$  to complete the formula for  $\kappa$ :

$$\bar{P} = \frac{1}{N} \sum_{i=1}^N P_i \quad (2.11)$$

$$= \frac{1}{Nn(n-1)} \left( \sum_{i=1}^N \sum_{j=1}^k n_{ij}^2 - Nn \right) \quad (2.12)$$

$$\bar{P}_e = \sum_{j=1}^k p_j^2 \quad (2.13)$$

### 2.2.2 Regression

The correlation coefficient,  $r^2$ , represents the proportion of the variation within the model of the predicted values against the observed. It is a useful single value representation of the quality of the model. Several quantities can be calculated which lead to  $r^2$ :

$$\text{Total Sum of Squares: } TSS = \sum_{n=i}^N (y_n - \hat{y})^2 \quad (2.14)$$

$$\text{Explained Sum of Squares: } ESS = \sum_{n=i}^N (y_{calc,n} - \hat{y})^2 \quad (2.15)$$

$$\text{Residual Sum of Squares: } RSS = \sum_{n=i}^N (y_n - y_{calc,n})^2 \quad (2.16)$$

where  $N$  is the total number of molecules in the data set,  $n$  is the current molecule,  $y_n$  is the observed value and  $y_{calc,n}$  is the predicted value for molecule  $n$ . These three quantities are related as follows:

$$TSS = ESS + RSS \quad (2.17)$$

Therefore, there are several ways to calculate  $r^2$ :

$$r^2 = \frac{ESS}{TSS} \equiv \frac{TSS - RSS}{TSS} \equiv 1 - \frac{RSS}{TSS} \quad (2.18)$$

The  $r^2$  applies to the original model. Cross-validated data (see the next section) produce  $q^2$ , the cross-validated  $r^2$ .  $q^2$  is more interesting, as it measures predictive ability, opposed to how well all the data were modelled.  $q^2$  is calculated using PRESS, which is analogous to RSS, except the predicted, not observed value is used:

$$\text{Predictive Residual Sum of Squares: } PRESS = \sum_{n=i}^N (y_i - y_{pred,i})^2 \quad (2.19)$$

$$q^2 = 1 - \frac{PRESS}{\sum_{n=1}^N (y_i - \bar{y})^2} \quad (2.20)$$

Note that  $\hat{y}$  becomes the mean  $\bar{y}$  for  $q^2$ , as the mean for the appropriate

cross-validation group should be used, as opposed to the mean for the whole data set (which is not an accurate representation when cross-validating).

However, both  $r^2$  and  $q^2$  can be misleading. While a value of 0.75 may sound good, the actual data could reflect several large outliers, rather than good overall correlation. Therefore, any one single statistic should not be used to judge a model, but several in conjunction to avoid potential overconfidence in a model. In addition, even a seasoned modeller should visualise his/her predicted and observed data.

Measuring error for numeric predictions is more involved than simply a count or percentage of incorrectly classified instances. Two common error statistics are mean absolute error (MAE) and root mean-squared error (RMSE). Using the same notation as earlier, MAE is given as:

$$MAE = \frac{|y_{calc,1} - y_1| + \dots + |y_{calc,n} - y_n|}{N} \quad (2.21)$$

RMSE is defined as:

$$RMSE = \sqrt{\frac{(y_{calc,1} - y_1)^2 + \dots + (y_{calc,n} - y_n)^2}{N}} \quad (2.22)$$

### 2.2.3 Cross-validation

Cross-validation allows one to measure the predictive ability of a model. This can be done for extra statistics or when insufficient data are available to populate a test set. The training data are split into  $n$  folds; five or ten are commonly used. Each fold should be stratified, that is they contain a representative sample of the data set.<sup>76</sup> Nine folds are combined to form the training data on which the model is built; the remaining fold is used to test upon. This is repeated so each fold is the test set only once. All results are averaged to give the final predictive measure.

It could be argued that leave-one-out (LOO) cross-validation is the optimal method as it gives the most available data for training. The model is

only as good as the training data provided to it. So one should maximise the amount of data where possible. The downside of LOO is  $N$  models are required. For large data sets and/or slow learning techniques, this can lead to an unacceptable compute time. Experiments in Chapter 3 did run for a week using ten-fold cross-validation, which would not be practical for most purposes.

## 2.3 Nonparametric Multiple-Comparison Statistical Tests

Traditional pairwise statistics (such as the  $t$ -test) are inappropriate when making multiple comparisons. Here, we present a two-stage, nonparametric approach. The first test that should be applied is to determine if any significant differences between the classifiers can be detected. For this, the Friedman test is used.<sup>77,78</sup> The null hypothesis for the Friedman test is that there is no difference between any of the classifiers. If the null hypothesis is rejected, it does not determine which groups are different from each other; for this, a separate test is required. However, the Friedman test should be applied first, to determine if further analysis is justified. In the following, we shall assume that there are  $k$  classifiers and  $N$  data sets, and that all classifiers have been applied to all data sets, in each case yielding a real-valued measure of the performance of the classifier. In the work discussed in this thesis, we have used percentage accuracy, but other measures, such as the area under the receiver operating characteristic curve, are also possible.<sup>79</sup> The Friedman test proceeds as follows. For each data set, the performances of the classifiers are ranked, in ascending order; that is, the best performing classifier has rank  $k$ , the next best rank  $k - 1$ , and so on. Then, for each classifier  $j$ , the mean average rank across all data sets,  $\bar{R}_j$ , is calculated. The individual rank of a given classifier,  $j$ , and data set,  $i$ , is denoted by

$r_{ij}$ .

$$\bar{R}_j = \frac{\sum_i r_{ij}}{k} \quad (2.23)$$

The Friedman statistic,  $\chi_F^2$ , is then calculated as

$$\chi_F^2 = \frac{12N}{k(k+1)} \left[ \sum_j \bar{R}_j^2 - \frac{k(k+1)^2}{4} \right] \quad (2.24)$$

For sufficiently large  $k$  and  $N$  (Demšar<sup>80</sup> suggests  $k > 5$  and  $N > 10$ ), the Friedman statistic is distributed according to the  $\chi$ -squared statistic,  $\chi^2$ , with  $k - 1$  degrees of freedom. Critical values of  $\chi_F^2$  for smaller values of  $k$  and  $N$  are provided in ref<sup>81</sup>. In the event of tied ranks (i.e., two or more classifiers giving identical performances on a data set), a correction factor may be applied to  $\chi_F^2$ . One such correction (used, for example, in the statistical package R<sup>82</sup>) where the sum of the ranks across all data sets,  $R_j$ , is

$$R_j = \sum_i r_{ij} \quad (2.25)$$

and the correction statistic is

$$\chi_C^2 = \frac{12 \sum_j \left\{ \left[ R_j - \frac{n(k+1)}{2} \right]^2 \right\}}{nk(k+1) - C} \quad (2.26)$$

where  $C$ , the correction factor, is

$$C = \frac{\sum_{i=1}^m (t_i^3 - t_i)}{m} \quad (2.27)$$

$m$  is the number of groups of tied ranks for a classifier, and  $t_i$  is the number of ties in the  $i^{th}$  tied group. Equation 2.26 without the correction factor,  $C$ , gives the same result as equation 2.24. Iman and Davenport<sup>83</sup> demonstrated



that the Friedman statistic was too conservative and suggested the following improvement:

$$F_F = \frac{(N - 1) \chi_F^2}{N(k - 1) - \chi_F^2} \quad (2.28)$$

This is distributed according to the  $F$  distribution with  $k - 1$  and  $(k - 1)(N - 1)$  degrees of freedom. If the null hypothesis is rejected, then statistically significant differences between the classifiers are present. The Nemenyi test<sup>84</sup> can be applied to determine these differences in a pair-wise fashion. To compare classifiers  $a$  and  $b$ , the difference in mean ranks,  $\bar{R}_a - \bar{R}_b$ , is calculated. This value is compared to the critical difference,  $CD$ :

$$CD = q'_\alpha \sqrt{\frac{k(k + 1)}{6N}} \quad (2.29)$$

where the corrected  $q$  statistic,  $q'_\alpha$ , is given by

$$q'_\alpha = \frac{q_\alpha}{\sqrt{2}} \quad (2.30)$$

$q_\alpha$  is the critical value of the “Studentized range” statistic at a given level of significance,  $\alpha$ . Tables of the critical values of the  $q$  distribution can be found in statistical packages such as R and elsewhere.<sup>81</sup> If the difference between the average ranks is larger than  $CD$ , then the difference between the classifiers is significantly different at the specified value of  $\alpha$ . Demšar has discussed multiple classifier comparisons in greater detail.<sup>80</sup>

## 2.4 Topological maximum cross correlation descriptors

*James Melville is the author of the TMACC descriptors*

Topological maximum cross correlation (TMACC) descriptors<sup>85</sup> are 2D

descriptors for use in interpretive QSAR. They were inspired by the grid-independent descriptors (GRIND).<sup>86</sup> The GRIND descriptors, unlike TMAcc, are 3D descriptors. TMAcc descriptors vary in two ways. We replace force field interactions measured from a grid with atomic physicochemical values and the 3D distance between points is replaced with the 2D topological bond distance. TMAcc descriptors were validated against HQSAR,<sup>87</sup> as this 2D method is seen as comparable to the 3D QSAR techniques CoMFA and CoMSIA.<sup>88</sup> In addition, a large corporate study found HQSAR performed best over 1000 data sets.<sup>89</sup> TMAcc perform competitively against HQSAR, yielding higher cross-validated  $q^2$  in five out of eight data sets.<sup>85</sup> Therefore, we are confident these descriptors can produce predictive models. However, we now wish to validate their interpretative ability. This is only briefly examined in the original work. We perform a more extensive validation and have built further tools to enable this.

The TMAcc descriptors encode four atomic properties. Electrostatics are provided by Gasteiger partial charges.<sup>90</sup> Steric and polarisability are provided by Crippen-Wildman molar refractivity.<sup>91</sup> Hydrophobicity is from Crippen-Wildman  $\log P$ .<sup>91</sup> Finally solubility and solvation phenomena are calculated from Xu et al.  $\log S$ .<sup>92</sup> Solubility can be both a descriptor and prediction target. For example, the TMAcc descriptors are used with the solubility challenge data set in Chapter 5. The TMAcc descriptors individually only calculate predicted solubility for a given pair of atoms, a contribution of solubility, not the whole compound. It is not solubility but Xu parameters representing solubility and hence solvation phenomena. We are not assigning a solubility value to a pair of atoms independent of the rest of the molecule. For the solubility challenge in particular we wanted to test how accurate TMAcc descriptors are, given they include this solubility property. The method developed by Xu uses atom typing rules, determined by 76 SMARTS rules and two correction factors, hydrophobic carbon and

squared molecular weight. Multiple linear regression was used to determine the contribution of each atom type.

The values of each property are rescaled between -1 and +1, except  $\log S$ , which only has positive values. The absolute negative and positive values were used to rescale to -1 and +1 respectively. For the partial charges the absolute values were determined from the fragmentlike subset of the ZINC database,<sup>93</sup> consisting of 49134 molecules. The calculations were carried out with Open Babel 2.0.0.<sup>94,95</sup> By having a negative and positive range they can be treated separately. This gives seven total atomic properties: positive charge, negative charge, positive  $\log P$ , negative  $\log P$ , positive molar refractivity, negative molar refractivity and  $\log S$ . From these seven, 28 possible combinations of pairs are possible, and they are the base descriptors. For each pair of atoms in the molecule, all 28 descriptors values are calculated. The descriptor is additionally encoded with the topological bond distance. Therefore, the total number of descriptors for the dataset is 28 multiplied by the maximum topological bond distance. For QSAR datasets, this value is typically under a thousand. For a molecule with topological bond distance of ten, e.g. aspirin (Figure 2.5) one block of descriptors would be:

```
Maximum:ScaledAtomPartialPositiveCharge:ScaledAtomPartialPositiveLogP:0  
Maximum:ScaledAtomPartialPositiveCharge:ScaledAtomPartialPositiveLogP:1  
Maximum:ScaledAtomPartialPositiveCharge:ScaledAtomPartialPositiveLogP:2  
Maximum:ScaledAtomPartialPositiveCharge:ScaledAtomPartialPositiveLogP:3  
Maximum:ScaledAtomPartialPositiveCharge:ScaledAtomPartialPositiveLogP:4  
Maximum:ScaledAtomPartialPositiveCharge:ScaledAtomPartialPositiveLogP:5  
Maximum:ScaledAtomPartialPositiveCharge:ScaledAtomPartialPositiveLogP:6  
Maximum:ScaledAtomPartialPositiveCharge:ScaledAtomPartialPositiveLogP:7  
Maximum:ScaledAtomPartialPositiveCharge:ScaledAtomPartialPositiveLogP:8  
Maximum:ScaledAtomPartialPositiveCharge:ScaledAtomPartialPositiveLogP:9
```

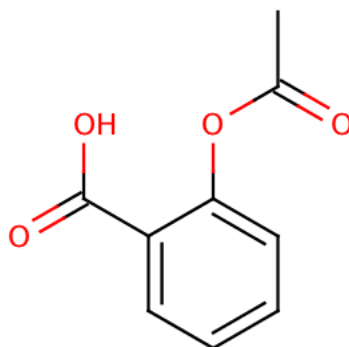


Figure 2.5: Aspirin. TMAcc descriptors are based on topological distances. The maximum distance here is 11 between the two carbonyl oxygens. There is more than one route to reach 11. The topological distance includes a zero term.

Here the descriptor positive charge-positive  $\log P$  is shown, with 11 potential bond distances (we include a distance term of zero, the value with itself). The total number of descriptors for aspirin would be 308; that is 28 TMAcc descriptors multiplied by 11 maximum topological bond distances. There are multiple pairs of atoms which are five bonds apart. The TMAcc only records the maximum value reported across the molecule. However, other pairs that register a value are recorded for the purpose of interpretation, and used later.

The equation for calculating an autocorrelation descriptor,  $x_{ac}$ , is

$$x_{ac}(p, d) = \sum p_i p_j \quad (2.31)$$

where  $p$  is a property, e.g. negative partial charge and  $d$  is the topological bond distance between atoms  $i$  and  $j$ . The sum is over all atom pairs separated by distance  $d$ . The TMAcc descriptors alter this equation in three ways. First, each atomic property that has a negative and positive value is treated as two separate properties. Second, we calculate cross-correlation in addition to the autocorrelation. We allow the property of atom  $i$  to vary from atom  $j$ , allowing positive charge-positive charge; nega-

tive charge-negative charge and positive charge-negative charge. Third, like the GRIND descriptors, we only keep the maximum value calculated for any given distance. The TMAcc equation is, therefore

$$x_{tmacc}(p, d, q) = \max(p_i q_j, q_i p_j) \quad (2.32)$$

where  $q$  is the property of the second atom. There are two terms to consider for each atom pair, because when  $p \neq q$ ,  $q_j p_i \neq p_j q_i$ .

While the TMAcc descriptors are interpretable in nature, used with PLS they become more powerful than a simple frequency of fragments. FGRAM details the atoms which contribute to each descriptor. For example the descriptor positive charge-positive charge with a bond distance of two has a value of 0.3. Together with the PLS coefficients, the FGRAM allows an activity contribution level to be assigned to each atom.

TMAcc and FGRAM generated were originally available only as a Java command-line application. To assess the interpretative abilities of the method more fully, it would be useful and obviate time consuming atom labelling if this could be automated. NCW - Nottingham Cheminformatics Workbench was created for this purpose. It is a GUI designed to accommodate in-house code, primarily TMAcc. It offers little benefit over the command-line for generating the original descriptors, but it has ChemAxon's Marvin<sup>96</sup> embedded, giving the user the ability to draw and edit molecules. NCW can generate models, interpret the descriptors using the PLS model, generated through Weka, analyse the results and display the dataset with a colour coding scheme in Marvin.

The PLS regression produces a model in the format

$$(desc_1 C \times desc_1 SE) + (desc_2 C \times desc_2 SE) + (desc_x C \times desc_x SE) \quad (2.33)$$

where  $desc_x C$  is the coefficient for  $desc_x$  and  $desc_x SE$  is the standard error for  $desc_x$ . Not all descriptors present in the dataset will be in the model. For each descriptor present, one calculates the activity contribution,  $AC$

$$AC = TMACC \times PLS_C \quad (2.34)$$

where  $TMACC$  is the TMACC value for this descriptor and  $PLS_C$  is the coefficient from the PLS for this descriptor. From the corresponding FGRAM one extracts the all atom pairs for descriptor  $i$ . All pair weights must be normalised to sum to 1.0. For each pair the pair activity contribution,  $P_{AC}$  is

$$P_{AC} = w_{pair} \times AC \quad (2.35)$$

where  $w_{pair}$  is the normalised pair weight. This enables one to calculate the contribution from each atom

$$atom_{contrib}x = \frac{P_{AC}}{2} \quad (2.36)$$

and

$$atom_{contrib}y = \frac{P_{AC}}{2} \quad (2.37)$$

It is assumed that each atom contributes equally to the overall contribution. This method is repeated for all descriptors present in the model and on each molecule in the dataset. For each molecule, one sums the individual atom contributions to give an overall contribution. One can take all atom contributions over the whole dataset, sort into ascending order and split into five equally sized groups. The first group represents very negative activity contribution, the last very positive activity contribution.

NCW has the machinery to generate the TMACC and FGRAM, using

Impact on activity	Colour
Very negative	Red
Negative	Orange
Neutral	Yellow
Positive	Green
Very positive	Blue

Table 2.2: Colour codes used in TMACC interpretation

Melville's code.<sup>85</sup> The code released with the original publication is taken directly, as obtainable on our website, <http://comp.chem.nottingham.ac.uk/download/tmacc>, and modified to enable NCW to access to the output programmatically. All further analysis is performed by new code in NCW. PLS is applied and the model used to determine the partial activity contribution from each atom. This value is converted to the one of five labels, represented by colour, then drawn and coloured by Marvin. Table 2.2 details the colour scheme implemented. Figure 2.6 depicts a sample ACE compound after complete processing through NCW.

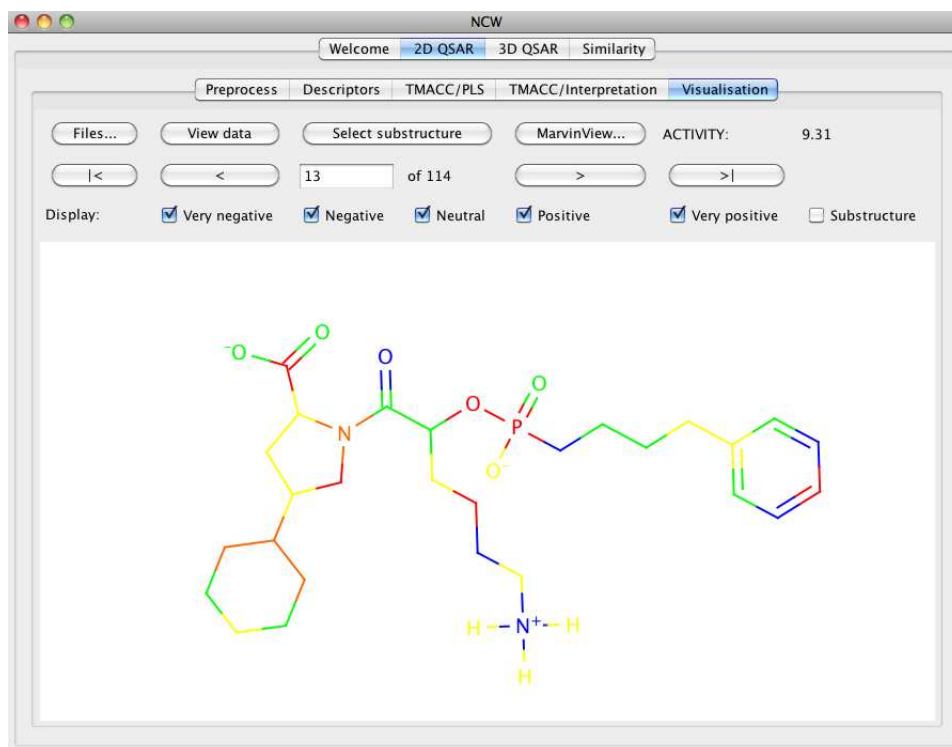


Figure 2.6: A molecule from the ACE dataset in NCW, after the each atom has been assigned an activity contribution by colour

This chapter has highlighted the numerous methods which will be used throughout the following chapters. Many concepts are central to Computer Science, specifically the fields of Data Mining and Machine Learning. This emphasis demonstrates the cross-disciplinary nature of this research, which is paramount in producing new novel approaches. Cheminformatics, like Computational Chemistry, is entwined with Computer Science. The next chapter compares a number of classifiers. Relative performance of multiple classifiers over multiple data sets is demonstrated using appropriate statistical tests.



# Chapter 3

## Contemporary QSAR classifiers compared

*This chapter is a reproduction of the peer-reviewed article.<sup>97</sup>*

### 3.1 Abstract

We present a comparative assessment of several state-of-the-art machine learning tools for mining drug data, including support vector machines (SVMs) and the ensemble decision tree methods boosting, bagging, and random forest, using eight data sets and two sets of descriptors. We demonstrate, by rigorous multiple comparison statistical tests, that these techniques can provide consistent improvements in predictive performance over single decision trees. However, within these methods, there is no clearly best-performing algorithm. This motivates a more in-depth investigation into the properties of random forests. We identify a set of parameters for the random forest that provide optimal performance across all the studied data sets. Additionally, the tree ensemble structure of the forest may provide an interpretable model, a considerable advantage over SVMs. We test this possibility and compare it with standard decision tree models.

## 3.2 Introduction

The pharmaceutical industry needs to address the increasing cost and time for drug development,<sup>98,99</sup> and *in silico* lead discovery and lead optimization are becoming increasingly important means to achieve this. Lead optimization often involves quantitative structure-activity relationships (QSARs),<sup>15,100</sup> which focus on predicting the biological activity of a compound from a vectorial representation of molecular structure. In the past few years, the computer science community has developed new machine learning algorithms<sup>20</sup> suitable for QSAR development. One success story is the support vector machine (SVM),<sup>101</sup> which has featured regularly in the bioinformatics<sup>102-104</sup> and cheminformatics literature.<sup>105-108</sup> Some studies have suggested that SVMs show improvement over neural networks for classification and QSAR.<sup>109-111</sup> The advantages offered by SVMs are robust predictions even with sparse and noisy data, because the formulation of the SVM solution ensures that there is only one minimum, thus avoiding the problems of premature convergence found with neural networks.

Another popular machine learning method is the decision tree, also known as recursive partitioning.<sup>45,112</sup> By partitioning the data into disjoint groups, decision trees produce nonlinear models that are interpretable, a valuable property of any statistical machine learning method when applied to QSAR studies.<sup>113,114</sup> A further improvement in the accuracy of decision tree predictions was achieved with the introduction of ensemble methods, where multiple trees are constructed and the outputs combined to produce a final prediction.<sup>48,115</sup> The most popular of the ensemble techniques are boosting<sup>56</sup> and bagging;<sup>53</sup> other variants are known as decision forests<sup>62,64</sup> and random forests.<sup>67</sup> We note that ensembles are not restricted to consist only of decision trees; other algorithms used for base learners include linear discriminant analysis,<sup>116</sup> neural networks,<sup>117,118</sup> and partial least-squares regression.<sup>119,120</sup>

Given the increasingly widespread adoption of these advanced machine learning techniques in chemometric<sup>121–123</sup> and cheminformatics fields,<sup>54,117,124</sup> it is timely to carry out rigorous assessment of these algorithms. Recently, Plewczynski et al. presented a comparison of some machine learning methods for virtual screening.<sup>125</sup> The focus of our study differs in several aspects. First, we seek to apply rigorous statistical tests to our results. It is still rare when evaluating classifiers in cheminformatics to make use of tests of statistical significance. However, there is a further caveat: in wide-scale comparisons of learning algorithms, it is not sufficient to use pair-wise statistical comparison tests, such as the paired *t*-test. Therefore, we make use of nonparametric statistical tests that are suitable for multiple comparisons. Second, apart from predictive ability, there are other requirements that make such tools valuable for mining chemical data. First, it is desirable to avoid having to manipulate multiple parameters to find the optimal performance for an algorithm. The requirement to tweak such parameters can lead to overfitting,<sup>126</sup> giving a false picture of predictivity, as well as being potentially time-consuming. Therefore, techniques with few parameters, or for which a widely applicable set of parameters can be obtained, are desirable. Second, an interpretable model is extremely valuable in extracting structure-activity relationships and relating the predictions of algorithms to physicochemical principles. SVMs are difficult to interpret; ensemble methods using decision trees may therefore have an advantage in this area. The translation of individual tree-based methods into classification rules is already widely documented,<sup>11,127–130</sup> but there have been only limited attempts to extend the interpretation to ensembles.<sup>54,131</sup> We investigate and highlight some challenges in this endeavour, compared to a single decision tree.

### 3.3 Methods

**Learning Algorithms.** For generating classifiers, we used the Java machine learning workbench Weka version 3.4.7.<sup>20</sup> Details of the algorithms we use in this study have been described in detail previously.<sup>20</sup> However, we provide a brief introduction. SVMs are considered to provide state-of-the-art classification performance, and we therefore use them as a benchmark to compare the performance of the ensemble methods, which make use of decision trees. SVMs create a separating hyperplane in the descriptor space of the training data, and molecules are classified on the basis of what side of the hyperplane they are located.<sup>101</sup> The advantage of the SVM is that, by use of the so-called kernel trick, the distance between a molecule and the hyperplane can be calculated in a transformed (nonlinear) feature space, but without requiring the explicit transformation of the original descriptors. A variety of kernels have been suggested, such as the polynomial and radial basis function (RBF). A polynomial kernel with an exponent of one reduces to the case of the linear SVM. Finding the optimal separating hyperplane requires the use of quadratic programming. As this can be time-consuming, we make use of the sequential minimal optimization<sup>71</sup> (SMO) variant of SVMs, which provides an approximation to the quadratic programming step. SVMs come with a range of parameters: those that affect the overall SVM and those specific to the kernel. Even with the speedup associated with SMO, a full search of the entire parameter space would be prohibitively time-consuming, and it is necessary to focus on the parameters that are most crucial to the performance of the algorithm, such as the choice of kernel.<sup>132</sup> We tune one kernel-independent parameter, two different kernels, and one kernel-specific parameter. The two kernels we investigate are the polynomial and RBF. In addition, the complexity constant is varied from the default 1 to 0.05 and 50. For each kernel, one kernel-specific parameter is altered, the exponent for the polynomial (default 1, altered to 2

and 3) and  $\gamma$ , the RBF width (default 0.01, altered to 0.001 and 0.1). The complexity constant controls the tolerance of misclassified molecules: the higher the value, the greater the importance of reducing misclassifications in the training model.

Decision trees recursively partition molecules into a series of disjoint sets or nodes, starting from a pool of all training data (called the root node). The node into which a molecule is placed is dependent on a threshold value of a particular descriptor (a branching rule). When a node is reached for which no branching rule is defined (a terminal node, or “leaf”), the molecule is classified, on the basis of the properties of the molecules with which it shares the node. This is normally achieved via majority vote. Each path through the tree from the root to a leaf can be extracted and represented as a classification rule, enabling interpretations to be made, which accounts for the popularity of this technique.<sup>11,127–130</sup> The tree-building algorithm used in this study is J48, a Java implementation of the C4.5 algorithm due to Quinlan.<sup>112</sup> A feature of the J48 algorithm is that it “prunes” leaves that do not contribute greatly to the predictive accuracy of the tree. This creates smaller trees, which may be more resistant to overfitting.

While decision trees have the advantage of being interpretable, their predictive abilities are normally inferior to that of more advanced techniques. Ensemble techniques attempt to compensate for the weakness of an individual tree by combining the predictions of multiple trees. The key to ensemble methods is therefore producing diverse selections of trees. This is normally achieved by training each tree on a different subset of the data. This can involve subsampling both molecules and descriptors. A commonly used technique is the bootstrap,<sup>133</sup> which samples the molecules with replacement. The resulting bootstrap sample is likely to contain duplicate molecules, while some of the original training data may not appear at all. The simplest ensemble method we study is bagging.<sup>53</sup> Here, the ensemble

is formed by simply repeatedly training trees on bootstrap samples of the original data. Classification of new molecules is by majority voting across all trees. Boosting<sup>57</sup> also uses bootstrap samples. However, the accuracy of the previous tree is used to bias the selection of molecules for the next sample, with poorly predicted molecules being given a greater chance of selection (or a larger weight) so that the next tree will focus on these more difficult cases. Again, voting is used to classify new molecules, but the vote is weighted to give more influence to trees with greater accuracy. Boosting is widely considered to be more accurate than bagging.<sup>134</sup> In our experiments we use the AdaBoostM1 algorithm.<sup>56</sup> A random forest<sup>67</sup> is similar to bagging, except that, as well as sampling training molecules randomly, only small subsets of descriptors are used to build each tree. Like bagging, classification is by majority vote. For bagging and boosting, we use the J48 algorithm to build the base decision tree classifiers. For random forest, we use random trees as a base classifier rather than trees built with J48. The main difference between J48 trees and random trees is that no pruning is carried out for random trees.

**Data Sets.** Eight data sets (Table 3.1) have been taken from the study of Sutherland et al.:<sup>88</sup> (1) a set of angiotensin-converting enzyme (ACE) inhibitors originally used for comparative molecular field analysis (CoMFA) modelling;<sup>135</sup> (2) a set of acetyl-cholinesterase (AChE) inhibitors, a subset of which was used in CoMFA studies;<sup>136</sup> (3) a set of ligands for the benzodiazepine receptor (BZR) also used for validating several QSAR methods;<sup>137</sup> (4) a set of cyclooxygenase-2 (COX2) inhibitors - a subset was used in CoMFA studies;<sup>138</sup> (5) a set of dihydrofolate reductase (DHFR) inhibitors, which were also used for comparative molecular similarity indices and analysis modeling;<sup>139</sup> (6) a set of glycogen phosphorylase b (GPB) inhibitors;<sup>140</sup> (7) a set of thermolysin (THER) inhibitors;<sup>30</sup> and (8) a set of thrombin (THR) inhibitors.<sup>141</sup>

Our study focuses on classification of activity, either active or inactive. The original data sets provide continuous numerical values for activity ( $\text{pIC}_{50}$  for the first five data sets and  $\text{p}K_i$  for the last three); each data set shows a uniform distribution of activity values. Therefore, the median activity value was used as a threshold between active or inactive compounds to create a 50/50 split of active/inactive observations. Balancing the data set in this way simplifies the validation of the classifiers and allows the use of percentage classification accuracy as a measure of classifier performance. However, the arbitrary split between active and inactive classes will actually make the problem harder for the classifiers. The compounds which fall near the median should be excluded from the model to create a real split between classes.

To create QSARs, descriptors of the molecules are required. We use the 2.5D descriptors generated by Sutherland et al.<sup>88</sup> using Cerius<sup>2</sup>.<sup>142</sup> In addition, linear fragment descriptors for these data sets were computed. For each data set, descriptors containing data on the atomic number, bond types, connectivity, chirality, and number of hydrogen atoms associated with each atom were generated for all non-branched molecular fragments of four to seven atoms in size.<sup>85</sup> The number of occurrences of each fragment in each molecule is recorded, producing an integer string as a descriptor of each molecule. The fragment data sets are of much larger dimensionality than the 2.5D descriptors. The original eight data sets are henceforth referred to as 2.5D data sets, while the new data sets are referred to as the linear fragment data sets.

To validate the performance of each classifier, we have used the percentage of correctly classified molecules from 10-fold cross-validation as the measure for the model, averaged over 10 entire repeats of the cross-validation, using different random seeds. In cross-validation, the data set is split into  $n$  folds; one fold is used for testing, the rest for training. This is repeated

data set	compound type	no. compounds.	no. descriptors	
			2.5D	fragments
ACE	angiotensin converting enzyme	114	56	1024
AchE	acetyl-cholinesterase inhibitors	111	63	774
BZR	benzodiazepine receptor	163	75	832
COX2	cyclooxygenase-2 inhibitors	322	74	660
DHFR	dihydrofolate reductase inhibitors	397	70	952
GPB	glycogen phosphorylase b	66	70	692
THER	thermolysin inhibitors	76	64	575
THR	thrombin inhibitors	88	66	527

Table 3.1: Summary of QSAR Data Sets

$n$  times, so all the data have been used as test data once. The  $n$  errors are averaged, giving the error rate for the data set. Ten is a commonly used value for  $n$ .<sup>76</sup> We stress that the reported cross-validated results are for genuine predictions and distinct from the internal validation used by some resampling techniques, which make use of the fact that approximately one-third of the original training data is unused after bootstrap sampling. Predictions on this unused data are known as out-of-bag (OOB) data. In addition to the classification accuracy, we assessed the robustness of the models using the standard deviation of the accuracy across the 10 repeats of the entire 10-fold cross-validation results, averaged for all data sets.

### 3.4 Results and discussion

**Classifier Accuracy.** We first consider the results of the 2.5D data set. The percentage of correctly classified molecules is given in the first five columns of Table 3.2. In general all classifiers perform reasonably, with classifications between 67 and 90%. The strongest trends are clearly dataset-dependent, with classification being most successful on the ACE and DHFR data sets, and least successful on the THER and THR data sets. As anticipated, the decision tree classifier is the least effective of all the classifiers studied, being the least accurate of the five methods studied in six of the



data set	tree	bagged tree	boosted tree	random forest	SVM	tuned forest <sup>b</sup>	tuned SVM <sup>c</sup>
ACE	86.9	86.5	86.6	85.4	<b>90.3</b>	89.3	89.9
AchE	70.6	71.6	72.7	72.6	72.0	<b>79.5</b>	74.3
BZR	71.7	75.5	75.4	74.0	77.4	79.5	<b>81.6</b>
COX2	75.6	75.7	<b>76.1</b>	73.4	75.4	75.7	75.2
DHFR	78.8	83.2	83.4	83.1	79.6	<b>84.9</b>	82.2
GPB	70.6	74.5	76.2	74.1	73.9	<b>76.7</b>	75.3
THER	67.2	69.2	67.8	69.7	69.5	<b>74.6</b>	<b>74.6</b>
THR	66.5	69.1	68.0	69.1	67.2	<b>72.5</b>	69.0

Table 3.2: Percentage of Correctly Classified Molecules for Different Classifiers on 2.5D Descriptor Data Sets<sup>a</sup>

<sup>a</sup>Values in bold denote the highest accuracy for that data set. <sup>b</sup>100 trees. <sup>c</sup>Polynomial kernel, exponent = 2, complexity constant = 0.05.

eight data sets. However, it still performs creditably, as the difference between the best and worst and classifier is no more than 6% on any data set. Additionally, it should be borne in mind that the average standard deviation of the cross-validation accuracies was 2.5%, with the smaller data sets (GPB, THER, and THR) showing larger standard deviations in the results than the larger data sets. Contrary to our expectations, SVM was the best classifier on only two data sets. However, the Weka default is to use a linear SVM; we assess the performance of a nonlinear SVM below.

The ensemble classifiers (bagging, boosting, and random forest) improved classification accuracy over the decision tree for all of the data sets except ACE. Within the ensemble techniques, there was little difference in performance. Boosting had a slight edge over bagging and random forest, outperforming both on five data sets. Having established the default behaviour of the algorithms as implemented in Weka, an obvious avenue of exploration for improving the performance of the ensemble classifiers is to increase the number of trees in the ensembles. Additionally, random forests have an extra parameter, the number of descriptors available to the tree-building algorithm when creating branching rules, which we investigated

separately. First, we increased the number of trees in the aggregates from 10 to 200 in steps of 10. Similar results were observed for bagging, boosting, and random forest. We concentrate on random forest in the following discussion, because it improved the most upon adding more trees, but the comments below apply equally to boosting and bagging. Accuracy increased upon adding more trees, reaching a maximum for all data sets between 30 and 50 trees. Little improvement in accuracy was observed beyond this point. The robustness of the accuracies was also improved with an increasing number of trees, although the improvement is not monotonic, as can be observed in Figure 3.1. Robustness is the average standard deviation of the accuracy across all 10 repeats of the 10-fold cross-validation results.

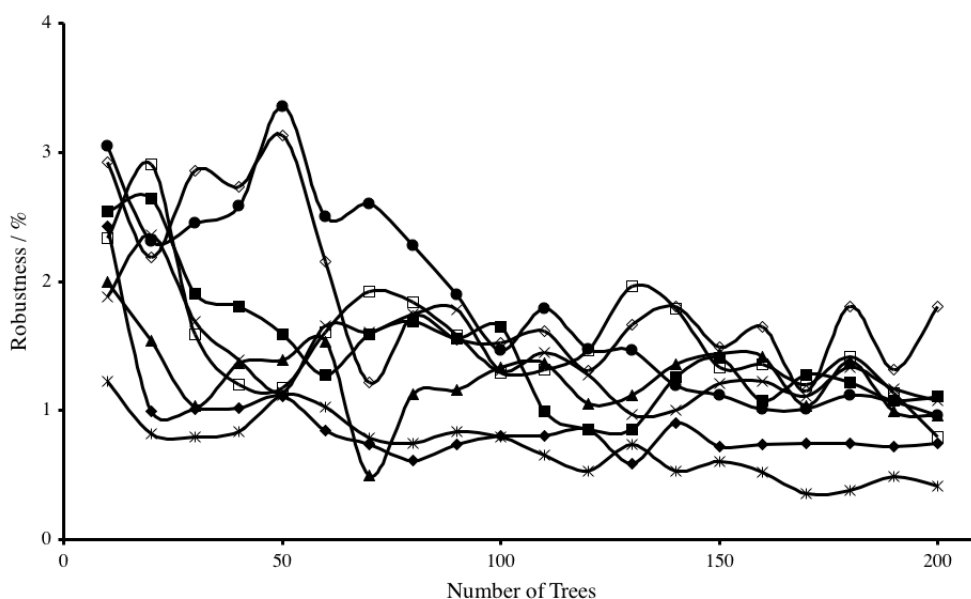


Figure 3.1: Robustness with increasing number of trees (on 2.5D descriptors). Legend: ACE ( $\blacklozenge$ ); AchE ( $\blacksquare$ ); BZR ( $\blacktriangle$ ); COX2 ( $\times$ ); DHFR ( $*$ ); GPB ( $\bullet$ ); THER ( $\square$ ); THR ( $\diamond$ )

Nonetheless, at 100 trees, the robustness of all trees is increased over that observed for 10 trees, with all standard deviations below 2%. No obvious improvement is observed upon increasing the number of trees to 200. Hence, we consider the 100-tree forest as “converged”. The accuracies for random forest classifiers built with 100 trees are shown in Table 3.2, in

the random forest column marked “tuned”. The improvement in accuracy over the 10-tree forest ranges from 2 to 7%. To establish whether the difference in performance was statistically significant, a two-tailed paired  $t$ -test<sup>81</sup> was carried out on 10 versus 100 trees, using 100 repeats (10-fold cross-validation repeated 100 times with different random seeds). For all data sets, the improvement in performance is statistically significant at  $p = 0.001$  for 100 trees. Second, we looked at the number of features available during tree construction, a parameter available only for random forests. By default, only  $\log_2 M + 1$  descriptors, selected randomly, are available for selection to construct each branching rule in a tree, where  $M$  is the total number of descriptors in the data set. For the 2.5D descriptors, this corresponds to six or seven descriptors per branch. We therefore looked at increasing the descriptor choice to 10, 20, 30, and 40 descriptors. To ensure our results were not dependent on the choice of random seed, we repeated the procedure 10 times, for two different forest sizes with 10 and 30 trees. The results showed that increasing the descriptor choice did not provide a consistent improvement, and in most cases, the default value was optimal. Therefore, we retained the default setting.

While increasing the number of trees in ensemble algorithms provides a clear means for optimizing performance, a principled approach to improving the SVM is more difficult to achieve, simply because it has a much larger number of parameters to modify. Our initial experiments suggested that two parameters had the greatest effect on SVM: the complexity constant and the type of kernel (polynomial or RBF). Associated with both kernels was a single parameter: the exponent for the polynomial kernel and the  $\gamma$  value (RBF width) for the RBF kernel. We therefore chose an optimal set of these parameters based on the mean accuracy across all eight data sets. For the polynomial kernel, setting the exponent to 2, with the lower complexity constant, produces the best predictions (see Table 3.3). Using

Complexity Constant	Polynomial exponent			RBF $\gamma$		
	1	2	3	0.001	0.01	0.1
0.05	68.7	<b>77.8</b>	75.5	53.5	51.4	59.0
1	75.7	76.1	74.0	52.5	67.2	75.2
50	74.1	74.0	73.6	71.1	76.5	<b>77.0</b>

Table 3.3: Mean percentage of correctly classified molecules for different parameters of SVMs on 2.5D descriptor datasets. Values in bold denote the highest accuracy for that kernel.

the RBF kernel, the SVM’s performance is more sensitive to the complexity constant. With the default complexity constant, an increased  $\gamma$  value gives improved results, which are similar to the default  $\gamma$  with a larger complexity constant. Increasing the  $\gamma$  with the higher complexity constant improves most datasets further, giving the best result with a RBF kernel.

Only the accuracies for the best SVM results are given in Table 3.2, under the SVM column “tuned”. The results show that a nonlinear SVM can improve performance for six out of eight data sets over the default linear SVM implementation in Weka. However, the performance of the 100-tree random forest is still superior to the tuned SVM for five of the eight data sets.

Having established some useful parameters and observed a pattern of behavior across the eight data sets with the 2.5D descriptors, we investigated whether similar results could be obtained using the fragment descriptors. Classification accuracies are given in Table 3.4. The SVM performance, after tuning the complexity constant and one kernel-specific parameter, is shown in Table 3.5. Altering the complexity constant on the polynomial kernel has little effect on performance, no matter which exponent is used. Using the RBF kernel with the default complexity constant and increasing the  $\gamma$  has a positive effect. The higher  $\gamma$  setting can give the best results.

Results are comparable to those of the 2.5D descriptors. The difference between the most and least accurate classifier was slightly increased, ranging between 2 and 8%, depending on the data set. Again, the ACE and DHFR

data set	tree	bagged tree	boosted tree	random forest	SVM	tuned forest <sup>b</sup>	tuned SVM <sup>c</sup>
ACE	80.4	82.0	81.0	80.5	78.9	80.0	<b>82.2</b>
AchE	64.1	68.0	68.8	70.5	69.4	70.5	<b>77.1</b>
BZR	74.0	75.0	69.8	67.3	74.0	68.7	<b>75.8</b>
COX2	71.1	71.5	71.0	68.1	<b>72.6</b>	68.7	71.1
DHFR	84.4	85.4	83.1	84.9	83.5	85.5	<b>86.5</b>
GPB	73.8	75.6	76.2	74.5	<b>77.4</b>	75.2	76.7
THER	72.2	75.8	75.5	75.4	75.3	<b>76.7</b>	73.4
THR	<b>71.5</b>	69.2	68.8	66.7	71.1	68.4	69.8

Table 3.4: Percentage of Correctly Classified Molecules for Different Classifiers on Linear Fragment Descriptor Data Sets<sup>a</sup>

<sup>a</sup>Values in bold denote the highest accuracy for that data set. <sup>b</sup>100 trees. <sup>c</sup>RBF kernel,  $\gamma = 0.1$ , complexity constant = 1.

Complexity Constant	Polynomial exponent			RBF $\gamma$		
	1	2	3	0.001	0.01	0.1
0.05	74.5	73.4	73.1	51.8	53.8	59.5
1	<b>75.3</b>	73.3	73.1	62.0	72.8	<b>76.6</b>
50	73.3	73.3	73.1	75.9	74.7	73.6

Table 3.5: Mean percentage of correctly classified molecules for different parameters of SVMs on linear fragment descriptor datasets. Values in bold denote the highest accuracy for that kernel.

data sets are the easiest to classify. However, a clear difference emerges when comparing performances based on the number of molecules in the data set. For the “small” data sets ( $\leq 100$  molecules), accuracies improve upon moving to a higher dimensionality of descriptors. Conversely, for the large data sets ( $\geq 100$  molecules), the majority of classifiers record a decrease in accuracy. As the larger data sets are described by a larger pool of descriptors, this may suggest that the increase in the dimensionality of the descriptor space outstrips the increase in information provided by the larger data sets; that is, the “curse of dimensionality”<sup>143</sup> is occurring. The decision tree classifier was not invariably the worst classifier tested, and for the THR data set, it was the best. However, for all other data sets, at least one of the ensemble algorithms was able to improve upon the decision tree performance. Bagging was the best at this, while boosting and random forest could improve over the standard decision tree in four and five data sets, respectively. As with the 2.5D descriptors, the SVM was the best classifier on two data sets. Increasing the number of trees to 100 in the random forest improved the accuracy for six out of eight data sets. However, there was less improvement in robustness in going to 100 trees, compared to the 2.5D descriptors, as shown in Figure 3.2.

A two-tailed *t*-test showed no significant difference for  $p = 0.001$  for the AchE, COX2, and GPB data sets. It is conceivable that the larger dimensionality of the fragment data set requires a larger number of trees in the ensembles; however, increasing the number of trees to 1000 did not result in any major increase in accuracy or robustness. Again, there was no clear improvement in the performance of the random forest when the number of descriptors available to the tree-building algorithm was increased. For the fragment descriptors, a RBF SVM was found to be optimal and gave the best performance of any algorithm on the fragment descriptors for four data sets. However, boosting, bagging, and random forests gave comparable

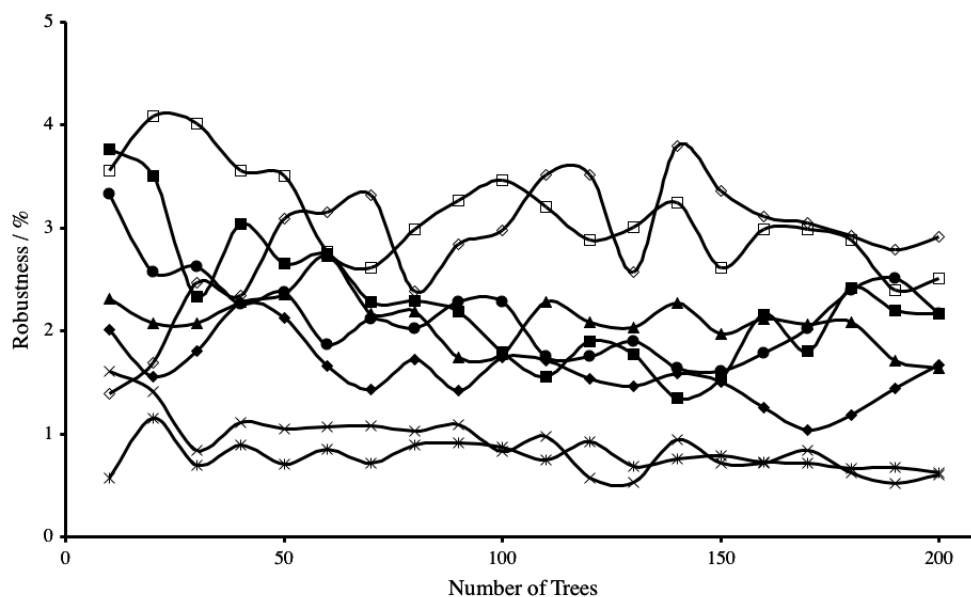


Figure 3.2: Robustness with increasing number of trees (on linear fragment descriptors). Legend: ACE ( $\blacklozenge$ ); AchE ( $\blacksquare$ ); BZR ( $\blacktriangle$ ); COX2 ( $\times$ ); DHFR ( $*$ ); GPB ( $\bullet$ ); THER ( $\square$ ); THR ( $\diamond$ ).

accuracies on all of the data sets, except AchE and COX2, where one or more of the algorithms struggled.

In order to put these observations on a more quantitative footing, we carried out a multiple-comparison statistical analysis, using the Friedman statistic, with a correction by Iman and Davenport, to detect the existence of a statistically significant difference between the classifiers. This test does not, however, identify *which* classifiers are different, only that a difference exists. To identify significantly different classifiers, we perform a post-hoc test using the Nemenyi test. Details of the procedure to carry out these tests are given in the Chapter 2. For these tests, we pooled the results for both sets of descriptors, giving  $N = 16$  data sets. Both the default and tuned versions of the SVM and random forest classifiers were considered separately, making  $k = 7$  classifiers. Carrying out the Friedman test with the Iman and Davenport correction indicated that statistically significant differences between the classifiers existed at the  $p = 0.01$  significance level. We therefore carried out the Nemenyi post-hoc analysis to determine which classifiers

were significantly different from each other. It is a known weakness of the Nemenyi test that it has a smaller power than the Friedman test. However, it was still possible to detect statistically significant differences between classifiers at  $p = 0.05$ . Thus, we can deduce that the performance of the tuned SVM (with the best average rank over all 16 datasets), and tuned random forest are significantly different from the performance of a single decision tree (with the lowest average rank over all datasets). However, it is not possible to detect whether SVM is statistically different from the ensemble algorithms, or if, in turn, the ensemble algorithms are significantly different from the use of a single decision tree.

**Interpretation of Tree Models.** Beyond assessing the predictive capabilities of the ensemble methods, we also sought to characterize the interpretability of the resulting models. Decision trees are widely used for their interpretability; therefore, an examination of the ensemble might provide similar insights. There are two main issues to account for in an interpretation of an ensemble: the size and shape of each individual tree and dealing with the large number of trees generated. Figure 3.3 shows two trees, one generated by the J48 algorithm (a) and one that is part of a random forest ensemble (b). For clarity, we have not displayed the value of the descriptor threshold applied at each branch. The trees were used to predict activity for the ACE data set using the 2.5D descriptors but have been chosen to represent the typical structure observed across all eight data sets. It is apparent from Figure 3.3 that decision trees generated by J48 are less “bushy” and less balanced than those used in random forest. This is a consequence of the pruning that takes place in the J48 algorithm, which is not applied to the trees grown for use in random forest. Terminal leaf populations are therefore on average smaller in trees in the random forest ensemble, and this makes interpretation less reliable for these nodes. Note also that the random tree contains the IC descriptor (a descriptor related to information



theoretic concepts of entropy) twice. An analysis of the descriptors used in the ensembles and the J48 trees shows a reasonable degree of overlap for most data sets. One exception to this was with the ACE data set, where J48 decision trees often contain a descriptor indicating the presence of a nitrogen atom type; these were rarely chosen in the ensembles.

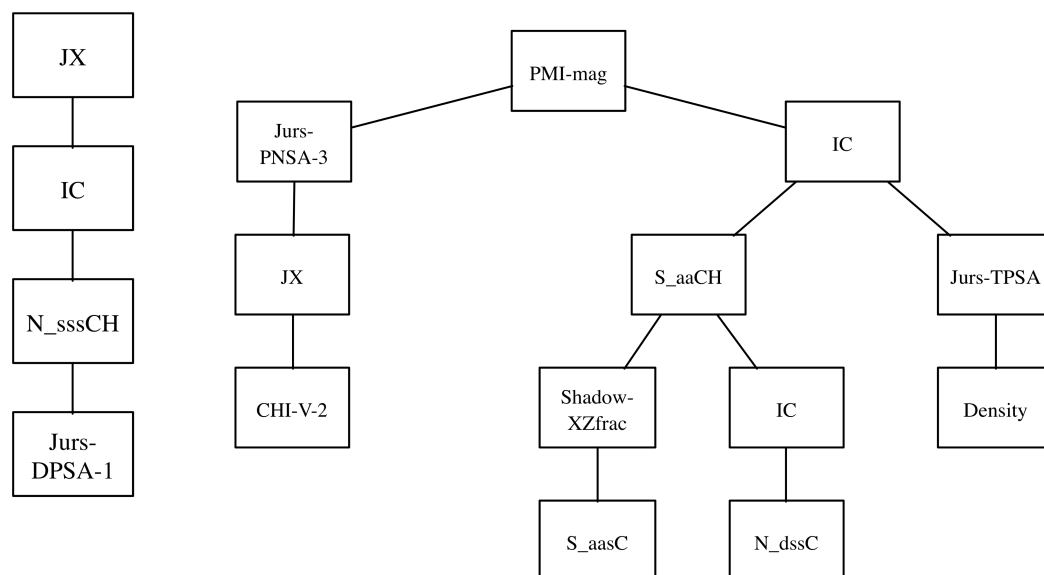


Figure 3.3: Decision trees classifying activity of the ACE data set generated by (a) the J48 algorithm with pruning and (b) the random tree algorithm without pruning. Descriptors definitions are in Table 3.6

We next consider the descriptor analysis of ensembles as a whole in more detail. We focus here on the frequency with which descriptors are chosen to appear in the trees. One obvious aid to interpretability would be if a subset of descriptors was chosen with a frequency much higher than that of others. To see whether this is the case, we recorded which descriptor was selected for each branching rule in a 100-tree random forest, for all eight data sets. We then plotted the number of descriptors as a percentage of the total number used in the forest against the number of unique descriptors selected. An example for the DHFR data set is shown in Figure 3.4.

Results for all the other data sets showed the same shape. The diagram can be interpreted similarly to a receiver operating characteristic curve. If only a few descriptors were selected by the tree-building algorithm, we

Abbreviation	Definition	Reference
CHI-V-2	Valence-modified connectivity index, 2nd order	144
Density	Density (spatial descriptor)	
IC	Multigraph information content index	
Jurs-DPSA-1	Difference in charged partial surface areas: partial positive solvent-accessible surface area minus partial negative solvent-accessible surface area	145
Jurs-PNSA-3	Atomic charge weighted negative surface area: sum of the product of solvent-accessible surface area x partial charge for all negatively charged atoms	145
Jurs-TPSA	Total polar surface area: sum of solvent-accessible surface areas of atoms with absolute value of partial charges greater or equal than 0.2	145
JX	Balaban index	146
N_dssC	Number of times that each intrinsic state occurs for a double/single/single bonded carbon cluster	147
N_sssCH	Number of times that each intrinsic state occurs for a single/single/single bonded methyne group	147
PMI-mag	Principal moment of inertia	148
S_aaCH	Summed differences between all intrinsic state values for an aromatic/aromatic bonded methyne group	147
S_aasC	Summed differences between all intrinsic state value for an aromatic/aromatic-single bonded carbon cluster	147
Shadow-XZfrac	Fraction of area of molecular shadow in the XZ plane over area of enclosing rectangle	149

Table 3.6: Descriptor definitions from the decision trees in Figure 3.3

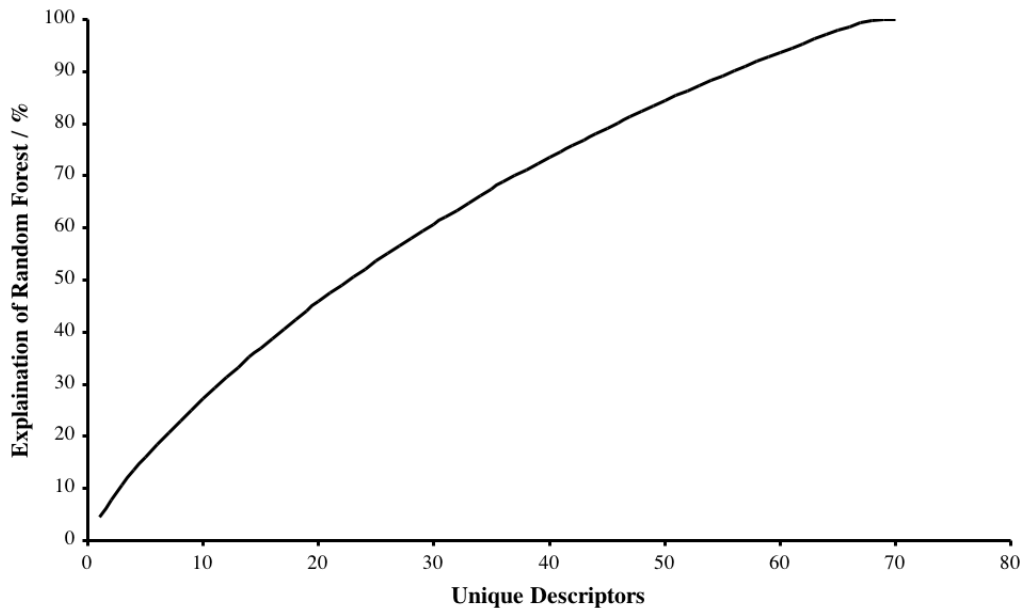


Figure 3.4: Percentage of model explained by unique descriptors for the DHFR data set.

would expect to see an initially steep vertical line at  $x = 0$ , indicating that most trees consisted of a few descriptors. This pattern is not observed to a large extent, although it can be seen that 50% of the branching points in the DHFR ensemble consist of 20 descriptors, which represents only 25% of the total number of descriptors chosen. Therefore, it seems that no “super descriptors” emerge from the random forest ensemble. This is perhaps to be expected, given that only approximately seven descriptors, chosen randomly from all descriptors, are available for selection at any given branch point during tree construction. A related issue is whether the same descriptors are chosen most frequently in ensembles of different sizes. We considered the top 10 most frequently chosen descriptors for this purpose and examined the effect of increasing the number of trees in the forest from 100 to 1000 in steps of 100 trees. Larger data sets show greater consistency: the same top 10 descriptors are present in all forests for the DHFR, the largest data set.

As the data sets grow smaller, there is less overlap between the ensembles, with only seven descriptors consistently found in all forests for ACE, AchE, and BZR. For the small data sets (GPB, THER, and THR),

at most three descriptors are in common across all forests. Despite this variability for some data sets, some commonly occurring descriptors can be identified. Atom type descriptors are commonly found, which describe hybridization and bonding information associated with an atomic centre; these are conceptually similar to the fragment descriptors. Additionally, some frequently occurring whole-molecule descriptors could be identified for some data sets. For the ACE data set, we identified IC and JX as falling into this category; they correspond to information entropy and the Balaban index,<sup>146</sup> respectively. The latter topological descriptor accounts for both cycles and conjugation in a structure, and active compounds in this data set are well-characterized by the presence of aromatic rings. Conversely, for the COX2 data set, electronic and partial surface area descriptors (Apol and Jurs-RPCG) are frequently selected. This would suggest that the overall electronegativity of the compounds is important for activity. For the DHFR data set, topological and partial surface area descriptors appeared in all forests (Kappa-3, Jurs-FPSA-3, and Jurs-FNSA-3). This shape and surface information can help distinguish between inactive compounds which may have longer, positively charged chains, for example, nitrogen cations, which are less favourable than short, negatively charged chains.

Using the tools developed in Chapter 4 we can extend our original interpretation. Based on a 100 tree forest built on the DHFR data set. Kappa-3, Jurs-FPSA-3, and Jurs-FNSA-3 were identified as descriptors which appeared frequently when altering the size of the forest. For this forest the top descriptors are S\_aaN, S\_sNH2, S\_aaCH, Kappa-3, S\_aaaC, S\_ssCH2, Jurs-FNSA-3, AlogP98, S\_aasC and Jurs-FPSA-3. Those previously highlighted are identified again. These top ten descriptors account for 29% of the branches in the overall forest. A total of 69 descriptors are used in the forest, the data set contains 70. Analysing the majority vote reveals that each tree on average incorrectly classifies 26 compounds. All the trees build

capable models, no single tree stands out as a poor predictor. Looking at the performance of individual compounds 46 are correctly predicted by all trees. 72 out of 100 votes is the lowest majority across the forest. 14 compounds received 80 or fewer correct votes.

### 3.5 Conclusions

Over the eight data sets, the support vector machine was the best of the classifiers studied, whether using the 2.5D descriptors or the fragments. We confirmed this superiority over a single decision tree through a multiple comparison statistical test. However, statistical significance should not be confused with practical significance, and the performance of boosting, bagging, and random forest was in most cases comparable with that of SVM, and in some cases superior. Furthermore, achieving the optimal performance for SVM can be a formidable task, because of the large number of parameters associated with it. We did not succeed in finding a set of parameters that was universally applicable. For example, a quadratic polynomial kernel was optimal for the 2.5D descriptors, while a RBF kernel was best for the fragment descriptors. Conversely, boosting, bagging, and random forest have fewer parameters, while performing competitively with the best SVM results. All of the ensemble algorithms benefited from the use of an increasing number of trees - these had a generally positive effect on both classification accuracy and robustness. We were able to identify 100 trees as being an optimal setting for most cases, and we recommend that users of the Weka machine-learning workbench use this value for cheminformatics applications, rather than the smaller number available as a default. Other packages such as R have a higher default value of 500 trees. However, this has an impact on training speed. For the larger datasets and high dimensionality descriptors training one forest could take up to a week. We did run some larger forests of 1000 trees but did not see statistically significant

improvement in accuracy. When considering interpretation a larger forest only increases the size of the model. This may be not desirable and could make interpretation more complex. To generate large forests in a reasonable time a parallel implementation of random forest were be required. The algorithm is ideally suited to being run in parallel. However, our recommendation would be to have a forest which is larger enough, for example 100 trees not 10. Adding thousands of more trees may not add any value, just computational cost. In general, all the algorithms we studied were marginally less effective for the more numerous linear fragment descriptors. While the dimensionality of these descriptors was large, even by current standards (1000 descriptors per data set), data sets are likely to become larger in the future, rather than smaller. Although ensemble algorithms like random forest were designed with high dimensional data sets in mind, cheminformatics data sets tend to be at least an order of magnitude larger, in terms of the ratio of descriptors to observations, than those commonly found in machine learning. An additional advantage of decision-tree-based methodologies is their potential to provide interpretable models. We have found that there are challenges associated with extending this interpretation to ensembles. In particular, trees in the ensembles are “bushier” than standard decision trees. A descriptor-level frequency analysis can provide some insight, but for the data sets studied here, there was a wide distribution of descriptors in the ensemble, so attempting to select a small subset of descriptors from the ensemble is somewhat subjective. Therefore, choosing a decision-tree-based learning algorithm over the SVM may be less advantageous than hoped. Despite the success of random forest and SVM on our data sets, a single decision tree was surprisingly competitive, and its interpretability is not in doubt. Clearly, therefore, along with more sophisticated ensemble-building algorithms, there is substantial scope for concomitantly advanced procedures for extracting information from decision tree ensem-

bles, and this will be the focus of Chapter 4.

# Chapter 4

## Random forest: an interpretable classifier

### 4.1 Introduction

Random forest is a capable and increasingly popular machine learning technique. Part of its appeal is the availability of interpretation, typically using a descriptor frequency count. We have previously shown that a random forest performs well compared to a SVM, widely considered the current benchmark. In addition, extensive parameter optimisation is not necessary and even if undertaken is a far quicker task, as fewer parameters are available. The insight that a model can provide is possibly of more importance than gaining an extra 5% in predictive accuracy. Insight from the model can be relayed to medicinal chemists, which can be used in lead optimisation strategies. Any information that explains the activity of a compound series is invaluable within drug discovery.

Within the QSAR community there are doubts over the usefulness of QSAR. Some question if QSAR contributes positively to drug discovery or reports chance correlations, not novel insight. The field must not stagnate by focusing purely on predictive performance. Recent work has highlighted



the concern with QSAR.<sup>27,150–152</sup>

Specific tools are required to interpret a random forest. While someone can interpret a single decision tree, it would be a painstaking task to review 100 larger trees from a forest. The construction of random trees in Weka leads to larger unpruned trees, making even a single tree unwieldy. In addition, the relationships between the trees could be of importance. We have explored various avenues to extract information from the forest.

This chapter will first validate the suitability of random forests on larger and more complex data sets. Here, complexity is in terms of a three class problem, where the balance of classes is skewed unfavourably. Most machine learning techniques struggle with skewed data. Therefore commonly used techniques to compensate skewed data are applied. Second the interpretability of the forest will be probed. Our in-house software, FBI, will demonstrate various features. Functionality of FBI includes descriptor frequency counts, tree depiction, transforming trees into SMILES and SMARTS notation, details on the majority vote and individual molecule performance.

## 4.2 Methods

The performance of random forests has already been demonstrated in Chapter 3. However, these data sets are smaller than the data sets typically used in pharma. We demonstrate the application of random forests on a larger and more challenging data set from GlaxoSmithKline. This data is proprietary, but is used here to demonstrate the predictive performance of the forest. Unlike the Sutherland and Weaver data sets, this has three classifications, and is thus a harder task. The distribution of the classes is uneven. This also poses a challenge to a data mining technique. Due to the greater availability of data at GlaxoSmithKline, three data sub-sets are possible. A large training set, a small hold or validation set to perform parameter

Data set size			
	Train	Hold	Test
Instances	7999	2217	4000
Class distribution			
	Low	Medium	High
Percentage	77	14	9

Table 4.1: Composition of GSK data set and distribution of classes

tuning and a medium-sized test set are used. This has the advantage of the data being completely independent. The model never sees the same data twice, as is the case with cross-validation. Each stage of the model has new, unseen, data. The data set itself comprises in-house data with 122 in silico descriptors. The descriptors represent common physical chemistry properties and e-states<sup>153</sup>, also generated in-house. The electrotopological state, or e-state descriptors generate a value for each atom of a molecule encoding the topological environment and electronic interactions with all other atoms. E-state indices can be formed for common functional groups. One such index denoted as “SssCH2”, represents CH<sub>2</sub> groups. “Sss” represent three single bonds.

The Table 4.1 summarises the data set composition. As one would expect, less high activity data is available than low or medium. It is the high activity compounds that are of most interest and the successful prediction of other high activity compounds. In highly skewed data sets data mining techniques have a tendency to classify everything as the majority class, in this case low activity. Statistically this would look like good performance, e.g. 90% is an excellent score. This underlines the importance of not relying on a single statistic. On closer inspection, this model is flawed for high activity prediction, as it successfully identified all the low activity compounds and labelled most other compounds low activity as well. Close attention to the confusion matrix is required for data sets like this. Skewed data sets are not a new problem and there are techniques to adjust the model. We

use cost sensitive classification<sup>20</sup> and over sampling.<sup>154</sup>

In this chapter, our data are quite different to those used in Chapter 3. Not only are they larger and thus allow train, hold and test data sets, but possess three classification classes. Having more data is advantageous as we can perform both the training and testing of the model with unseen data. Multiple classes are not a problem as long as the method can handle them. Modern implementation of these methods are suitable for multiple classes. SVM as implemented in Weka cannot handle multiple classes. This is overcome by performing multiple pairwise calculations of the classes to obtain a prediction for each class. The issue comes with the distribution of the classes. If they are unbalanced the model may be biased to the majority class, and this may not be the class of interest. Therefore, in addition to the normal procedure of model building and statistics, there are other techniques one would use in this situation. In cost sensitive classification, a cost matrix is used to penalise the learning algorithm when a certain classification is made. Weka supports this across most classifiers, including random forest. The cost matrix used is determined by trial and error, but a default matrix as in Table 4.2 is a reasonable starting point. Depending on the distribution of the classes one will need to adjust the costs to avoid a misclassification. The cost matrix can be used with any number of classes, although fine tuning will become more complicated as the number of classes increases. Ultimately one will find that one goal must be picked, e.g. optimise one class.

Another approach to unbalanced data is under and over sampling. These techniques relate to altering the data set composition. The reason the model is biased to the majority class is the high frequency of instances. Under and over sampling offer two approaches to remove the split and return the data set to equal distributions of the classes. Therefore, when the data is modelled, all classes have equal representation. Only the training data sets

Actual/Predicted	A	B	C
A	0	1	1
B	1	0	1
C	1	1	0

Table 4.2: Default cost matrix. No penalty for predictions on the diagonal, which are correct.

are modified. Subsequent data sets are not altered. Under sampling reduces the size of the data set so that all classes have the same number of instances as the minority class. This has the disadvantage that not all available data is used in the model. Over sampling is the opposite and increases the size of the dataset. Minority classes are duplicated to match the size of the majority class. The model will learn the same instances more than once and this will bias the overall model to recognise that class. To analyse the model, the confusion matrix is important to check for false negatives, typically in the minority classes, see Chapter 2. For statistics the Matthews correlation coefficient is not sufficient as this not a binary class problem. Instead one can use the Kappa statistic, as described in Chapter 2.

FBI was developed in this thesis to allow interpretation of random forest models. It is a Java-based GUI suitable for running on any major operating system. It does not have a batch or command-line version, as interpreting models is far less computationally demanding than generating them. To avoid rewriting core machine learning code Weka is included as its existing framework can be taken advantage of. This is possible thanks to its API, documentation and open source (Apache<sup>155</sup>) license. FBI was not developed for Weka’s random forest implementation. It can read saved models from other popular applications. However, with the presence of Weka its implementation is available to allow anyone to generate a forest.

FBI’s development followed the programmatic approach as described in the Chapter 1. All source code is held in our group subversion repository. JUnit tests were written and run whenever the code is compiled. For deploy-

ment IzPack is used to enable a simple wizard-based installation. An Ant build script (the Java equivalent of a Makefile) can compile, test and package the code into the IzPack installer. This script is what CruiseControl uses to independently build everything after a commit to the repository. Program documentation is available as HTML webpages or in PDF format. Documentation is written in reStructuredText, a lightweight markup language. A Python package, Sphinx<sup>156</sup>, will convert the source into the target format via a Makefile. The Ant build script also handles the documentation compilation. The installer for FBI along with documentation is available on our group website: <http://comp.chem.nottingham.ac.uk/download/fbi>. IzPack<sup>44</sup> provides native installers for users of Windows and Mac OS X, so users will not notice they are using Java. Using the installer can also, optionally, add Desktop and Start Menu shortcuts for the user. An installer is only as good as the uninstaller it provides. All IzPack installations are simple to completely remove. The Java runtime environment is the only prerequisite, which is commonly installed otherwise, it is readily and freely available from <http://www.java.com>. For the same reason that Weka is included with FBI so are Marvin and JFreeChart. Marvin provides the depiction of SMILES<sup>5</sup> and SMARTS,<sup>9</sup> while JFreeChart provides plots from some of the analyses. FBI was not written to provide its own API, but this could be introduced retrospectively.

Arguably the first task to perform when analysing a forest is to view the trees produced. Weka displays a text version of the trees it generates, that while technically accurate are not visually useful. Subsequently Weka introduced a tree visualiser that allows basic viewing and manipulation of a tree. FBI contains custom code, most of which has been contributed back to Weka that allows trees from a forest to be extracted and viewable in the tree visualiser. The tree component in the random forest was not compatible with the tree visualiser. In addition, instance data is also available by

clicking the various nodes when using the tree visualiser.

The concept of a tree is common and highly studied in Computer Science. Graph theory is one implementation of a tree structure, a selection of nodes and edges (or leaves and branches). Cheminformatics also uses graph theory to encode chemical structures as atoms (nodes) and bonds (edges). Given we are interested in chemical insight to our forest it was opportune to convert our tree structures into chemical entities (although they will not contain any actual chemical meaning). With the forest in a popular chemical format there are many pre-existing tools with which they can now be processed. SMILES<sup>5</sup>, simplified molecular input line entry specification, is a popular and de facto 2D chemical format. SMILES are simple and human-readable where letters represent atoms and bonds are implicit. Benzene is encoded as c1ccccc1, where the ring opening and closing is signified by the same number. Lower case letters encode for aromatic atoms, while upper case represents aliphatic atoms. Therefore, cyclohexane would be C1CCCCC1. Compounds with branches are handled by parentheses, for example, acetic acid, CC(=O)O. Bonds are denoted by equals or hash symbols for double and triple bonds respectively. There are only a handful of simple rules to enable one to interpret SMILES, which is part of the reason they are so popular. A valid SMILES representation can start from any point of a molecule. Therefore, there are many potentially valid and unique SMILES for the same compound. Canonicalisation is a process to produce a unique SMILES representation. This is particularly useful in databases or when comparing SMILES against each other.

Actual/Predicted	L	M	H
L	3026	13	23
M	516	14	40
H	272	17	79

Table 4.3: Confusion matrix for the default random forest reporting the test set results.

Actual/Predicted	A	B	C
A	0	2	5
B	20	0	5
C	20	10	0

Table 4.4: Cost matrix for the cost sensitive random forest. No penalty for the (correct) predictions on the diagonal.

## 4.3 Results and discussion

### 4.3.1 Larger and skewed data

Using the training data to build our forest of 100 trees, we report the results of the training data (4000 instances) and apply techniques to take account of the skewed class distribution. Using just the random forest, we obtain 78% correctly classified compounds and Kappa statistic of 0.17. Table 4.3 shows the confusion matrix. It is clear that the bulk of compounds have been predicted as low activity. We are more interested in the medium and high activity classes, of which only 2.5% and 21.5%, respectively, were correctly predicted. The overall correctly classified percentage is misleading, but the low Kappa statistic highlights the incorrect classifications. The low activity class is nearly perfectly predicted at 98.8%.

By applying a cost matrix, we can influence the forest not to classify

Actual/Predicted	L	M	H
L	2758	205	99
M	363	114	93
H	150	81	137

Table 4.5: Confusion matrix for the cost sensitive random forest reporting the test set results.

Actual/Predicted	L	M	H
L	2741	214	107
M	356	121	93
H	144	86	138

Table 4.6: Confusion matrix for the oversampled random forest reporting the test set results.

medium and high activity instances as low activity. Several cost matrices were used. Table 4.4 is what produces the confusion matrix in Table 4.5. The correctly classified rate was 75% and the Kappa statistic 0.30. The use of a cost matrix improves the overall correct classification, as reflected by a higher Kappa statistic. The total number of correct classifications is lower, but the improvement to the medium and high activity compounds is obvious. 20% and 37.2% of the medium and high activity compounds, respectively, are correctly classified. This represents a two-fold improvement for the high activity compounds and eight-fold for medium activity compounds. The medium activity compounds are harder to predict, as they have boundaries with both low and high activity compounds. Further attempts to improve the cost matrix led to either the medium or the high correct classification falling at the expense of the other. The low activity compounds are not as well predicted in this model. However, the correct percentage is still very high and our focus is the more active compounds.

We applied over sampling to the training data set, which increased its size. The test set remains the same. Using just over sampling achieves similar results to the cost sensitive classification. The fraction of correctly classified compounds was 75% and the Kappa statistic was 0.30. The confusion matrix is very similar, as seen in Table 4.6 and demonstrates the effectiveness of this technique for handling the skewed classes. The percentage correct by class is essentially the same. However, we would ideally want the percentages to increase nearer 50% for all classes.

The techniques complement each other, and combining them improves



Actual/Predicted	A	B	C
A	0	10	5
B	20	0	5
C	20	15	0

Table 4.7: Cost matrix for the cost sensitive and oversampled random forest. No penalty for (correct) predictions on the diagonal.

Actual/Predicted	L	M	H
L	2314	600	148
M	215	238	117
H	80	132	156

Table 4.8: Confusion matrix for the cost sensitive and oversampled random forest reporting the test set results.

the results. A different cost matrix is used, as the composition of the training data set has now changed, shown in Table 4.7. The overall correctly classified rate is 68%, while Kappa remains at 0.3. The confusion matrix is shown in Table 4.8. Clearly, the total number of correct classifications is lower than previously. However, that is due to the majority class only being 75% correctly classified. The classes of interest have both risen to over 40%, especially the middle class which often suffers at the expense of the terminal classes. Random forest continues to prove its competence at classification of pharmaceutical data of varying type. The use of additional machine learning techniques enables it to perform well with skewed data sets.

### 4.3.2 Interpretation

When it comes to interpreting a forest, it is clear that visual inspection alone is insufficient. In fact, by default Weka does not display the trees from the forest. This highlights the different priorities between the machine learning and cheminformatics communities. Even with the trees available, they are far too large and unwieldy to be of any use, especially when presented as text. A tool was needed to assist with forest interpretation and

handling the model. A Java application called FBI, Forest Based Interpretation, was written to house all the required tools to create and interpret models. Java was chosen, due to its platform independence. This became especially useful as FBI was developed on a Mac, yet deployed on Windows and Linux. FBI contains a copy of Weka with some slightly modified files to enable one to obtain the trees produced by the forest. All further analysis was performed outside Weka, to enable FBI to run newer versions of Weka.

When starting with FBI one can either generate a new random forest or load a previously saved model. Once the forest is built, all analysis options become available. Previously, descriptor importance was ascertained by counting the frequency of a descriptor in the trees as each split undergoes attribute selection, even though the pool of available descriptors is only a random subset. For the following examples, the ACE data set from Sutherland and Weaver<sup>88</sup> was used to create a 10 tree forest. Figure 4.1 depicts the summary of descriptors across the forest. Graphs are produced in FBI using the open source library JFreeChart<sup>157</sup>.

It is clear there are no super-descriptors, as Chapter 3 details. However, it is reasonable to assume that the more commonly picked descriptors are better at classifying this data set. FBI can drill further down to graph by each tree in the forest. Tree 10 is shown in Figure 4.2. This is a small data set and hence there is a lower overall number of descriptors. FBI provides the numbers behind the graph as well, for the user to save and use elsewhere. As well as the most common descriptors being of potential interest, the descriptors at the top of each tree correctly classify a larger number of instances. Therefore, they are also of interest. FBI can summarise the top descriptors to a given depth.

One can view the trees in multiple formats. Firstly, Weka includes a tree visualiser. This has been modified to enable easy viewing of multiple trees, as shown in Figure 4.3. Secondly, we can produce SMILES<sup>5</sup> and

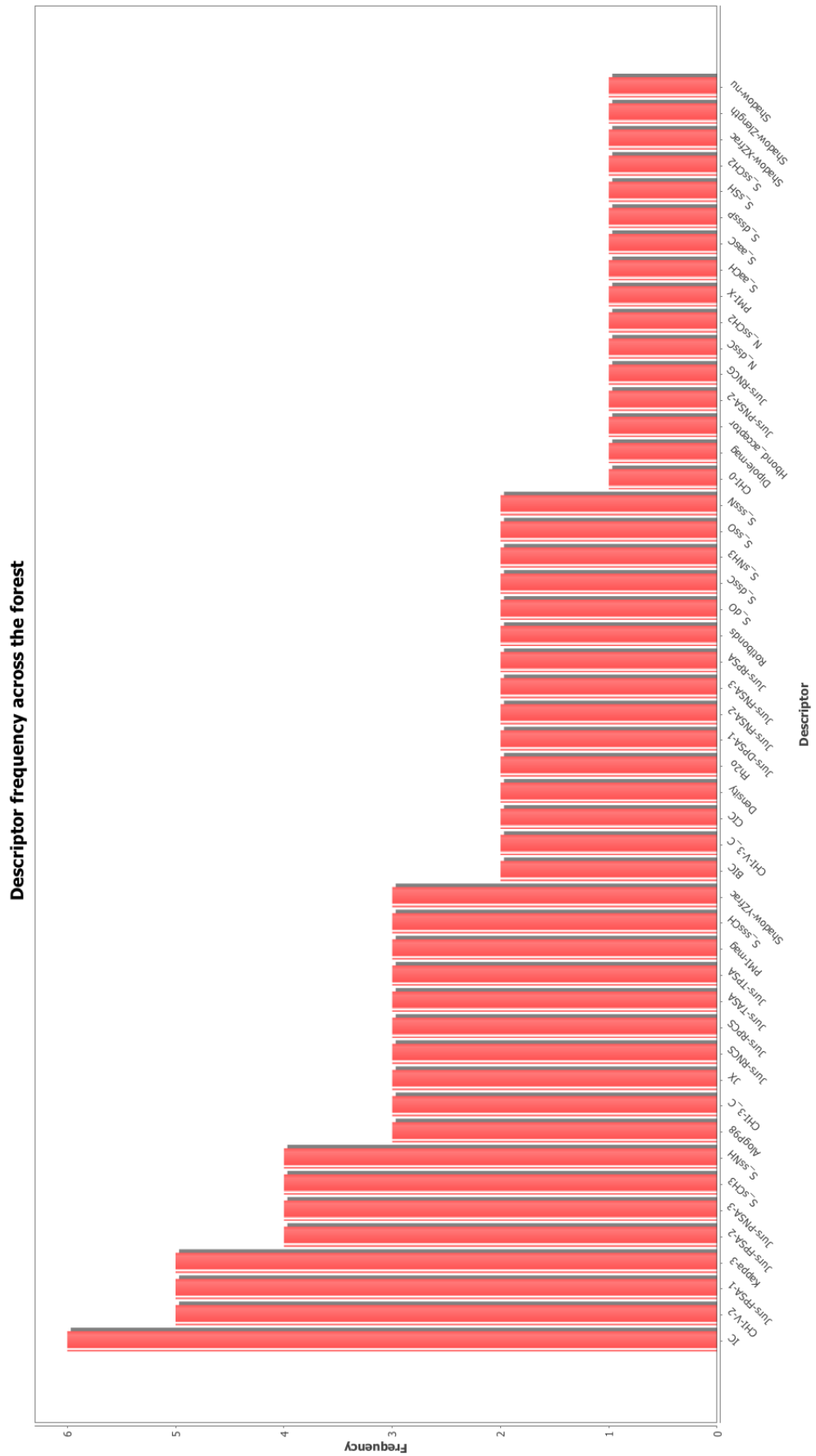


Figure 4.1: Summary of descriptor frequency across a forest using the ACE data set

SMARTS<sup>9</sup> trees. Our motivation for converting the trees into a chemical format is to allow ease of manipulation and enabled the use of existing tools. The depictions from the chemical trees are instantly more familiar to chemists and more meaningful than the visualisation that Weka provides. The SMILES trees are encoded using boron for the branches and carbon for the leaves. In order to satisfy valency oxygen is used as the root node. The SMARTS tree gives each descriptor in the tree a unique number. This is encoded as the atomic number. Using SMARTS allows an extra dimension of information to be encoded, namely identifying the descriptor. Just like the SMILES tree the topology is also encoded. Identifying the descriptor is useful for further analysis. The root node is represented with the wildcard (\*) atom. All information encoded in the SMILES and SMARTS trees hold no chemical meaning; atom types were picked for convenience. The information encoded in the trees is different. The SMILES tree details the topology of the tree, but tells one nothing about the descriptors. The SMARTS tree additionally encodes the descriptor detail with different descriptors represented as ascending atomic numbers. Samples of both are seen in Figures 4.4 and 4.5. Due to the tree structure, which is not typically found in compounds, the actual SMILES and SMARTS representations are parenthesis heavy and not particularly readable. The original SMILES representation is O(B(B(C)B(C)(C))B(C)-B(C)B(C)(C))B(B(C)(C))B(C)B(C)B(C)B(C)(C), which is canonicalised to B(B(B(B(C)C)OB(B(B(B(C)C)C)C)B(B(C)C)C)C)(B(B(C)C)C)C. The SMARTS is \*(-[#1](-[#8](-[#9])-[#9](-[#10])(-[#10]))-[#8](-[#11])-[#11](-[#12])-[#12](-[#13])(-[#13]))-[#1](-[#2](-[#3](-[#3]))-[#2](-[#4])-[#4](-[#5])-[#5](-[#6])-[#6](-[#7])(-[#7])). These trees have no chemical meaning but useful information about the subgraphs is encoded in a format for which various tools are readily available. For example, subgraph fragments can be searched across all the whole forest to see if the same path repeats, which would

indicate this particular path encodes some importance of the selected descriptors.

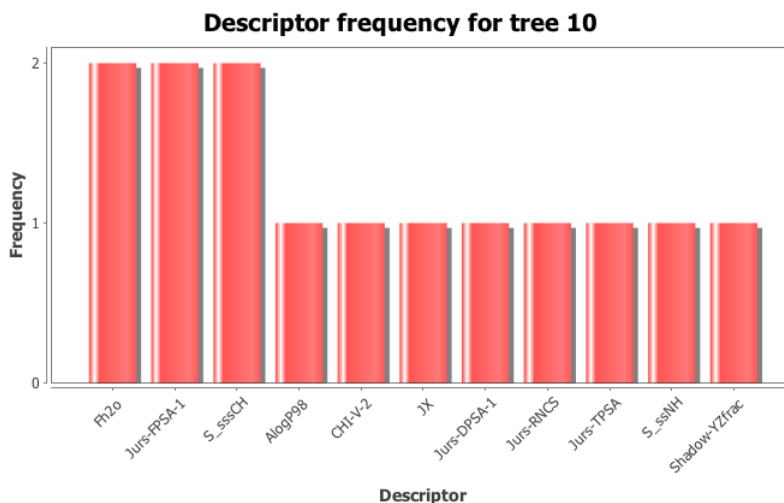


Figure 4.2: Summary of descriptor frequency across tree 10 using the ACE data set

As FBI contains Weka, which has a fully fledged API, one can take advantage of this. For example, we can produce a scoring matrix based on trees or compounds. They contain binary values for whether or not that tree or compound was successfully classified. Weka also contains clustering algorithms and a clusterer visualiser. Using the existing tools in Weka, we can cluster our scoring matrix. It identifies three clusters from this matrix. Using the GUI interface the user is able to identify the instance number of the cluster members.

The overall classification is given by the majority vote of the trees. We can now produce these numbers and plot them, as in Figure 4.6. No single tree has a disastrous performance. This data set is fairly straightforward to classify. Likewise, we can plot the performance of each molecule. A so called molecule performance graph highlights which molecules posed the biggest challenge to classification. This is shown in Figure 4.7. Most molecules are unanimously correctly predicted. However, there is a handful that are misclassified by a few trees. These molecules are of interest to see why some trees could not successfully classify them. In order to assist with

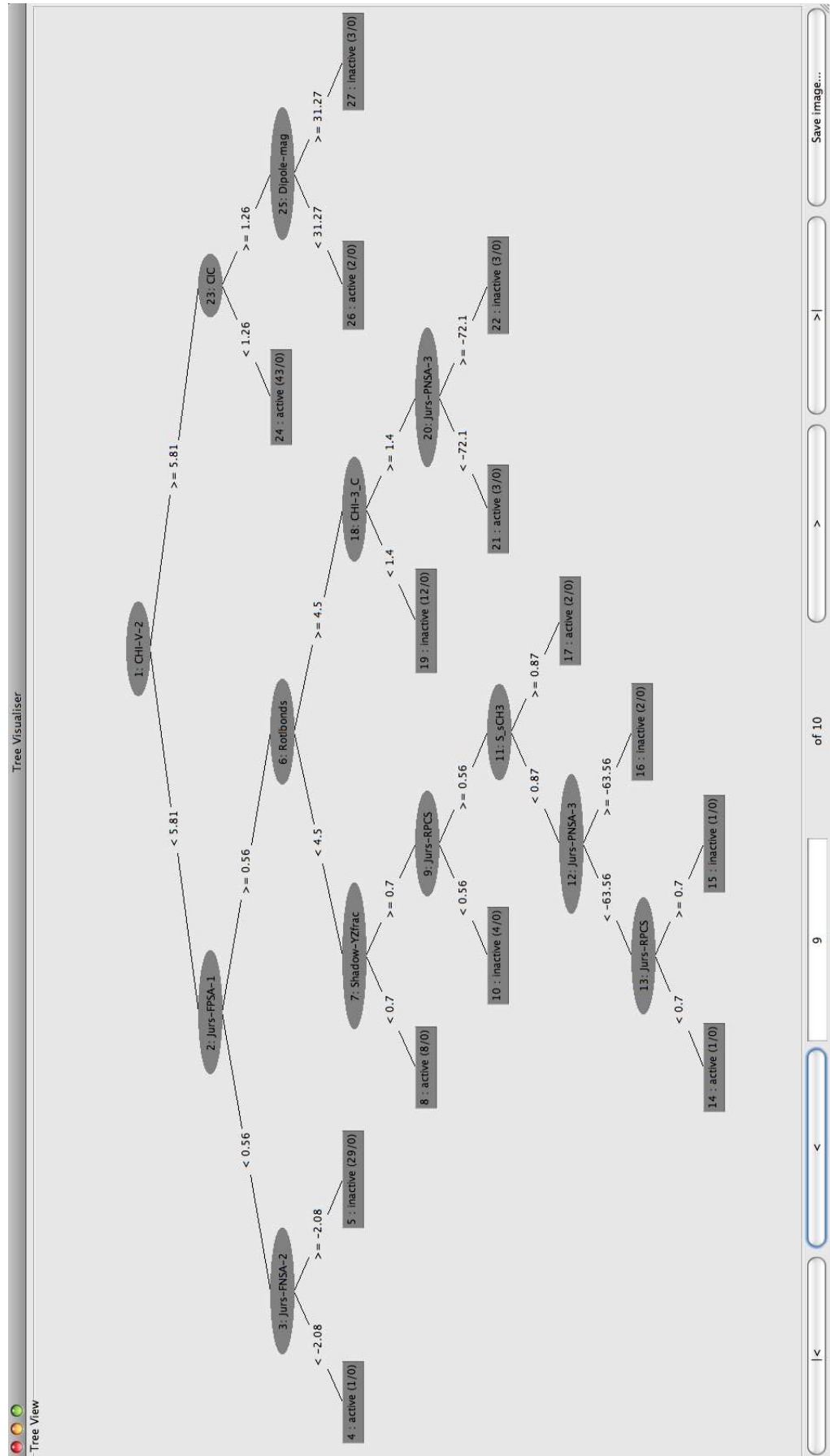


Figure 4.3: Weka tree visualiser with added navigation panel to view trees in a forest

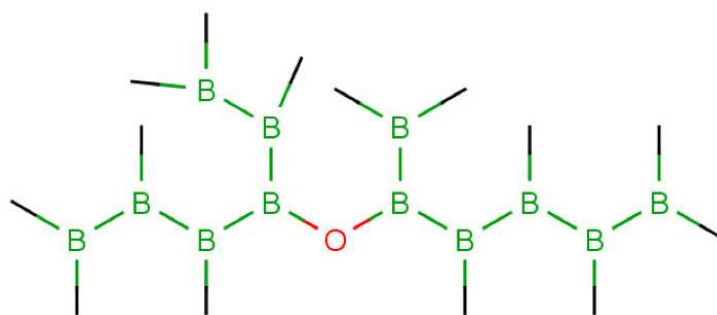


Figure 4.4: A tree encoded as SMILES. Oxygen (O) is the root node. This representation has no chemical meaning.

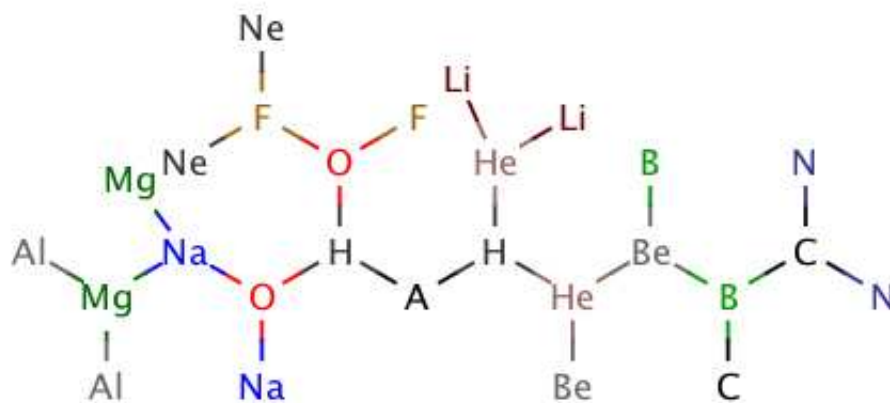


Figure 4.5: A tree encoded as SMARTS. The wildcard atom type (A) is the root node. This representation has no chemical meaning.

finding misclassified compounds FBI can produce a list a molecules that are incorrectly classified by  $x\%$  of trees.

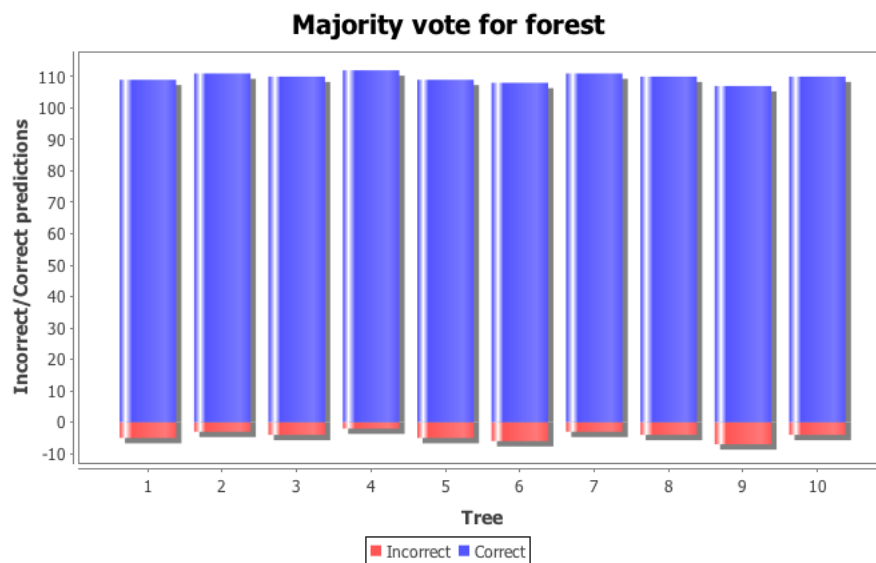


Figure 4.6: Summary of the majority vote for a forest built using the ACE data set

## 4.4 Conclusions

Random forest is a powerful technique. It is competitive with other classifiers on a variety of data sets, both small and large. While in theory it is an interpretable method, it is clear that as an ensemble this is no longer straightforward. The tools in FBI assist one in exploring the contents of the forest. Simple statistics are the first step to understanding the composition of the forest and individual trees. With the trees available in multiple formats, text, SMILES, SMARTS and the tree visualiser, further analysis is possible. The scoring matrix allow us to assess the performance of individual trees and compounds. The overall classification is the outcome of a majority vote. We can examine the details of the vote and highlight the border-line classifications. All the features within FBI add extra value and insight to our model and the data that created it. The use of interpretable descriptors enhances the benefit of highlighting descriptor importance.



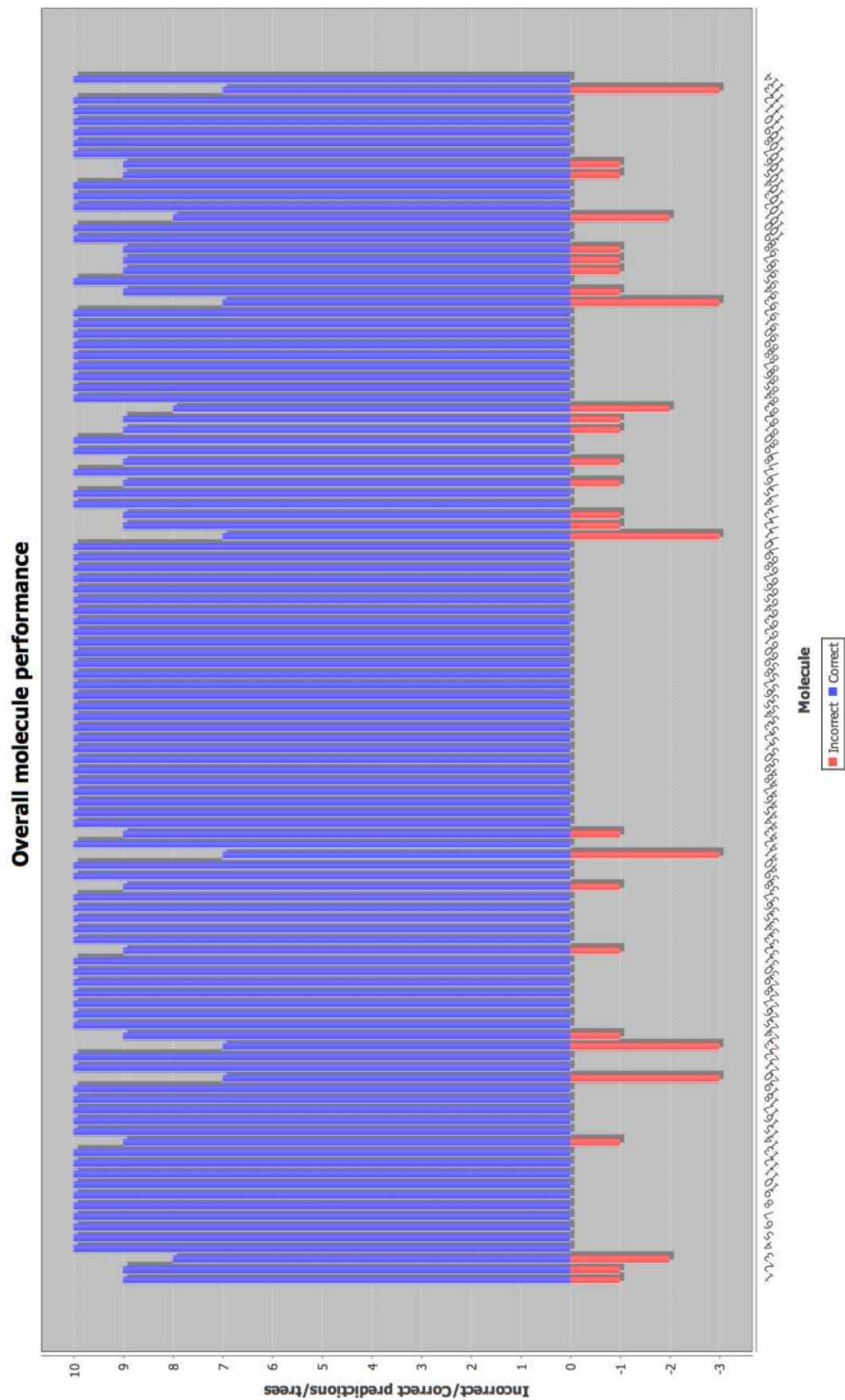


Figure 4.7: Molecule performance for a forest built using the ACE data set

The methods in FBI are by no means exhaustive, There are many avenues to explore further. However, it is a step forward from simple counts and is a foundation for further research. Access to the trees in multiple formats, particularly chemical, is perhaps the most important feature, as this allows analysis in other applications. We have shown how a forest does indeed contain useful insight into the model. The challenge is getting it to a readily understandable form. FBI allows the user to gain a greater understanding of the model. Black box methods offer no benefit post prediction, whereas model interpretation is invaluable in downstream analysis.

# Chapter 5

## TMACC: interpretable descriptors

### 5.1 Introduction

The descriptors used within a QSAR model are arguably the most important component. One can only assess and interpret a model based on the inputs - the chosen descriptors. While thousands are available through software packages and web services, how does one pick the best descriptors? It was originally thought the more the better. However, more descriptors increase the model generation time. Also extra descriptors add noise to the model and increase chance correlations, both of which detract from the aim of finding a chemical relationship. The increase in descriptor space is not proportional to information gain. This is known as the curse of dimensionality.<sup>143</sup> Therefore, a reasonable number of descriptors should be selected for a model. For interpretation, whole molecule descriptors are not as useful. For example, Lipinski-like rules<sup>46</sup> are useful filters, but not suitable for structure-based drug design. This is because the rules classify compounds on the likelihood of absorption. For compounds highlighted as poor for absorption they do not suggest which part of the molecule is

responsible. Descriptors that connect specific regions of the molecule with an impact of a property are better suited for structure-based drug design. Fragment-based descriptors are an example, as they represent a distinct chemical entity.

The TMACC descriptors (See Chapter 2) were developed in this research group.<sup>85</sup> They are highly predictive topological maximum cross correlation descriptors for use in QSAR. They are based on the autocorrelation method and encode several chemical properties. Their generation does not require 3D structures or alignments. The original work demonstrates qualitative agreement with more time consuming 3D and 4D QSAR methods. Here, we focus on their interpretive ability by developing tools to test their interpretive power. Used in conjunction with PLS, we explore the interpretation of an angiotensin converting enzyme (ACE) dataset<sup>88</sup> and performance in the Journal of Chemical Information and Modeling (JCIM) solubility challenge.<sup>28</sup>

Activity, lipophilicity and solubility are important properties in pharmaceutical compounds and, thus, common targets for SAR studies. Activity can be measured as  $IC_{50}$  or  $pK_i$ .  $IC_{50}$  is the half maximal inhibitory concentration. This value represents how much drug is required to inhibit the biological function of interest by half.  $K_i$  is the inhibition constant. This is the equilibrium constant that measures the reaction of the ligand and protein dissociation.  $pK_i$  is equal to  $-\log_{10} K_i$ . The octanol-water partition coefficient,  $\log P$ , is acknowledged to be relatively straightforward to predict compared to other properties, such as solubility.<sup>158</sup> The availability of experimental data is crucial in predicting properties. Solubility poses a challenge as producing reliable results is not as simple as other properties using either experimental or computational methods. This, in part, could explain why  $\log P$  is easier to predict than solubility. Models built are sensitive to the error of the data. Greater error yields less reliable predictions.

This is common for solubility.

Solubility is the ability of a substance (known as a solute) to dissolve in a solvent. Solubility is a dynamic equilibrium, which is maintained by a constant rate when no more solute can be added to the solvent as it is fully saturated. Due to the equilibrium nature of solubility it is pressure and temperature dependent. In order for substances to dissolve a suitable amount of energy is required to break the hydrogen bonds, otherwise they will be insoluble, for example water and oil. If energetically favourable the solute will form new hydrogen bonds with the solvent. Drugs need to be water soluble in order to be orally bioavailable, which is the preferred method of administration. Poor solubility can lead to late-stage failure, which is costly. High-throughput technologies have led to increased lipophilicity and molecular weight of screened compounds, which in turn reduces solubility.<sup>159</sup>

## 5.2 Methods

The ACE dataset from Sutherland and Weaver<sup>88</sup> is used retrospectively to validate the TMAcc interpretation. The JCIM solubility challenge data set was also used. Both are regression based tasks. The method follows that outlined in Chapter 2. The goal of the solubility challenge was predictive accuracy. Therefore, some modifications were implemented to improve this. Two alterations to standard TMAcc generation were explored. First, given solubility is the goal and TMAcc includes an explicit solubility term, we altered the standard 28 atomic property pairs. We investigated only using pairs with either one or both components being solubility, comparing to the standard TMAcc. As detailed in Chapter 2 TMAcc descriptors are composed of molecular property pairs based on electrostatics, molar refractivity,  $\log P$  and  $\log S$ . Given the nature of this study we have explored using only bias pairs to improve our solubility predictions. Second, we reduced the cutoff for the topological distance; by default it is the maximum possible.

Both of these alterations enhanced performance.

TMACC interpretation is made possible by using NCW, Nottingham Cheminformatics Workbench. Melville was responsible for the original TMACC descriptor code, written in Java. NCW extends this by adding a GUI to descriptor generation. The original code only generates descriptors. One would then have to build a model and analyse the interpretation by hand. Descriptor interpretation is not practical to be done manually. NCW, therefore, was written as part of this thesis work to automate it. Using NCW one can generate descriptors, build a model and analyse the interpretation in a graphical output, using Marvin.<sup>96</sup> NCW is now the default location for all in-house cheminformatics code. This includes 3D QSAR and similarity metrics<sup>12</sup>. A public version is available for download from our website <http://comp.chem.nottingham.ac.uk/download/ncw>. NCW works on all major operating systems and removes the need for extensive command-line usage. A GUI installer makes installation simple while documentation is available. In adherence with the open source Apache license,<sup>155</sup> full source is available, allowing others to extend and develop further if desired. The GUI installer and documentation is provided using the open source projects IzPack<sup>44</sup> and Sphinx<sup>156</sup>. This work would not have progressed as far without computational tools to enable analysis of the complete datasets, not just a few molecules.

With solubility being such a challenging property various methods have tried to predict it *in silico*. Four classes of descriptor are frequently used: melting point and  $\log P$ , atom or group contributions, physicochemical and quantum chemical descriptors, and topological indices.<sup>158</sup> The TMACC descriptors are atomic physicochemical based. Unfortunately the techniques employed by the 99 entrants of the solubility challenge were not disclosed, this would be very interesting to see. Palmer et al have used random forests to predict aqueous solubility.<sup>160</sup> They used random forest, PLS, SVM and

artificial neural networks for model generation using a variety of descriptors from the Molecule Operating Environment (MOE). A comparison of the models concluded that random forest was the most accurate. The random forest implementation in Weka is limited to categorical data only. Therefore, we did not use random forest in this exercise. Quantum chemical descriptors have been used as well as the high level of theory accessible could aid reliable prediction.<sup>161</sup> However, the required level of theory is excessively expensive. In this work the authors combined a lower level of theory with informatics in order to enable a reasonable compute time. Both of these studies have used different approaches to how the TMACC operates. They all have produced reasonable predictions for solubility.

**Solubility training data:** From the 101 molecules comprising the training data, we removed the following: aspirin, chlorprothixene, 5-fluorouracil, levofloxacin, L-proline, orbifloxacin, Pen G, phthalic acid, procainamide, sulindac, trichlormethiazide and chlorzaxazone. They were removed as they either had no intrinsic solubility value or were polymorphs. This left 89 training compounds and 32 in the test set. All structures are available in Appendix A. We converted the intrinsic solubility into log units. The predicted values are converted back to intrinsic solubility for the purposes of assessment.

**Solubility test data:** After the results were published<sup>162</sup>, it was evident that several compounds would pose a challenge for our method. 2-chloromandelic acid, 1R-2S-ephedrine, marbofloxacin and 1R-2R-pseudoephedrine have no intrinsic solubility. Diflunisal and trazodone are polymorphs. Although we made predictions, our method is not capable of predicting these types of compounds.

## 5.3 Results and discussion

### 5.3.1 Angiotensin converting enzyme

*This work was carried out by Benson Spowage, an undergraduate project student under my supervision, and it was only made possible by the availability of the NCW software.<sup>163</sup> Figures 5.1, 5.3 and 5.4 were drawn by Benson Spowage.*

Angiotensin converting enzyme, ACE, is part of the body's system to regular the volume of blood and arterial vasoconstriction. Specifically it catalyses the conversion of angiotensin I to angiotensin II by removal of terminal amino acids which led to vasoconstriction. In addition, it degrades bradykinin, a vasodilator. The net effect of these two actions is to constrict arterial vessels. ACE inhibition is therefore a target for pharmaceutical research as it can treat a variety of conditions such as hypertension and type II diabetes. ACE has been studied extensively and therefore makes it very suitable for retrospective analysis.

To validate the interpretive ability of the TMACC descriptors the ACE data set was used. ACE induces hypertension, which makes it a popular pharmaceutical target.<sup>164</sup> The original TMACC work only briefly covered interpretation. With NCW it is now possible to assess a whole data set with greater ease. Melville's implementation created leave-one-out (LOO) cross-validation PLS models. Melville wrote a PLS algorithm before it was added to Weka. The  $q^2$  for this model is 0.70, which matches the original values reported.<sup>85</sup> The data set consists of 114 molecules and was originally constructed for a 3D QSAR study.<sup>135</sup> TMACC descriptors do not require 3D coordinates, only the 2D connection tables. Activity is determined by  $IC_{50}$  values.  $IC_{50}$  is the half maximal inhibitory concentration, the amount of a compound required to inhibit a given biological response by 50%. Several studies determined  $IC_{50}$  using the substrate hippuryl-histidyl-leucine



(HHL).<sup>165,166</sup> One method is to measure the rate of hippuric acid production from HHL catalysed by ACE.<sup>167</sup> However, it was subsequently discovered that HHL is a C-terminal domain specific substrate of ACE.<sup>168,169</sup> This data set consequently is likely to reflect that of C-terminal domain specific ACE inhibition, rather than general inhibition.

To demonstrate the TMACC interpretation is highlighting pharmacologically interesting features, a retrospective analysis must be performed. From the literature several, potentially important features are extracted and shown in Figure 5.1. This data set contains three commercial compounds which are all licensed ACE inhibitors, captopril, enalaprilat and lisinopril. Applying these features to the known commercial inhibitors in Figure 5.2 a corroboration can be seen.

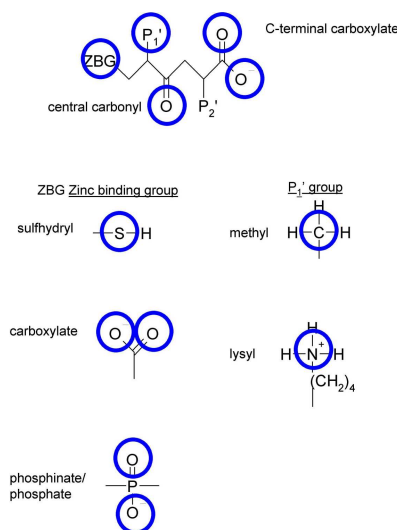
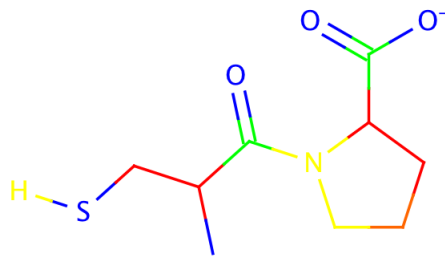
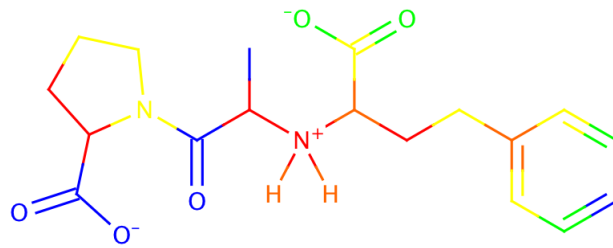


Figure 5.1: ACE inhibitor features investigated. Position of features shown in 2D relation to one another. Blue circles surround atoms studied for activity.

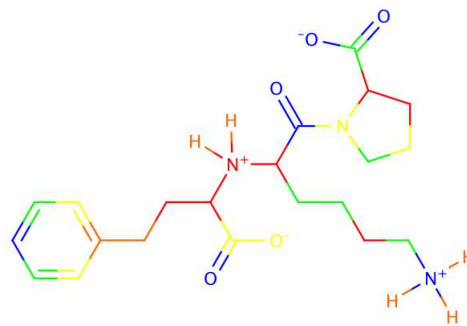
An essential feature is the presence of a zinc coordinating group. The catalytic zinc iron is coordinated by three highly conserved residues present in both domains. The functional role of the zinc ion in the active site has led to the development of peptide based inhibitors, such as enalaprilat. The importance of zinc binding has been shown in the crystal structure<sup>170</sup> and SAR studies.<sup>171</sup> Three zinc binding groups present in the data set are inves-



A



B



C

Figure 5.2: TMACC interpretation of ACE inhibitors. A. Captopril. B. Enalaprilat. C. Lisinopril.

tigated: sulfhydryl, carboxylate and phosphinate/phosphate. All sulfhydryl sulfur atoms were identified as positive contributors towards activity. Analysis of phosphinate zinc binding groups showed all phosphorus atoms were labelled as positive for activity. However, phosphinyl oxygen atoms were mostly shown as negative for activity. In contrast, the interpretation most frequently identified both carboxylate zinc binding group oxygen atoms to be positive for activity, although the results do not fully capture the correlation between the type of zinc-ligand and inhibitor activity observed in structure-activity studies, (phosphinate carboxylate sulfhydryl).<sup>90</sup> Perhaps the negative activity attributed to the phosphinate oxygen atoms reflects its weak zinc-binding ability in comparison to the other zinc binding groups.

The central carbonyl group is a feature found in most ACE inhibitors. It forms two hydrogen bonds within both domains of ACE.<sup>172,173</sup> Docking studies suggest this interaction is frequently present in ACE-inhibitor binding and it has been identified in many ACE-inhibitor crystal structure complexes (Figure 5.3).<sup>174</sup> Mutation of <sup>513</sup>His to alanine causes a 120,000-fold decrease in the binding of lisinopril to the C-domain of somatic ACE (sACE).<sup>175</sup> This suggests the interaction of the conserved histidine residues with the carbonyl group of an inhibitor is important for ACE inhibition. The TMACC interpretation identified the central carbonyl as favourably contributing towards the activity (Table 5.1). The high frequency of positive activity shown for this feature by the TMACC interpretation is consistent with the aforementioned literature. The features highlighted in Table 5.1 have resonance structures, for example the first three C-terminal features. The TMACC descriptors do not have any knowledge of resonance structures. However, all nonpolar hydrogens are treated implicitly, so the interpretation diagrams do show a particular contributing structure, which is what the descriptors values are based on. It is not unusual for cheminformatics techniques to ignore resonance.

ACE inhibitor feature	Activity		
	Negative	Neutral	Positive
C-terminal carboxylate carbonyl	5	0	105
C-terminal carboxylate hydroxyl	0	5	105
Central carbonyl	13	3	96
Zinc binding carboxylate - carbonyl	3	3	22
Zinc binding carboxylate - hydroxyl	0	4	24
Zinc binding sulfhydryl sulfur	0	0	33
Zinc binding phosphinate phosphorus	0	0	22
Zinc binding carbonyl	20	0	2
Zinc binding hydroxyl	20	0	2
P1' methyl	4	2	27
P1' lysyl nitrogen	0	0	20

Table 5.1: Frequency of activity of ACE inhibitor features as determined by the TMACC interpretation.

The crystal structures of testicular ACE (tACE) in complex with various inhibitors (Figure 5.3) show the intermolecular interactions responsible for ACE inhibition in tACE and correspondingly the C-terminal domain of sACE.<sup>171,172,174</sup> In contrast to most zinc protease inhibitors, which primarily rely on the strength of their zinc binding groups for activity, domain-specific ACE inhibitors utilize weak zinc binding groups and exploit both primed and unprimed sides of the active site in order to mimic peptide substrates, thereby achieving domain selective inhibition.<sup>176</sup> Domain-specific inhibition of ACE is important, as each domain possesses individual functions.<sup>177</sup> This discovery has developed the number of applications of ACE inhibitors, extending from treating hypertension to protecting stem cells during chemotherapy.<sup>178</sup> A recent study has also suggested ACE may be involved in many physiological processes other than blood pressure regulation.<sup>179</sup>

The two domains of sACE contain many conserved residues, which are vital for substrate and inhibitor binding (Table 5.2). The identification of conserved residues within ACE and their role in inhibitor binding has highlighted several important features required for ACE inhibition, providing a

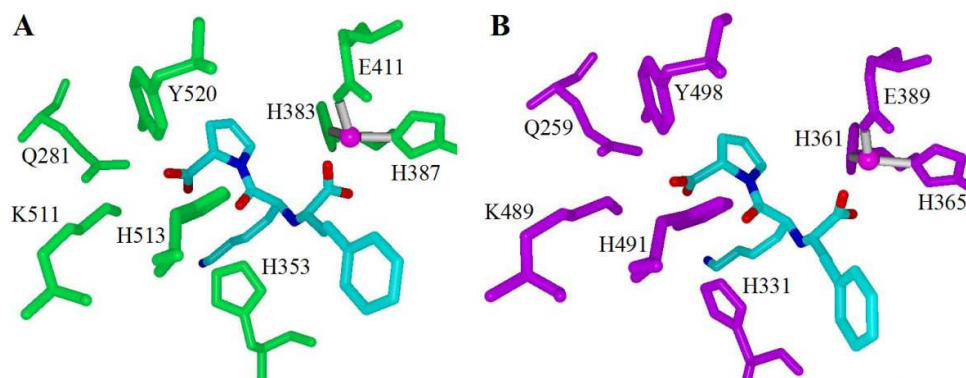


Figure 5.3: Conserved ACE residues that interact with lisinopril. A) tACE active site (green).<sup>172</sup> B) The N-domain active site of sACE (purple).<sup>173</sup> Zinc ion shown in magenta; atoms are coloured as follows: red for oxygen, blue for nitrogen, cyan for carbon and grey for hydrogen.

rationale for the structure-activity relationship of ACE inhibitors.

A C-terminal carboxylate is found in many ACE inhibitors. This feature interacts with several conserved residues in both domains of sACE, hydrogen bonding with tyrosine and glutamine residues, and also forms an electrostatic interaction with a lysine residue (Figure 5.3). Both C-terminal carboxylate oxygen atoms were identified as positive by the TMACC interpretation (Table 5.1).

Despite the high level of conserved residues present in both domains of sACE, variation between the domains confers different substrate and inhibitor preferences. The presence of hydrophobic residues <sup>379</sup>Val and <sup>380</sup>Val in the S1' sub-site of the C-domain of sACE provides hydrophobic interactions between the sub-site and the P1' residue of inhibitor molecules, such as the P1' methyl group of captopril and enalaprilat.<sup>173</sup> The corresponding residues found in the N-terminal domain, <sup>357</sup>Ser and <sup>358</sup>Thr, provide a polar environment and, therefore, do not form similar hydrophobic interactions with the P1' residue of inhibitors.<sup>173,174</sup> In the C-terminal domain the lysyl chain of lisinopril extends into the S1' sub-site and forms an electrostatic interaction with <sup>162</sup>Glu and a water-mediated interaction with <sup>377</sup>Asp.<sup>172</sup> However, in the N-terminal domain the S1' sub-site makes fewer contacts with the lysyl chain of lisinopril (Figure 5.4). For example, <sup>162</sup>Glu

Functional interaction	C-domain residue	N-domain residue
Zinc-binding	<sup>383</sup> His	<sup>361</sup> His
Zinc-binding	<sup>387</sup> His	<sup>365</sup> His
Zinc-binding	<sup>411</sup> Glu	<sup>389</sup> Glu
Inhibitor carbonyl hydrogen bonding	<sup>513</sup> His	<sup>491</sup> His
Inhibitor carbonyl hydrogen bonding	<sup>353</sup> His	<sup>331</sup> His
Inhibitor carboxy terminal carboxylic ionic bonding	<sup>511</sup> Lys	<sup>489</sup> Lys
Inhibitor carboxy terminal carboxylic hydrogen bonding	<sup>281</sup> Gln	<sup>259</sup> Gln
Inhibitor carboxy terminal carboxylic hydrogen bonding	<sup>520</sup> Tyr	<sup>498</sup> Tyr

Table 5.2: Conserved ACE residues important for inhibitor interactions. Table formulated using information from refs<sup>170,173,180</sup>

(C-domain) is replaced by <sup>140</sup>Asp (N-domain), and due to the larger distance between the lysyl chain and this residue, no electrostatic interaction is observed at this location in the N-domain.<sup>173</sup> Additionally, <sup>377</sup>Asp (C-domain) is replaced by <sup>355</sup>Gln (N-domain), thereby abolishing the water-mediated interaction shown between the lysyl residue of lisinopril and the C-domain.<sup>173</sup> This evidence suggests that methyl and lysyl groups located in the P1' position of ACE inhibitors can form favourable interactions with the S1' sub-site of the C-terminal domain of sACE.

Interestingly, the TMACC interpretation identified all inhibitor P1' lysyl nitrogen atoms as favourably contributing towards activity (Table 5.1). The interpretation also identified inhibitor P1' methyl groups as positive for activity. Thus, the TMACC interpretation identified P1' groups important for C-domain specific ACE inhibition, as illustrated in Figure 5.2. This C-domain specific bias in the data set, reflected by the TMACC interpretation, has not been shown in previous QSAR investigations using this data set.<sup>88,135</sup>

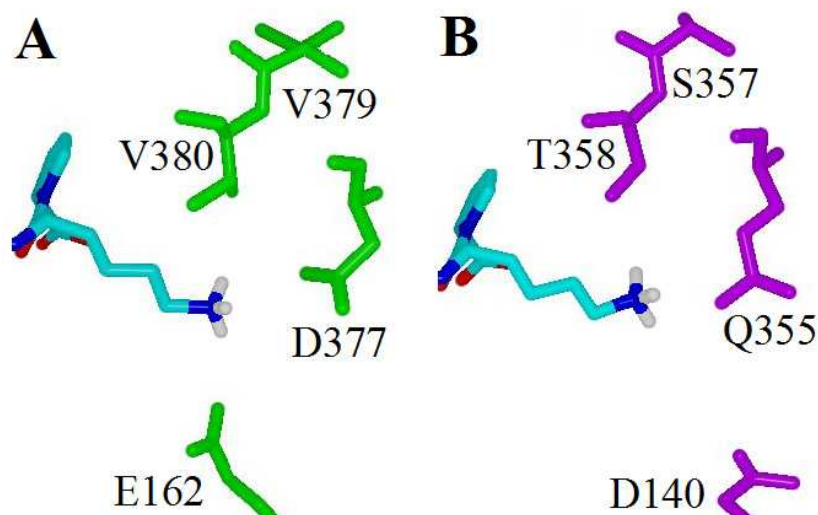


Figure 5.4: Comparison of the S1' sub-site residues which bind the lysyl group of lisinopril. A) tACE (green)<sup>172</sup> and B) the N-terminal domain of sACE (purple).<sup>173</sup>

### 5.3.2 Solubility challenge

The measurable component of the solubility challenge relates to predictive accuracy on an external test set. Therefore, the steps we have taken aim to optimise this. We used a selection of methods from Weka to analyse our training set, based on the ten-fold cross-validated correlation coefficient,  $q^2$ . From Weka we tried support vector regression (SVM and SMO variants), PLS and M5P.<sup>181,182</sup> M5P is the Weka implementation of the M5 system which constructs tree-based piecewise linear models. We report on the 89 training data using standard TMAcc descriptors, see Table 5.3. We also reduce the number of descriptors to those that include either one or both components relating to solubility. At this stage we use the default parameters for all techniques from Weka. By imposing a criterion on the TMAcc atomic pairs, the number of descriptors drops dramatically, see Table 5.4.

M5P does not perform as well as the support vector regression or PLS. In addition only using descriptors which both contain a solubility component has a detrimental effect on the overall prediction. Both of these avenues are not explored further. Both support vector variants have similar performances. However, SVM is dropped in favour of SMO. Now with two

TMACC options	Solubility ( $\mu\text{g}/\text{mL}$ )	$q^2$			
		SVM	SVR	PLS	M5P
Default	S	0.17	0.17	0.13	0.00
	$\log S$	0.65	0.65	0.62	0.40
One $S$	$S$	0.27	0.27	0.09	0.11
	$\log S$	0.66	0.66	0.67	0.43
Both $S$	$S$	0.13	0.13	0.07	0.05
	$\log S$	0.44	0.44	0.25	0.23

Table 5.3: Weka classifiers with multiple TMACC variants. SVM: Support vector regression using SMO, weka package SMOREg. SVR: Support vector regression using the weka package SVMreg. PLS: Partial Least Squares. M5P: Tree-based model based on the M5 system.

TMACC variant	Number of attributes
Default	617
One $\log S$ component	287
Both $\log S$ components	67

Table 5.4: Number of attributes in different variants of TMACC. Number of  $\log S$  components refers to those included.

Descriptor	SVM		SMO		PLS	
	MAE	RMSE	MAE	RMSE	MAE	RMSE
Default	0.81	0.99	0.81	0.99	0.92	1.12
One $\log S$	0.76	0.99	0.76	0.99	0.80	1.05

Table 5.5: Errors for the cross-validation for the three techniques.



Cutoff	SMO			PLS		
	$q^2$	MAE	RMSE	$q^2$	MAE	RMSE
None	0.66	0.76	0.99	0.67	0.80	1.05
19	0.66	0.76	0.99	0.68	0.80	1.05
17	0.66	0.75	0.98	0.67	0.81	1.08
13	0.67	0.75	0.98	0.66	0.83	1.08
11	0.63	0.76	1.02	0.46	1.08	1.43
6	0.30	0.96	1.24	0.12	1.83	2.39

Table 5.6: Final two techniques with cutoff of maximum topological distance

Property	Default	Minimum	Maximum	Step
$C$	1	1	20	1
$\gamma$	0.01	$1.0 \times 10^{-5}$	$1.0 \times 10^{+2}$	1
$\epsilon$	0.001	$1.0 \times 10^{-5}$	$1.0 \times 10^{+2}$	1

Table 5.7: GridSearch for  $C$ ,  $\gamma$  and  $\epsilon$ . Step increment applies only to the exponent, except  $C$ .

techniques we look at optimising them and reducing the maximum distance of the topological bond distance. From this point, we only use the one  $\log S$  component TMAcc variant, as it outperforms the default TMAcc. The  $q^2$  value does not alter. However, the error values decrease before the  $q^2$  suddenly drops. PLS has consistently higher errors and similar  $q^2$  values. Therefore, the SMO variant of support vector regression is chosen as the best technique. Further modification of the TMAcc is achieved by applying a cutoff to the topological distance component. Table 5.6 shows various cutoff results, from which 13 is reasonable and used from now on.

Finally, further optimisation on support vector regression can be performed. The RBF kernel is known to perform well. Next we tune three parameters,  $C$ ,  $\gamma$  and  $\epsilon$ , which represent the complexity constant, the RBF width and the round-off error, respectively. A grid search is possible within Weka for two components, optimising for  $q^2$ .  $C$  and  $\gamma$  are optimised first, then  $C$  and  $\epsilon$ .

The GridSearch reveals for SMO with an RBF kernel using the 13 cutoff TMAcc that  $C$  should be 1, with  $\gamma = 0.001$  and  $\epsilon = 0.1$  (See Table 5.7).

These are the parameters used in SMO for our predictions, detailed in Table 5.8. Figure 5.5 shows the observed versus predicted data for the training data. Figure 5.6 shows the cross-validated results for the same data. All test predictions are within the same range as the training observations. Figures 5.7, 5.8 and 5.9 display the observed versus predicted solubilities for the three compound sets 32, 28 and 24 respectively. Test compounds which were classed as “Too Soluble to Measure” are not included in these plots. The extra lines on this graph show the 0.5 log unit zone where correct predictions are counted.

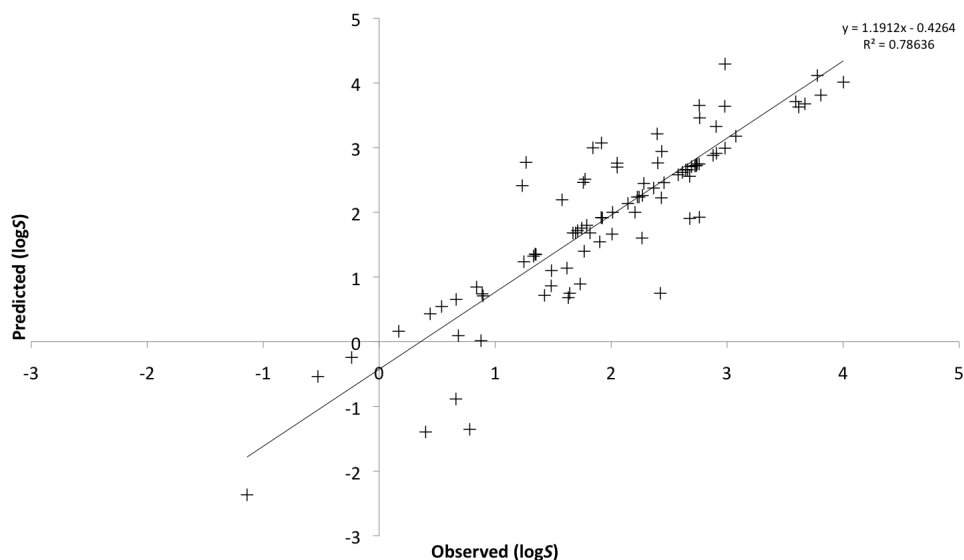


Figure 5.5: Predicted versus observed solubility for the training data ( $r^2 = 0.79$ ).

Using the interpretation from the TMAcc some known trends can be seen. Halogens reduce solubility and increase lipophilicity as seen in 5.11. The colour scheme is depicted in Figure 5.10. Cyclopropane is reactive and hence good for solubility as demonstrated by 5.12.<sup>29</sup>

In terms of the solubility challenge we generally did not score highly against the other entries. The scoring criteria were as follows, (a) Percentage correct of full test set (32 compounds). (b) Percentage correct from soluble compounds (28 compounds). (c)  $r^2$  of 28 compounds. (d) Per-

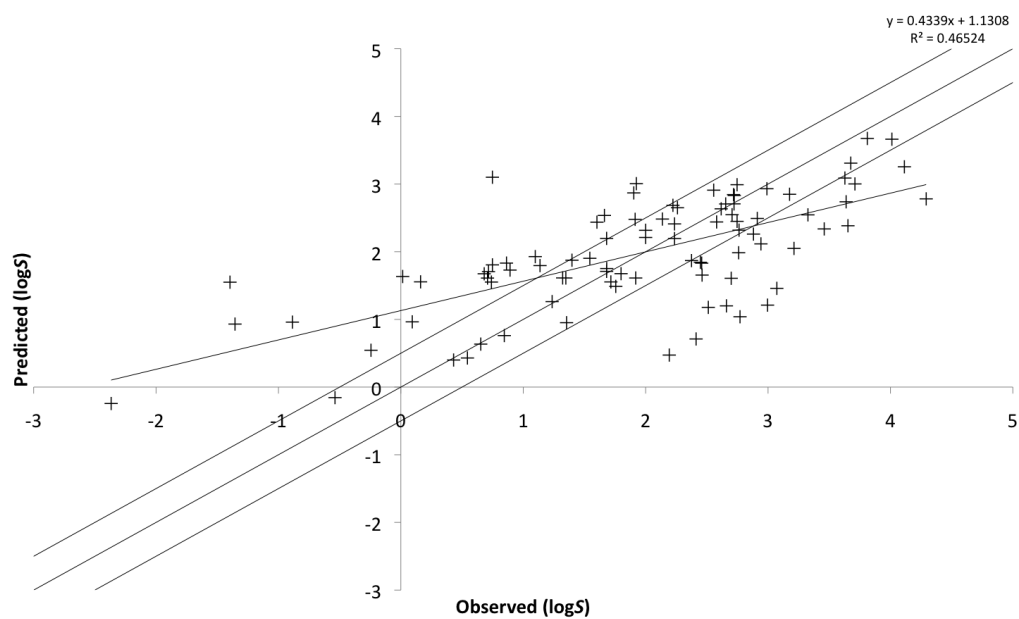


Figure 5.6: Predicted versus observed solubility for the cross-validated data. For the solubility challenge the predictions must fall with  $\pm 0.5$  log units to be considered correct.

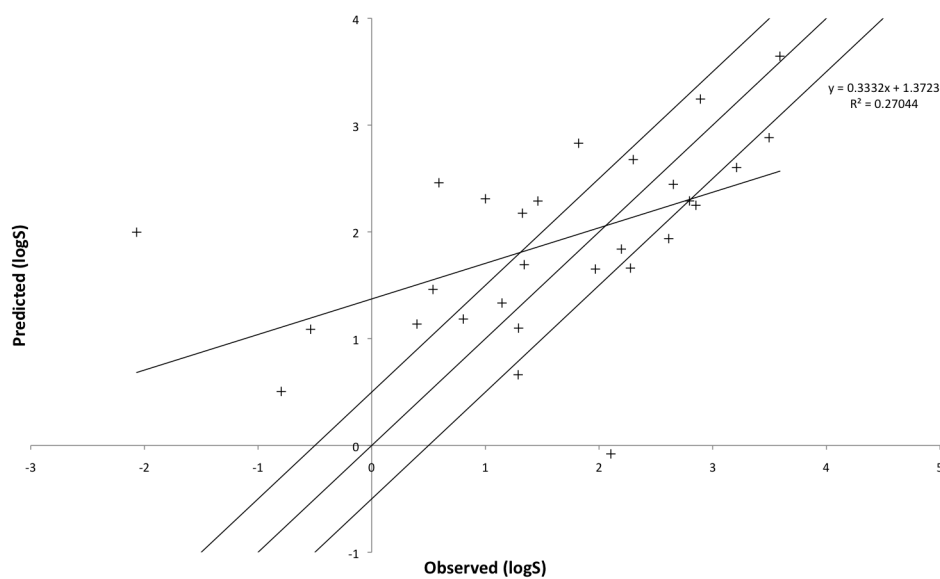


Figure 5.7: Predicted versus observed solubility for the 32 molecule set.

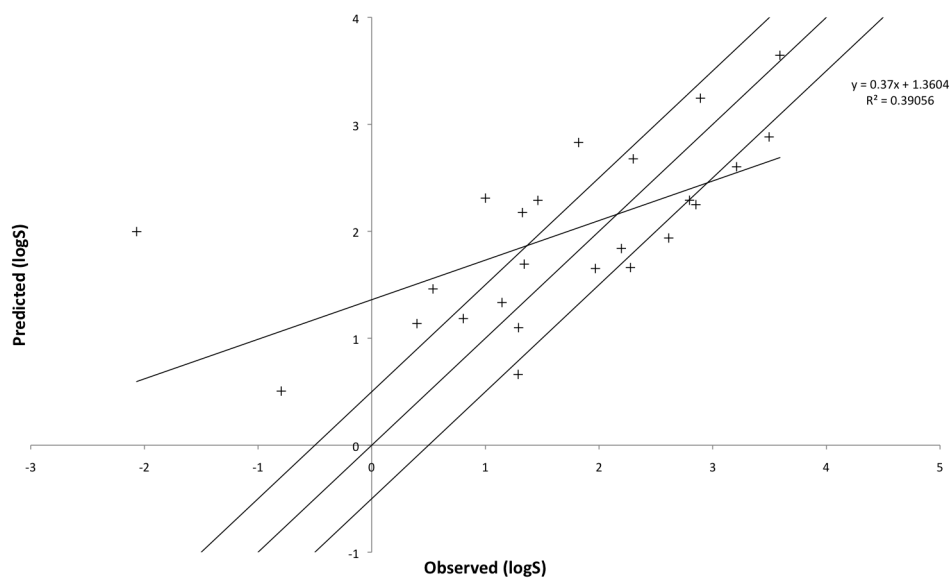


Figure 5.8: Predicted versus observed solubility for the 28 molecule set.

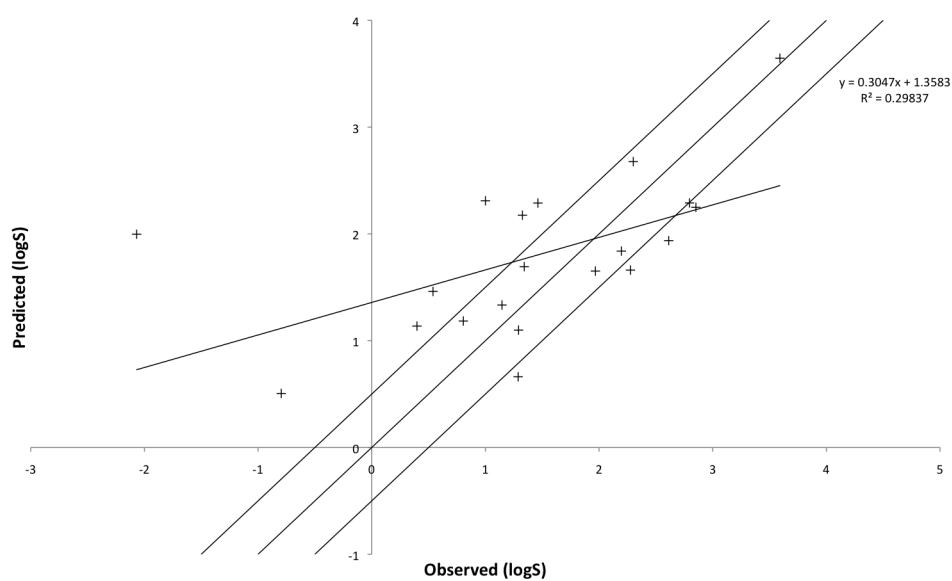


Figure 5.9: Predicted versus observed solubility for the 24 molecule set.

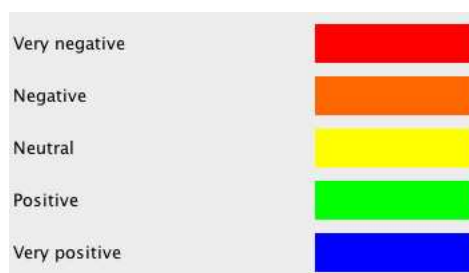
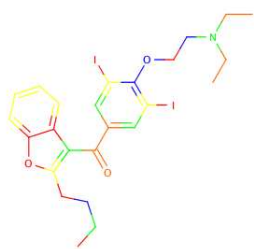


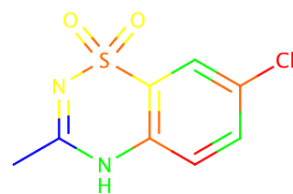
Figure 5.10: TMACC colour scheme

Compound	Observed solubility ( $\mu\text{g}/\text{mL}$ )	Predicted solubility ( $\mu\text{g}/\text{mL}$ )
1 acebutolol	711	177
2 amoxicillin	3900	4416
3 bendroflumethiazide	780	150
4 benzocaine	780	1754
5 benzthiazide	6.4	15
6 2-chloromandelic acid	too soluble to measure	869
7 clozapine	189	46
8 dibucaine	14	22
9 diethylstilbestrol	10	204
10 diflunisal	26, 7.6, 0.93, 0.29	12
11 dipyridamole	3.46	29
12 1R-2S-ephedrine	too soluble to measure	288
13 folic acid	2.5	14
14 furosemide	19.6	13
15 hydrochlorothiazide	625	195
16 imipramine	22	49
17 indomethacin	410	86
18 ketoprofen	157	69
19 lidocaine	3130	762
20 marboloxacin	too soluble to measure	1807
21 meclofenamic acid	0.16	3
22 naphthoic acid	28.96	195
23 1R-2R-psuedophedrine	too soluble to measure	32
24 probenecid	3.9	288
25 pyrimethamine	19.4	5
26 salicylic acid	620	401
27 sulfamerazine	200	475
28 sulfamethizole	450	279
29 terfenadine	0.00856	99
30 thiabendazole	66	676
31 tolbutamide	93	45
32 trazodone	460, 127	0.834

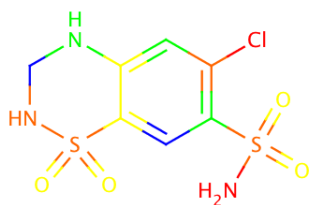
Table 5.8: Test set compounds and their observed and predicted  $\log S$  values. Compound 10 has multiple entries to represent the four polymorphs it forms.



A



B

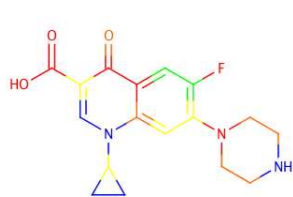


C



D

Figure 5.11: **A** amiodarone, **B** diazoxide, **C** hydrochlorothiazide & **D** pyrimethamine. The present of a halogen has increased lipophilicity and therefore reduced solubility.



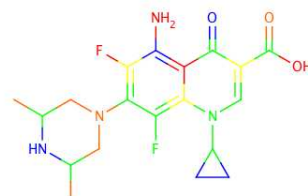
A



B



C



D

Figure 5.12: **A** ciprofloxacin, **B** danofloxacin, **C** enrofloxacin & **D** sparfloxacin. Cyclopropane is reactive and good for solubility.

Molecules	32	28		24	
Criteria	a (%)	b (%)	c ( $r^2$ )	d (%)	e ( $r^2$ )
$S$	60	62	3	62	3
$\log S$	57	63	74	63	51

Table 5.9: Results of solubility challenge, ranks against the other 99 entrants.

centage correct from soluble compounds minus the four largest outliers (24 compounds). (e)  $r^2$  of 24 compounds. This was done for both  $S$  and  $\log S$  values. For  $S$  to be considered correct it had to be within  $\pm 10\%$  of the actual value. For  $\log S \pm 0.5 \log S$  units of the observed value was required to be considered correct. Our models were generated using  $\log S$  values; using  $S$  produced poor  $r^2$  values. The  $\log S$  values were converted to  $S$  for the purpose of scoring. However, our  $r^2$  values on the 28 and 24 compound subsets were excellent compared to other entrants. Our excellent  $r^2$  values are somewhat surprising given zero compounds were correct by the criterion set. This is an artefact of the arbitrary nature of a clear cut criterion. Our ranks for all five measures against the other 99 entrants are in Table 5.9. Our  $\log S$   $r^2$  values were not quite as impressive. The  $r^2$  for the 28 molecule set is worse than the 24 molecule set. This is because the 4 outliers which were removed were on average 2.3 log units incorrect in their predictions. The large error associated with these 4 outliers reduced the overall  $r^2$ . Our results are shown in Table 5.10 along with the best and median performance from all other entrants. The  $\log S$  criterion of  $\pm 0.5 \log S$  units is unduly harsh. Assuming a criterion of  $\pm 1.0 \log S$  our results fare better. The percentage of correct predictions for the 32 molecule set is 71.9% instead of 37.5%. For the 28 molecule set, it is 75.0% instead of 35.7%. For the 24 molecule set 75.0% instead of 41.7%. These results demonstrate that for  $\log S$  our method is relatively successful, against this relaxed but not unrealistic error range.



Molecules		32	28		24	
Criteria		a (%)	b (%)	c ( $r^2$ )	d (%)	e ( $r^2$ )
TMACC	$S$	6.3	0.0	0.625	0.0	0.941
	$\log S$	37.5	35.7	0.269	41.7	0.621
Best	$S$	21.9	17.9	0.642	20.8	0.987
	$\log S$	62.5	60.7	0.650	70.8	0.835
Median	$S$	9.4	3.6	0.082	4.2	0.494
	$\log S$	43.8	42.9	0.360	50.0	0.622

Table 5.10: Results of solubility challenge, against performance markers. The best and median results are also shown for comparison.

## 5.4 Conclusions

In previous work, the TMACC descriptors have proved to be competitive against 2D and 3D methods, but the interpretative ability of these descriptors was only briefly touched upon. In order to make interpretation readily available, new computational tools were required to carry out additional calculations on the PLS model and FGRAM output. NCW encompasses the original software and provides the extra functionality to build the PLS model. The PLS model and FGRAM output are then used to calculate atomic contributions for each atom in the data set. The results are binned into five colour-coded categories, which are depicted using the ChemAxon Marvin suite. NCW allows models to be generated and interpreted quickly. The results shown here illustrate that the interpretation produced by TMACC is indeed chemically valid. There is scope for further automation of the interpretation, particularly automatic recognition of features of importance. Currently, the user is required to identify relevant information.

Analyses of the TMACC QSARs modelled for the ACE data set have shown that the TMACC interpretation can identify distinctive features of a structure-activity relationship. The TMACC interpretation provided a clear and precise representation of the activity of specific groups. Amalgamation of the atomic activity values determined for such groups within a data

set, showed strong correlation with experimental evidence, which shows the TMAcc interpretation can produce models which accurately depict the features of a structure-activity relationship.

Overall, the TMAcc interpretation of the ACE inhibitor structure-activity relationship highlighted important features for C-domain selective ACE inhibition. The TMAcc interpretation provided a consistent representation of the structure-activity relationship present in the ACE data set, albeit limited by the size of the data set. To obtain a more detailed analysis of components important or detrimental to the ACE inhibitor structure-activity relationship, it would be necessary to investigate a data set which represents a comprehensive range of functional groups and structural components. Investigation of the activity of features important for C-terminal domain selective inhibition in comparison to features important for N-terminal domain selective inhibition would provide further insight into the interpretive ability of the TMAcc descriptors.

An inherent weakness, due to the 2D nature of the TMAcc descriptors, is insensitivity to chirality. However, the use of chirality descriptors derived from topological data may provide a solution to this limitation and may also improve the predictive ability of the QSAR models.<sup>183</sup> Investigation of alternative or additional atomic properties used in the TMAcc descriptors would provide an insight into the properties which contribute towards activity. The effect of implementing more sophisticated partial charge calculations would be interesting, as a recent study has suggested that the method used for partial charge calculations can affect QSAR predictive accuracy.<sup>184</sup> Investigation of a wider range of data sets would provide further validation of the utility of the TMAcc interpretation.

# Chapter 6

## Conclusions

There has never been a more pressing time in drug discovery to reduce development time and cost. In silico techniques offer inexpensive methods to assist in early stage development. QSAR is just one of many techniques available from cheminformatics. This thesis has covered various aspects of QSAR from model choice and optimisation, to descriptors and appropriate statistics.

The so called patent cliff affects all major pharmaceutical companies. Virtually all top ten billion-dollar blockbusters come off patent in the next five years. There has been a reduction in new blockbusters so there are none to replace them. When Lipitor<sup>185</sup> goes off patent for Pfizer, it will leave a huge revenue gap. Pharma realises it is no longer the norm to have a few billion dollar blockbusters. It is not the lack of candidate drugs, but a lack of successful candidate drugs. The cost of clinical trials is prohibitive, especially if a compound has borderline pre-clinical results. At the same time registration has become more rigorous, following high profiles cases of negative, unknown, side effects, e.g. Vioxx.<sup>186</sup> In addition, only new in class or better than existing in class drugs will get registered. While a company may improve treatment and efficacy of a patented compound, if it does not outperform the market leader registration will not be granted. Competition from generics is now fierce. They do not wait for patents to expire and are

increasingly challenging existing patents, in some cases successfully. In turn this makes the original patent even more important, as it is required to be watertight and withstand challenge for the duration of the patent.

Pharma already employs a variety of methods to extend patents. Clinical trials on children earn an extra six months. Revising the dosage or combination therapy requires a new application, thus extending the total time of protection. To combat the competition from generics, once drugs go off-patent some companies have either bought into or set up their own generics divisions. Following the global trend of outsourcing some R&D have moved to Asia: India and China, primarily. The lure is no different to any other industry, cheaper labour than western counterparts. However, it has not been smooth. For example Novartis has had trouble securing intellectual property over their research. Regardless of the approach to ensure more compounds reach clinical trials and are successfully registered, the starting point of lead generation is perhaps the most important step. This is where cheminformatics comes in.

Throughout this thesis core components of QSAR have been covered. Every choice made before building the model is important as it affects the potential outcome and therefore usefulness of it. Data mining offers a plethora of techniques to build a model upon, each with advantages and disadvantages. One needs to generate a reasonably sized data set. Data is not always readily available. If the data set is to be split into train, hold and test sets, this requires careful handling to ensure the sets are equally distributed. Having a test set containing the most obscure or outliers is not sensible. Descriptors will need to be generated. However, there are thousands to pick from again, with advantages and disadvantages. Actually building the model will likely take less time than the data preparation and associated decision process. Ideally one will want to extract some insight from the model - this is dependent on the model and the descriptors picked.

How do we move this process forward? The machine learning community produces new techniques and it is not long before cheminformatics and bioinformatics practitioners try them. Popular methods will get attention and will be modified to suit the task at hand. Examples include adding interpretability to neural networks and creating chemical kernels for SVM. Another avenue becoming increasingly popular is the idea of AutoQSAR.<sup>187</sup> Once one has built a model it is a static snapshot. In pharma, particularly, more data will become available and the model could and arguably should be updated. AutoQSAR has dynamic models which are constantly refreshed using new data and even descriptors. Such a system makes QSAR available to non-experts as they do not need to build the model. However, rigorous validation is necessary as it is all too easy to generate hundreds of poor models that are assumed correct by non-experts.

We have attempted to advance QSAR in several ways. Firstly, we have explored the interpretive abilities of the TMAcc descriptors beyond the original work. Secondly, we have advocated the use of appropriate statistics for comparison of multiple classifiers and data sets. Most methodology articles include some sort of comparison to highlight reported advances. Appropriate statistics are vital for assessing a given advance as “statistically significant”. Thirdly, we have extended the interpretive ability of the random forest. Random forest is a competitive technique compared to the benchmark SVM. However, relatively little work has been carried out interpreting the model. We have addressed this by introducing FBI to aid modellers analyse their forests. The TMAcc descriptors offer competitive and interpretive descriptors for QSAR. Thanks to the development of NCW their generation, model building and analysis is far more practical than previously. NCW provides a complete work flow solution and importantly aids in interpreting one’s results. Throughout our comparison work it became clear that the ability of learning classifiers is high and competitive. Inter-

estingly they use completely different approaches to achieve high predictive accuracies. It transpires comparing multiple classifiers with multiple data sets is not an appropriate use of the  $t$ -test. More relevant techniques are required and we highlight several more applicable statistic tests.

We had a specific interest in trees due to their familiarity. Our advances in forest interpretation are numerous over the previous simple counts. The various approaches mentioned all can aid one in gaining insight to a non black box method that is no longer simple to understand.

QSAR has developed a lot, since its original conception by Hansch. It has had numerous advances and successful applications. This work aims to continue this development further. Throughout, we have shown that our developments can aid the community at large, especially as we release the software packages produced with open source licenses.

There are several avenues for this work to continue. Both NCW and FBI are suitable platforms to further develop without the need to rewrite the base code. Both programs give access to new interpretations. A core competency moving forward would be the ability to automatically extract a summary of the presented interpretation.

The TMAcc descriptor construction should be further investigated. This was touched upon in Chapter 5 where the presence of the  $\log S$  component in the descriptor pairs was set as one or both. This could be extended to the other descriptors which are deemed as important for a given dataset. Additionally, new atomic properties could be made available to build the complete TMAcc descriptor. This would increase dimensionality of the descriptor which is not necessarily desirable. One could build the full TMAcc of  $x$  atomic properties and perform a feature selection like approach to reduce which properties are most important in this model. This would leave a subset of the atomic properties and a smaller descriptor which would be targeted for a given data set. NCW has made inspection of the TMAcc

straight forward but it is very labour intensive. Ideally NCW should attempt to present the user with atomic features of interests. For example, those features which occur frequently throughout the data set. If you know what structural elements are important you can look for them, but this will not uncover some novel interpretation that the model may of detected. The TMAcc interpretation are graphs. Reduced graphs are a descriptor that capture important interactions that may represent several atoms.<sup>188</sup> You do not want to look for atomic interactions, but the trends, often several atoms contribute to an affect. However, the fine granularity of the TMAcc interpretation forces an atomistic approach. Viewing the interpretation as reduced graphs would be desirable and could be the output from an automated feature detection. Although this thesis has focussed on 2D techniques the TMAcc could be implemented as a 3D descriptor for pharmacophore related work.

Within NCW the ability to interrogate the forest contents is now possible. Further tools would aid this endeavour. Like the TMAcc interpretation there is still a large manual component required currently. It would be useful for NCW to suggest subgraphs of potential interest between the trees, perhaps starting with those which occur frequently. A pathway analysis could weight each pathway (from the root node to terminal leaf node) through the tree based on the number of molecules at each node. This weighted pathway could be useful in ranking subgraphs of potential interest. With a well understood data set looking for known chemistry is fairly straightforward. What is more interesting is if NCW can detect unknown chemistry, such as a new mode of action. The challenge here is how do you know one has uncovered some novel information and not noise. Other enhancements would include the ability to read random forest models from R and other popular implementations. The tree visualiser provided from Weka does not handle large trees very well, which is generally what Weka

produces. An alternative tree visualiser using a hyperbolic view would make visual inspection more practical.

There is an overriding theme of simplifying the interpretation of the results from both programs. Although there is no need to rewrite the base code it could be ported to other Java applications that receive more widespread usage by cheminformaticians, such as KNIME<sup>36</sup> and the Chemistry Development Kit (CDK).<sup>189</sup>



# References

- (1) Brown, F. K. Cheminformatics: What is it and How does it Impact Drug Discovery. *Annu. Rep. Med. Chem.* **1998**, *33*, 375–384.
- (2) Wiswesser, W. J. How the WLN began in 1949 and how it might be in 1999. *J. Chem. Inf. Comput. Sci.* **1982**, *22*, 88–93.
- (3) Brown, R. D.; Martin, Y. C. Use of Structure-Activity Data To Compare Structure-Based Clustering Methods and Descriptors for Use in Compound Selection. *J. Chem. Inf. Model.* **1996**, *36*, 572–584.
- (4) Balaban, A. T. Applications of graph theory in chemistry. *J. Chem. Inf. Comput. Sci.* **1985**, *25*, 334–343.
- (5) Weininger, D. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *J. Chem. Inf. Comput. Sci.* **1988**, *28*, 31–36.
- (6) Weininger, D.; Weininger, A.; Weininger, J. L. SMILES. 2. Algorithm for generation of unique SMILES notation. *J. Chem. Inf. Comput. Sci.* **1989**, *29*, 97–101.
- (7) *InChI*; <http://www.iupac.org/inchi> (accessed 1 March, 2010).
- (8) *OpenSmiles*; <http://www.opensmiles.org> (accessed 1 March, 2010).
- (9) *SMARTS - A Language for Describing Molecular Patterns*; <http://www.daylight.com/dayhtml/doc/theory/theory.smarts.html> (accessed 1 March, 2010).

- (10) Willett, P.; Barnard, J. M.; Downs, G. M. Chemical Similarity Searching. *J. Chem. Inf. Model.* **1998**, *38*, 983–996.
- (11) Leach, A. R.; Gillet, V. J. *An Introduction to Chemoinformatics*, 1st ed.; Springer: Dordrecht, Netherlands, 2003.
- (12) Melville, J. L.; Riley, J. F.; Hirst, J. D. Similarity by Compression. *J. Chem. Inf. Model.* **2007**, *47*, 25–33.
- (13) Ward, J. H. Hierarchical Grouping to Optimize an Objective Function. *J. Am. Stat. Assoc.* **1963**, *58*, 236–244.
- (14) Jarvis, R.; Patrick, E. Clustering Using A Similarity Measure Based on Shared Near Neighbors. *IEEE Transactions On Computers* **1973**, *C-22*, 1025–1034.
- (15) Hansch, C.; Maloney, P. P.; Fujita, T. Correlation of Biological Activity of Phenoxyacetic Acids with Hammett Substituent Constants and Partition Coefficients. *Nature* **1962**, *194*, 178–180.
- (16) Fujita, T.; Iwasa, J.; Hansch, C. A New Substituent Constant,  $\pi$ , Derived from Partition Coefficients. *J. Am. Chem. Soc.* **1964**, *86*, 5175–5180.
- (17) Hammett, L. P. *Physical Organic Chemistry: Reaction Rates, Equilibria, and Mechanisms*; McGraw-Hill: New York, NY, USA, 1970.
- (18) Swain, C. G.; Swain, M. S.; Powell, A. L.; Alunni, S. Solvent effects on chemical reactivity. Evaluation of anion- and cation-solvation components. *J. Am. Chem. Soc.* **1983**, *105*, 502–513.
- (19) Selwood, D. L.; Livingstone, D. J.; Comley, J. C. W.; O’Dowd, A. B.; Hudson, A. T.; Jackson, P.; Jandu, K. S.; Rose, V. S.; Stables, J. N. Structure-activity relationships of antifilarial antimycin analogs: a

- multivariate pattern recognition study. *J. Med. Chem.* **1990**, *33*, 136–142.
- (20) Witten, I. H.; Frank, E. *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed.; Morgan Kaufmann: San Francisco, CA, 2005.
- (21) Guha, R.; Jurs, P. C. Interpreting Computational Neural Network QSAR Models: A Measure of Descriptor Importance. *J. Chem. Inf. Model.* **2005**, *45*, 800–806.
- (22) Tetko, I. V. Neural Network Studies. 4. Introduction to Associative Neural Networks. *J. Chem. Inf. Comput. Sci.* **2002**, *42*, 717–728.
- (23) Fröhlich, H.; Wegner, J. K.; Sieker, F.; Zell, A. Kernel Functions for Attributed Molecular Graphs - A New Similarity-Based Approach to ADME Prediction in Classification and Regression. *QSAR Comb. Sci.* **2006**, *25*, 317–326.
- (24) Mahe, P.; Ueda, N.; Akutsu, T.; Perret, J.-L.; Vert, J.-P. Graph Kernels for Molecular StructureActivity Relationship Analysis with Support Vector Machines. *J. Chem. Inf. Model.* **2005**, *45*, 939–951.
- (25) Moro, S.; Bacilieri, M.; Cacciari, B.; Bolcato, C.; Cusan, C.; Pastorin, G.; Klotz, K.-N.; Spalluto, G. The application of a 3D-QSAR (autoMEP/PLS) approach as an efficient pharmacodynamic-driven filtering method for small-sized virtual library: Application to a lead optimization of a human A3 adenosine receptor antagonist. *Bioorg. Med. Chem.* **2006**, *14*, 4923–4932.
- (26) Sæbø, S.; Almøy, T.; Aarøe, J.; Aastveit, A. H. ST-PLS: a multi-directional nearest shrunken centroid type classifier via PLS. *J. Chemom.* **2008**, *22*, 54–62.

- (27) Doweyko, A. QSAR: dead or alive? *J. Comput.-Aided Mol. Des.* **2008**, *22*, 81–89.
- (28) Llinàs, A.; Glen, R. C.; Goodman, J. M. Solubility Challenge: Can You Predict Solubilities of 32 Molecules Using a Database of 100 Reliable Measurements? *J. Chem. Inf. Model.* **2008**, *48*, 1289–1303.
- (29) *The Practice of Medicinal Chemistry*, 2nd ed.; Wermuth, C. G., Ed.; Academic Press, 2003.
- (30) Klebe, G.; Abraham, U.; Mietzner, T. Molecular Similarity Indices in a Comparative Analysis (CoMSIA) of Drug Molecules to Correlate and Predict Their Biological Activity. *J. Med. Chem.* **1994**, *37*, 4130–4146.
- (31) Cramer, R. D.; Patterson, D. E.; Bunce, J. D. Comparative molecular field analysis (CoMFA). 1. Effect of shape on binding of steroids to carrier proteins. *J. Am. Chem. Soc.* **1988**, *110*, 5959–5967.
- (32) Hopfinger, A. J.; Wang, S.; Tokarski, J. S.; Jin, B.; Albuquerque, M.; Madhav, P. J.; Duraiswami, C. Construction of 3D-QSAR Models Using the 4D-QSAR Analysis Formalism. *J. Am. Chem. Soc.* **1997**, *119*, 10509–10524.
- (33) Vedani, A.; Dobler, M. 5D-QSAR: The Key for Simulating Induced Fit? *J. Med. Chem.* **2002**, *45*, 2139–2149.
- (34) Vedani, A.; Dobler, M.; Lill, M. A. Combining Protein Modeling and 6D-QSAR. Simulating the Binding of Structurally Diverse Ligands to the Estrogen Receptor. *J. Med. Chem.* **2005**, *48*, 3700–3703.
- (35) *Marvin*; ChemAxon Kft: Budapest, Hungary, 2010.
- (36) Berthold, M. R.; Cebon, N.; Dill, F.; Gabriel, T. R.; Kotter, T.; Meinel, T.; Ohl, P.; Sieb, C.; Thiel, K.; Wiswedel, B. KNIME: The

Konstanz Information Miner. *Data Analysis, Machine Learning and Applications*, Berlin, 2008; pp 319–326.

- (37) *Pipeline Pilot*; Accelrys Inc.: San Diego, CA, 2008.
- (38) *KNIME Extensions*; Schrodinger Inc.: Portland, OR, 2009.
- (39) Hunt, A.; Thomas, D. *The Pragmatic Programmer*; Addison-Wesley: London, 2000.
- (40) Collins-Sussman, B.; Fitzpatrick, B. W.; Pilato, C. M. *Version Control with Subversion*, 2nd ed.; O'Reilly Media, Inc., 2008.
- (41) Mason, M. *Pragmatic Version Control: Using Subversion*, 2nd ed.; Pragmatic Bookshelf: Raleigh, NC, 2006.
- (42) Loeliger, J. *Version Control with Git: Powerful tools and techniques for collaborative software development*; O'Reilly Media, Inc., 2009.
- (43) *CruiseControl*; <http://cruisecontrol.sourceforge.net> (accessed 1 Mar, 2010).
- (44) *IzPack*; <http://izpack.org> (accessed 1 March, 2010).
- (45) Breiman, L.; Friedman, J.; Stone, C. J.; Olshen, R. *Classification and Regression Trees*; Chapman & Hall: Boca Raton, FL, 1984.
- (46) Lipinski, C. A.; Lombardo, F.; Dominy, B. W.; Feeney, P. J. Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings. *Adv. Drug Delivery Rev.* **1997**, *23*, 3–25.
- (47) Dietterich, T. G. An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization. *Mach. Learn.* **2000**, *40*, 139–157.

- (48) Sollich, P.; Krogh, A. Learning with ensembles: How overfitting can be useful. *Advances in Neural Information Processing Systems 8*, Denver, Colorado, 1996; pp 190–196.
- (49) Dietterich, T. In *Multiple Classifier Systems*; Springer, 2000; Vol. 1857/2000, Chapter Ensemble Methods in Machine Learning, pp 1–15.
- (50) Friedman, J.; Hastie, T.; Tibshirani, R. Additive logistic regression: a statistical view of boosting. *Annals of Statistics* **2000**, *28*, 337–407.
- (51) Baker, L.; Ellison, D. The wisdom of crowds – ensembles and modules in environmental modelling. *Geoderma* **2008**, *147*, 1 – 7.
- (52) Domingos, P. The Role of Occam’s Razor in Knowledge Discovery. *Data Min. Knowledge Discovery* **1999**, *3*, 409–425.
- (53) Breiman, L. Bagging predictors. *Mach. Learn.* **1996**, *24*, 123–140.
- (54) Svetnik, V.; Wang, T.; Tong, C.; Liaw, A.; Sheridan, R. P.; Song, Q. Boosting: An Ensemble Learning Tool for Compound Classification and QSAR Modeling. *J. Chem. Inf. Model.* **2005**, *45*, 786–799.
- (55) Freund, Y. Boosting a Weak Learning Algorithm by Majority. *Information and Computation* **1995**, *121*, 256–285.
- (56) Freund, Y.; Schapire, R. R. Experiments with a New Boosting Algorithm. *Thirteenth International Conference on Machine Learning*, Bari, Italy, 1996; pp 148–156.
- (57) Freund, Y.; Schapire, R. E. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences* **1997**, *55*, 119–139.
- (58) Valiant, L. G. A Theory of the Learnable. *Communications of the ACM* **1984**, *27*, 1134–1142.

- (59) Diao, L.; Hu, K.; Lu, Y.; Shi, C. A Method to Boost Support Vector Machines. In *Advances in Knowledge Discovery and Data Mining : 6th Pacific-Asia Conference, PAKDD 2002, Taipei, Taiwan, May 6-8, 2002. Proceedings*; Springer, 2002; p 463.
- (60) Wolpert, D. H. Stacked generalization. *Neural Networks* **1992**, *5*, 241–259.
- (61) Breiman, L. Stacked Regressions. *Mach. Learn.* **1996**, *24*, 49–64.
- (62) Ho, T. K. The random subspace method for constructing decision forests. *IEEE Trans. Pattern Anal. Mach. Intell.* **1998**, *20*, 832–844.
- (63) Rodriguez, J. J.; Kuncheva, L. I.; Alonso, C. J. Rotation Forest: A New Classifier Ensemble Method. *IEEE Trans. Pattern Anal. Mach. Intell.* **2006**, *28*, 1619–1630.
- (64) Tong, W.; Hong, H.; Fang, H.; Xie, Q.; Perkins, R. Decision Forest: Combining the Predictions of Multiple Independent Decision Tree Models. *J. Chem. Inf. Model.* **2003**, *43*, 525–531.
- (65) Lee, S. S. F.; Sun, L.; Kustra, R.; Bull, S. B. EM-random forest and new measures of variable importance for multi-locus quantitative trait linkage analysis. *Bioinformatics* **2008**, *24*, 1603–1610.
- (66) Amaratunga, D.; Cabrera, J.; Lee, Y.-S. Enriched random forests. *Bioinformatics* **2008**, *24*, 2010–2014.
- (67) Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32.
- (68) Byvatov, E.; Schneider, G. SVM-Based Feature Selection for Characterization of Focused Compound Collections. *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 993–999.
- (69) Vapnik, V. N. *Statistical Learning Theory*; Wiley: New York, 1998; p 736.

- (70) Platt, J. C. *Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines*, 1998.
- (71) Platt, J. C. Using Analytic QP and Sparseness to Speed Training of Support Vector Machines. *Advances in Neural Information Processing Systems 11*, Cambridge, MA, 1999; pp 557–563.
- (72) III, W. J. D.; Wold, S.; Edlund, U.; Hellberg, S.; Gasteiger, J. Multivariate structure-activity relationships between data from a battery of biological tests and an ensemble of structure descriptors: The PLS method. *Quant. Struct.-Act. Relat.* **1984**, *3*, 131–137.
- (73) Matthews, B. W. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochim. Biophys. Acta* **1975**, *405*, 422–451.
- (74) Fleiss, J. L. Measuring nominal scale agreement among many raters. *Psychol. Bull.* **1971**, *76*, 378–382.
- (75) Cohen, J. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement* **1960**, *20*, 37–46.
- (76) Kohavi, R. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. *International Joint Conference on Artificial Intelligence*, Montréal, Québec, Canada, 1995; pp 1137–1145.
- (77) Friedman, M. The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance. *J. Am. Stat. Assoc.* **1937**, *32*, 675–701.
- (78) Friedman, M. A Comparison of Alternative Tests of Significance for the Problem of  $m$  Rankings. *The Annals of Mathematical Statistics* **1940**, *11*, 86–92.



- (79) Triballeau, N.; Acher, F.; Brabet, I.; Pin, J. P.; Bertrand, H. O. Virtual Screening Workflow Development Guided by the “Receiver Operating Characteristic” Curve Approach. Application to High-Throughput Docking on Metabotropic Glutamate Receptor Subtype 4. *J. Med. Chem.* **2005**, *48*, 2534–2547.
- (80) Demšar, J. Statistical Comparisons of Classifiers over Multiple Data Sets. *J. Mach. Learn. Res.* **2006**, *7*, 1–30.
- (81) Zar, J. H. *Biostatistical Analysis*, 4th ed.; Prentice Hall: Upper Saddle River, NJ, 1999.
- (82) R Development Core Team, *R: A Language and Environment for Statistical Computing*; 2006.
- (83) Iman, R.; Davenport, J. Approximations of the critical region of the Friedman statistic. *Communications in Statistics Part A-Theory and Methods* **1980**, *9*, 571–595.
- (84) Nemenyi, P. *Ph.D. Thesis*, Princeton University, 1963.
- (85) Melville, J. L.; Hirst, J. D. TMAcc: Interpretable Correlation Descriptors for Quantitative Structure-Activity Relationships. *J. Chem. Inf. Model.* **2007**, *47*, 626–634.
- (86) Pastor, M.; Cruciani, G.; McLay, I.; Pickett, S.; Clementi, S. GRid-INdependent Descriptors (GRIND): A Novel Class of Alignment-Independent Three-Dimensional Molecular Descriptors. *J. Med. Chem.* **2000**, *43*, 3233–3243.
- (87) Hurst, J. R.; Heritage, T. W. *Molecular Hologram QSAR*, U.S. Patent 5,751,605, 1996.
- (88) Sutherland, J. J.; O’Brien, L. A.; Weaver, D. F. A Comparison of

Methods for Modeling Quantitative Structure-Activity Relationships. *J. Med. Chem.* **2004**, *47*, 5541–5554.

- (89) Gedeck, P.; Rohde, B.; Bartels, C. QSAR - How Good Is It in Practice? Comparison of Descriptor Sets on an Unbiased Cross Section of Corporate Data Sets. *J. Chem. Inf. Model.* **2006**, *46*, 1924–1936.
- (90) Gasteiger, J.; Marsili, M. Iterative partial equalization of orbital electronegativity—a rapid access to atomic charges. *Tetrahedron* **1980**, *36*, 3219–3228.
- (91) Wildman, S. A.; Crippen, G. M. Prediction of Physicochemical Parameters by Atomic Contributions. *J. Chem. Inf. Comput. Sci.* **1999**, *39*, 868–873.
- (92) Hou, T. J.; Xia, K.; Zhang, W.; Xu, X. J. ADME Evaluation in Drug Discovery. 4. Prediction of Aqueous Solubility Based on Atom Contribution Approach. *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 266–275.
- (93) Irwin, J. J.; Shoichet, B. K. ZINC - A Free Database of Commercially Available Compounds for Virtual Screening. *J. Chem. Inf. Model.* **2005**, *45*, 177–182.
- (94) Guha, R.; Howard, M. T.; Hutchison, G. R.; Murray-Rust, P.; Rzepa, H.; Steinbeck, C.; Wegner, J.; Willighagen, E. L. The Blue Obelisk-Interoperability in Chemical Informatics. *J. Chem. Inf. Model.* **2006**, *46*, 991–998.
- (95) *The Open Babel Package, version 2.0.0*;  
<http://openbabel.sourceforge.net/> (accessed Dec 1, 2005).
- (96) *Marvin was used for drawing, displaying and characterizing chemical structures, substructures and reactions, Marvin 5.3.0, 2010, ChemAxon (<http://www.chemaxon.com>).*

- (97) Bruce, C. L.; Melville, J. L.; Pickett, S. D.; Hirst, J. D. Contemporary QSAR Classifiers Compared. *J. Chem. Inf. Model.* **2007**, *47*, 219–227.
- (98) Clark, D. E.; Pickett, S. D. Computational methods for the prediction of “drug-likeness”. *Drug Discovery Today* **2000**, *5*, 49–58.
- (99) Claus, B. L.; Underwood, D. J. Discovery informatics: its evolving role in drug discovery. *Drug Discovery Today* **2002**, *7*, 957–966.
- (100) Hansch, C.; Fujita, T.  $\rho$ - $\sigma$ - $\pi$  Analysis. A Method for the Correlation of Biological Activity and Chemical Structure. *J. Am. Chem. Soc.* **1964**, *86*, 1616–1626.
- (101) Burges, C. J. C. A tutorial on Support Vector Machines for pattern recognition. *Data Min. Knowledge Discovery* **1998**, *2*, 121–167.
- (102) Zhang, Q.; Yoon, S.; Welsh, W. J. Improved method for predicting  $\beta$ -turn using support vector machine. *Bioinformatics* **2005**, *21*, 2370–2374.
- (103) Furey, T. S.; Cristianini, N.; Duffy, N.; Bednarski, D. W.; Schummer, M.; Haussler, D. Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics* **2000**, *16*, 906–914.
- (104) Zhang, S.-W.; Pan, Q.; Zhang, H.-C.; Zhang, Y.-L.; Wang, H.-Y. Classification of protein quaternary structure with support vector machine. *Bioinformatics* **2003**, *19*, 2390–2396.
- (105) Czermiński, R.; Yasri, A.; Hartsough, D. Use of Support Vector Machine in Pattern Classification: Application to QSAR Studies. *Quant. Struct.-Act. Relat.* **2001**, *20*, 227–240.

- (106) Lepp, Z.; Kinoshita, T.; Chuman, H. Screening for New Antidepressant Leads of Multiple Activities by Support Vector Machines. *J. Chem. Inf. Model.* **2006**, *46*, 158–167.
- (107) Luan, F.; Ma, W. P.; Zhang, X. Y.; Zhang, H. X.; Liu, M. C.; Hu, Z. D.; Fan, B. T. QSAR Study of Polychlorinated Dibenzodioxins, Dibenzofurans, and Biphenyls using the Heuristic Method and Support Vector Machine. *QSAR Comb. Sci.* **2006**, *25*, 46–55.
- (108) Warmuth, M. K.; Liao, J.; Ratsch, G.; Mathieson, M.; Putta, S.; Lemmen, C. Active Learning with Support Vector Machines in the Drug Discovery Process. *J. Chem. Inf. Model.* **2003**, *43*, 667–673.
- (109) Byvatov, E.; Fechner, U.; Sadowski, J.; Schneider, G. Comparison of Support Vector Machine and Artificial Neural Network Systems for Drug/Nondrug Classification. *J. Chem. Inf. Model.* **2003**, *43*, 1882–1889.
- (110) Tetko, I. V.; Solov'ev, V. P.; Antonov, A. V.; Yao, X.; Doucet, J. P.; Fan, B.; Hoonakker, F.; Fourches, D.; Jost, P.; Lachiche, N.; Varnek, A. Benchmarking of Linear and Nonlinear Approaches for Quantitative Structure-Property Relationship Studies of Metal Complexation with Ionophores. *J. Chem. Inf. Model.* **2006**, *46*, 808–819.
- (111) Zernov, V. V.; Balakin, K. V.; Ivaschenko, A. A.; Savchuk, N. P.; Pletnev, I. V. Drug Discovery Using Support Vector Machines. The Case Studies of Drug-likeness, Agrochemical-likeness, and Enzyme Inhibition Predictions. *J. Chem. Inf. Model.* **2003**, *43*, 2048–2056.
- (112) Quinlan, R. *C4.5: Programs for Machine Learning*; Morgan Kaufmann: San Mateo, CA, 1993.
- (113) Rusinko, A.; Farmen, M. W.; Lambert, C. G.; Brown, P. L.; Young, S. S. Analysis of a Large Structure/Biological Activity Data

- Set Using Recursive Partitioning. *J. Chem. Inf. Model.* **1999**, *39*, 1017–1026.
- (114) Nilakantan, R.; Nunn, D. S.; Greenblatt, L.; Walker, G.; Haraki, K.; Mobilio, D. A Family of Ring System-Based Structural Fragments for Use in Structure-Activity Studies: Database Mining and Recursive Partitioning. *J. Chem. Inf. Model.* **2006**, *46*, 1069–1077.
- (115) Bauer, E.; Kohavi, R. An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants. *Mach. Learn.* **1999**, *36*, 105–139.
- (116) Arodz, T.; Yuen, D. A.; Dudek, A. Z. Ensemble of Linear Models for Predicting Drug Properties. *J. Chem. Inf. Model.* **2006**, *46*, 416–423.
- (117) Agrafiotis, D. K.; Cedeno, W.; Lobanov, V. S. On the Use of Neural Network Ensembles in QSAR and QSPR. *J. Chem. Inf. Comput. Sci.* **2002**, *42*, 903–911.
- (118) Fernandez, M.; Tundidor-Camba, A.; Caballero, J. Modeling of Cyclin-Dependent Kinase Inhibition by 1H-Pyrazolo[3,4-d]Pyrimidine Derivatives Using Artificial Neural Network Ensembles. *J. Chem. Inf. Model.* **2005**, *45*, 1884–1895.
- (119) Mevik, B.-H.; Segtnan, V. H.; Næs, T. Ensemble methods and partial least squares regression. *J. Chemom.* **2004**, *18*, 498–507.
- (120) Zhang, M. H.; Xu, Q. S.; Massart, D. L. Boosting Partial Least Squares. *Anal. Chem.* **2005**, *77*, 1423–1431.
- (121) He, P.; Fang, K.-T.; Liang, Y.-Z.; Li, B.-Y. A generalized boosting algorithm and its application to two-class chemical classification problem. *Anal. Chim. Acta* **2005**, *543*, 181–191.

- (122) Zhang, M. H.; Xu, Q. S.; Daeyaert, F.; Lewi, P. J.; Massart, D. L. Application of boosting to classification problems in chemometrics. *Anal. Chim. Acta* **2005**, *544*, 167–176.
- (123) He, P.; Xu, C.-J.; Liang, Y.-Z.; Fang, K.-T. Improving the classification accuracy in chemistry via boosting technique. *Chemom. Intell. Lab. Syst.* **2004**, *70*, 39–46.
- (124) Svetnik, V.; Liaw, A.; Tong, C.; Culberson, J. C.; Sheridan, R. P.; Feuston, B. P. Random Forest: A Classification and Regression Tool for Compound Classification and QSAR Modeling. *J. Chem. Inf. Model.* **2003**, *43*, 1947–1958.
- (125) Plewczynski, D.; Spieser, S. A. H.; Koch, U. Assessing Different Classification Methods for Virtual Screening. *J. Chem. Inf. Model.* **2006**, *46*, 1098–1106.
- (126) Hawkins, D. M.; Basak, S. C.; Mills, D. Assessing Model Fit by Cross-Validation. *J. Chem. Inf. Model.* **2003**, *43*, 579–586.
- (127) Zhang, H.; Yu, C.-Y.; Singer, B. Cell and tumor classification using gene expression data: Construction of forests. *Proc. Natl. Acad. Sci. U. S. A.* **2003**, *100*, 4168–4172.
- (128) Andres, C.; Hutter, M. C. CNS Permeability of Drugs Predicted by a Decision Tree. *QSAR Comb. Sci.* **2006**, *25*, 305–309.
- (129) DeLisle, R. K.; Dixon, S. L. Induction of Decision Trees via Evolutionary Programming. *J. Chem. Inf. Model.* **2004**, *44*, 862–870.
- (130) Deconinck, E.; Zhang, M. H.; Coomans, D.; VanderHeyden, Y. Classification Tree Models for the Prediction of Blood-Brain Barrier Passage of Drugs. *J. Chem. Inf. Model.* **2006**, *46*, 1410–1419.

- (131) Li, S.; Fedorowicz, A.; Singh, H.; Soderholm, S. C. Application of the Random Forest Method in Studies of Local Lymph Node Assay Based Skin Sensitization Data. *J. Chem. Inf. Model.* **2005**, *45*, 952–964.
- (132) Chapelle, O.; Vapnik, V.; Bousquet, O.; Mukherjee, S. Choosing Multiple Parameters for Support Vector Machines. *Mach. Learn.* **2002**, *46*, 131–159.
- (133) Efron, B.; Tibshirani, R. Statistical Data Analysis in the Computer Age. *Science* **1991**, *253*, 390–395.
- (134) Meyer, D.; Leisch, F.; Hornik, K. The support vector machine under test. *Neurocomputing* **2003**, *55*, 169–186.
- (135) DePriest, S. A.; Mayer, D.; Naylor, C. B.; Marshall, G. R. 3D-QSAR of angiotensin-converting enzyme and thermolysin inhibitors: a comparison of CoMFA models based on deduced and experimentally determined active site geometries. *J. Am. Chem. Soc.* **1993**, *115*, 5372–5384.
- (136) Golbraikh, A.; Bernard, P.; Chretien, J. R. Validation of protein-based alignment in 3D quantitative structure-activity relationships with CoMFA models. *Eur. J. Med. Chem.* **2000**, *35*, 123–136.
- (137) Maddalena, D. J.; Johnston, G. A. R. Prediction of Receptor Properties and Binding Affinity of Ligands to Benzodiazepine/GABAA Receptors Using Artificial Neural Networks. *J. Med. Chem.* **1995**, *38*, 715–724.
- (138) Chavatte, P.; Yous, S.; Marot, C.; Baurin, N.; Lesieur, D. Three-Dimensional Quantitative Structure-Activity Relationships of Cyclooxygenase-2 (COX-2) Inhibitors: A Comparative Molecular Field Analysis. *J. Med. Chem.* **2001**, *44*, 3223–3230.

- (139) Sutherland, J. J.; Weaver, D. F. Three-dimensional quantitative structure-activity and structure-selectivity relationships of dihydrofolate reductase inhibitors. *J. Comput.-Aided Mol. Des.* **2004**, *18*, 309–331.
- (140) Gohlke, H.; Klebe, G. DrugScore Meets CoMFA: Adaptation of Fields for Molecular Comparison (AFMoC) or How to Tailor Knowledge-Based Pair-Potentials to a Particular Protein. *J. Med. Chem.* **2002**, *45*, 4153–4170.
- (141) Bohm, M.; Sturzebecher, J.; Klebe, G. Three-Dimensional Quantitative Structure-Activity Relationship Analyses Using Comparative Molecular Field Analysis and Comparative Molecular Similarity Indices Analysis To Elucidate Selectivity Differences of Inhibitors Binding to Trypsin, Thrombin, and Factor Xa. *J. Med. Chem.* **1999**, *42*, 458–477.
- (142) *Cerius2*; Accelrys Inc.: San Diego, CA, 2001.
- (143) Bellman, R. E. *Adaptive Control Processes*; Princeton University Press: Princeton, NJ, 1961; p 255.
- (144) Kier, L. B.; Hall, L. H. *Molecular Connectivity in Chemistry and Drug Research*; Academic Press: New York, NY, USA, 1977.
- (145) Stanton, D. T.; Jurs, P. C. Development and use of charged partial surface area structural descriptors in computer-assisted quantitative structure-property relationship studies. *Anal. Chem.* **1990**, *62*, 2323–2329.
- (146) Balaban, A. T. Highly discriminating distance-based topological index. *Chem. Phys. Lett.* **1982**, *89*, 399–404.
- (147) Kier, L. B.; Hall, L. H. *Molecular Structure Description: The Electrotological State*; Academic Press: New York, NY, USA, 1999.



- (148) Hill, T. L. *Introduction to Statistical Thermodynamics*; Addison-Wesley: Reading, 1960.
- (149) Rohrbaugh, R. H.; Jurs, P. C. Descriptions of molecular shape applied in studies of structure/activity and structure/property relationships. *Anal. Chim. Acta* **1987**, *199*, 99 – 109.
- (150) Cronin, M. T. D.; Schultz, T. W. Pitfalls in QSAR. *J. Mol. Struct. - THEOCHEM* **2003**, *622*, 39–51.
- (151) Johnson, S. R. The Trouble with QSAR (or How I Learned To Stop Worrying and Embrace Fallacy). *J. Chem. Inf. Model.* **2008**, *48*, 25–26.
- (152) Guha, R. On the interpretation and interpretability of quantitative structure–activity relationship models. *J. Comput.-Aided Mol. Des.* **2008**, *22*, 857–871.
- (153) Hall, L. H.; Mohny, B.; Kier, L. B. The electrotopological state: structure information at the atomic level for molecular graphs. *J. Chem. Inf. Comput. Sci.* **1991**, *31*, 76–82.
- (154) Chawla, N. V.; Bowyer, K. W.; Hall, L. O.; Kegelmeyer, W. P. SMOTE: Synthetic Minority Over-sampling TEchnique. *Journal of Artificial Intelligence Research* **2002**, *16*, 341–378.
- (155) *Apache license version 2*; <http://www.apache.org/licenses/LICENSE-2.0.html> (accessed 1 March, 2010).
- (156) *Sphinx*; <http://sphinx.pocoo.org> (accessed 1 March, 2010).
- (157) *JFreeChart*; <http://www.jfree.org/jfreechart> (accessed 1 March, 2010).
- (158) Hughes, L. D.; Palmer, D. S.; Nigsch, F.; Mitchell, J. B. O. Why Are Some Properties More Difficult To Predict than Others? A Study of

- QSPR Models of Solubility, Melting Point, and Log P. *J. Chem. Inf. Model.* **2008**, *48*, 220–232.
- (159) Paolini, G. V.; Shapland, R. H. B.; van Hoorn, W. P.; Mason, J. S.; Hopkins, A. L. Global mapping of pharmacological space. *Nat. Biotechnol.* **2006**, *24*, 805–815.
- (160) Palmer, D. S.; O’Boyle, N. M.; Glen, R. C.; Mitchell, J. B. O. Random Forest Models To Predict Aqueous Solubility. *J. Chem. Inf. Model.* **2007**, *47*, 150–158.
- (161) Palmer, D. S.; Llinàs, A.; Morao, I.; Day, G. M.; Goodman, J. M.; Glen, R. C.; Mitchell, J. B. O. Predicting Intrinsic Aqueous Solubility by a Thermodynamic Cycle. *Mol. Pharm.* **2008**, *5*, 266–279.
- (162) Hopfinger, A. J.; Esposito, E. X.; Llinàs, A.; Glen, R. C.; Goodman, J. M. Findings of the Challenge To Predict Aqueous Solubility. *J. Chem. Inf. Model.* **2009**, *49*, 1–5.
- (163) Spowage, B. M.; Bruce, C. L.; Hirst, J. D. Interpretable correlation descriptors for quantitative structure-activity relationships. *J. Cheminf.* **2009**, *1*, 22.
- (164) Borer, J. S. Angiotensin-converting enzyme inhibition: a landmark advance in treatment for cardiovascular diseases. *European Heart Journal Supplements* **2007**, *9*, E2–9.
- (165) Ondetti, M. A.; Cushman, D. W.; Soffer, R. L. Angiotensin-Converting Enzyme Inhibitors: Biochemical Properties and Biological Action. *Crit. Rev. Biochem. Mol. Biol.* **1984**, *16*, 381–411.
- (166) Patchett, A. et al. A new class of angiotensin-converting enzyme inhibitors. *Nature* **1980**, *288*, 280–283.

- (167) Cushmana, D.; Cheunga, H. Spectrophotometric assay and properties of the angiotensin-converting enzyme of rabbit lung. *Biochem. Pharmacol.* **1971**, *20*, 1637–1648.
- (168) Fuchs, S.; Xiao, H. D.; Hubert, C.; Michaud, A.; Campbell, D. J.; Adams, J. W.; Capecchi, M. R.; Corvol, P.; Bernstein, K. E. Angiotensin-Converting Enzyme C-Terminal Catalytic Domain Is the Main Site of Angiotensin I Cleavage In Vivo. *Hypertension* **2008**, *51*, 267–274.
- (169) Wei, L.; Alhenc-Gelas, F.; Corvol, P.; Clauser, E. The two homologous domains of human angiotensin I-converting enzyme are both catalytically active. *J. Biol. Chem.* **1991**, *266*, 9002–9008.
- (170) Natesh, R.; Schwager, S. L. U.; Evans, H. R.; Sturrock, E. D.; Acharya, K. R. Structural Details on the Binding of Antihypertensive Drugs Captopril and Enalaprilat to Human Testicular Angiotensin I-Converting Enzyme. *Biochemistry* **2004**, *43*, 8718–8724.
- (171) Mayer, D.; Naylor, C. B.; Motoc, I.; Marshall, G. R. A unique geometry of the active site of angiotensin-converting enzyme consistent with structure-activity studies. *J. Comput.-Aided Mol. Des.* **1987**, *1*, 3–16.
- (172) Natesh, R.; Schwager, S. L. U.; Sturrock, E. D.; Acharya, K. R. Crystal structure of the human angiotensin-converting enzyme-lisinopril complex. *Nature* **2003**, *421*, 551–554.
- (173) Corradia, H. R.; Schwager, S. L.; Nchinda, A. T.; Sturrock, E. D.; Acharya, K. R. Crystal Structure of the N Domain of Human Somatic Angiotensin I-converting Enzyme Provides a Structural Basis for Domain-specific Inhibitor Design. *J. Mol. Biol.* **2006**, *357*, 964–974.

- (174) Corradi, H. R.; Chitapi, I.; Sewell, B. T.; Georgiadis, D.; Dive, V.; Sturrock, E. D.; Acharya, K. R. The Structure of Testis Angiotensin-Converting Enzyme in Complex with the C Domain-Specific Inhibitor RXPA380. *Biochemistry* **2007**, *46*, 5473–5478.
- (175) Fernandez, M.; Liu, X.; Wouters, M. A.; Heyberger, S.; Husain, A. Angiotensin I-converting Enzyme Transition State Stabilization by His1089. *J. Biol. Chem.* **2001**, *276*, 4998–5004.
- (176) Dive, V.; Georgiadis, D.; Matziari, M.; Makaritis, A.; Beau, F.; Cunniasse, P.; Yiotakis, A. Phosphinic peptides as zinc metalloproteinase inhibitors. *Cell. Mol. Life Sci.* **2004**, *61*, 2010–2019.
- (177) van Esch, J. H.; Tom, B.; Dive, V.; Batenburg, W. W.; Georgiadis, D.; Yiotakis, A.; van Gool, J. M.; de Bruijn, R. J.; de Vries, R.; Danser, A. J. Selective Angiotensin-Converting Enzyme C-Domain Inhibition Is Sufficient to Prevent Angiotensin I-Induced Vasoconstriction. *Hypertension* **2005**, *45*, 120–125.
- (178) Azizi, M.; Rousseau, A.; Ezan, E.; Guyene, T. T.; Michelet, S.; Grognet, J. M.; Lenfant, M.; Corvol, P.; Ménard, J. Acute angiotensin-converting enzyme inhibition increases the plasma level of the natural stem cell regulator N-acetyl-seryl-aspartyl-lysyl-proline. *The Journal of Clinical Investigation* **1996**, *97*, 839–844.
- (179) Shen, X.; Xiao, H.; Li, P.; Lin, C.; Billet, S.; Okwan-Duodu, D.; Adams, J.; Bernstein, E.; Xu, Y.; Fuchs, S.; Bernstein, K. New insights into the role of angiotensin-converting enzyme obtained from the analysis of genetically modified mice. *Journal of Molecular Medicine* **2008**, *86*, 679–684.
- (180) Tzakos, A. G.; Gerotheranassis, I. P. Domain-Selective Ligand-Binding Modes and Atomic Level Pharmacophore Refinement in Angiotensin I

- Converting Enzyme (ACE) Inhibitors. *ChemBioChem* **2005**, *6*, 1089–1103.
- (181) Quinlan, R. J. Learning with Continuous Classes. *5th Australian Joint Conference on Artificial Intelligence*, Singapore, 1992; pp 343–348.
- (182) Wang, Y.; Witten, I. H. Induction of model trees for predicting continuous classes. *Poster papers of the 9th European Conference on Machine Learning*, 1997.
- (183) Golbraikh, A.; Tropsha, A. QSAR Modeling Using Chirality Descriptors Derived from Molecular Topology. *J. Chem. Inf. Comput. Sci.* **2002**, *43*, 144–154.
- (184) Mittal, R. R.; Harris, L.; McKinnon, R. A.; Sorich, M. J. Partial Charge Calculation Method Affects CoMFA QSAR Prediction Accuracy. *J. Chem. Inf. Model.* **2009**, *49*, 704–709.
- (185) Kidd, J. Life after statin patent expiries. *Nat. Rev. Drug Discovery* **2006**, *5*, 813–814.
- (186) Couzin, J. DRUG SAFETY: Withdrawal of Vioxx Casts a Shadow Over COX-2 Inhibitors. *Science* **2004**, *306*, 384–385.
- (187) Cartmell, J.; Enoch, S.; Krstajic, D.; Leahy, D. Automated QSPR through Competitive Workflow. *J. Comput.-Aided Mol. Des.* **2005**, *19*, 821–833.
- (188) Harper, G.; Bravi, G. S.; Pickett, S. D.; Hussain, J.; Green, D. V. S. The Reduced Graph Descriptor in Virtual Screening and Data-Driven Clustering of High-Throughput Screening Data. *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 2145–2156.
- (189) Steinbeck, C.; Han, Y.; Kuhn, S.; Horlacher, O.; Luttmann, E.; Willighagen, E. The Chemistry Development Kit (CDK): An Open-

Source Java Library for Chemo- and Bioinformatics. *J. Chem. Inf. Model.* **2003**, *43*, 493–500.

# Chapter A

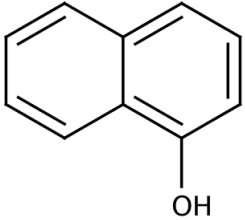
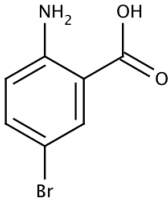
## Appendix

### A.1 Solubility challenge structures

These are the original structures provided for the solubility challenge.<sup>28</sup>

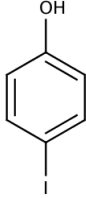
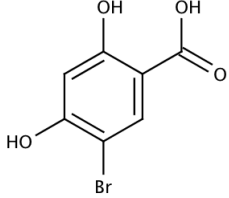
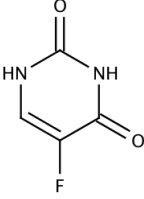
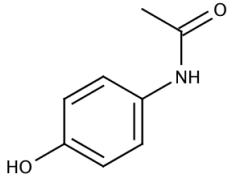
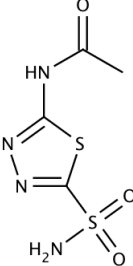
Table A.1 lists the training data and solubility data. Table A.2 contains the test structures, with unknown solubility values.

Table A.1: Solubility training set compounds

Number	Compound name	Structure
1	1-Naphthol	
2	2-amino-5-bromobenzoic acid	

Continued on next page

Table A.1 – continued from previous page

Number	Name	Structure
3	4-Iodophenol	
4	5-bromo-2,4-dihydroxybenzoic acid	
5	5-fluorouracil	
6	Acetaminophen	
7	Acetazolamide	

Continued on next page



Table A.1 – continued from previous page

Number	Name	Structure
8	Alprenolol	
9	Amantadine	
10	Amiodarone	
11	Amitriptyline	
12	Amodiaquine	

Continued on next page

Table A.1 – continued from previous page

Number	Name	Structure
13	Aspirin	
14	Atropine	
15	Azathioprine	
16	Benzylimidazole	
17	Bromogramine	

Continued on next page

Table A.1 – continued from previous page

Number	Name	Structure
18	Bupivacaine	
19	Carprofen	
20	Carvedilol	
21	Cephalothin	
22	Chlorphenamine	

Continued on next page

Table A.1 – continued from previous page

Number	Name	Structure
23	Chlorpromazine	
24	Chlorpropamide	
25	Chlorprothixene	
26	Chlorzoxazone	
27	Cimetidine	

Continued on next page

Table A.1 – continued from previous page

Number	Name	Structure
28	Ciprofloxacin	
29	Danofloxacin	
30	Deprenyl	
31	Desipramine	
32	Diazoxide	

Continued on next page

Table A.1 – continued from previous page

Number	Name	Structure
33	Diclofenac	
34	Difloxacin	
35	Diltiazem	
36	Diphenhydramine	
37	5,5-diphenylhydantoin	

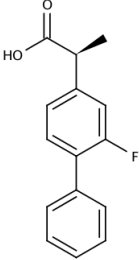
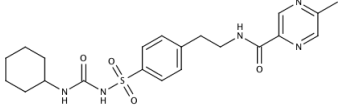
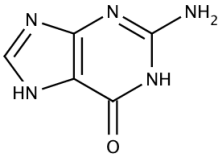
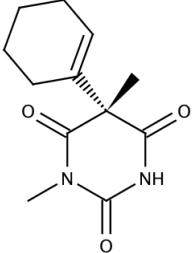
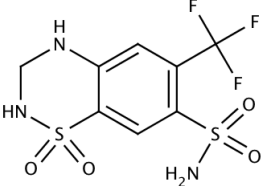
Continued on next page

Table A.1 – continued from previous page

Number	Name	Structure
38	Enrofloxacin	
39	Famotidine	
40	Fenoprofen	
41	Flufenamic acid	
42	Flumequine	

Continued on next page

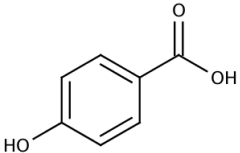
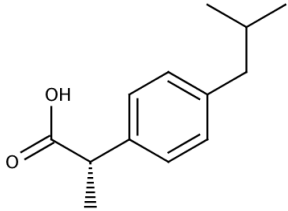
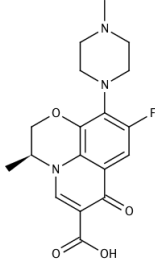
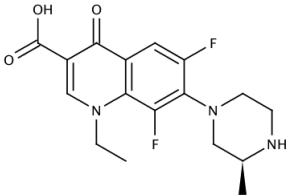
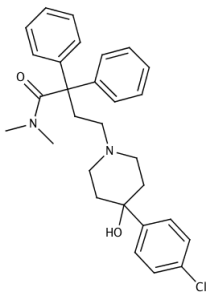
Table A.1 – continued from previous page

Number	Name	Structure
43	Flurbiprofen	
44	Glipizide	
45	Guanine	
46	Hexobarbital	
47	Hydroflumethiazide	

Continued on next page

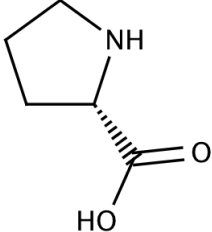
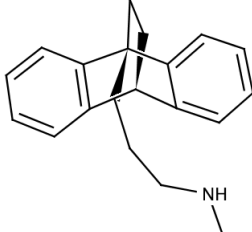
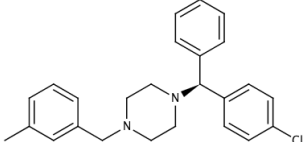
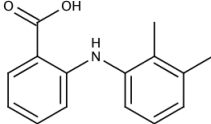
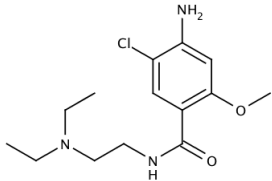


Table A.1 – continued from previous page

Number	Name	Structure
48	4-Hydroxybenzoic acid	
49	Ibuprofen	
50	Levofloxacin	
51	Lomefloxacin	
52	Loperamide	

Continued on next page

Table A.1 – continued from previous page

Number	Name	Structure
53	L-Proline	
54	Maprotiline	
55	Meclizine	
56	Mefenamic acid	
57	Metoclopramide	

Continued on next page

Table A.1 – continued from previous page

Number	Name	Structure
58	Metronidazole	
59	Miconazole	
60	Nalidixic Acid	
61	Naloxone	
62	Naproxen	

Continued on next page

Table A.1 – continued from previous page

Number	Name	Structure
63	Niflumic acid	
64	Nitrofurantoin	
65	Norfloxacin	
66	Nortriptyline	
66	Ofloxacin	

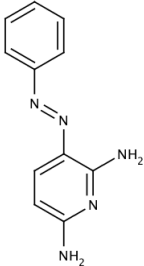
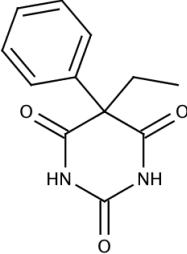
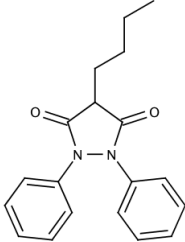
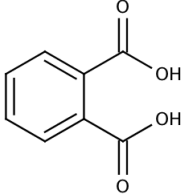
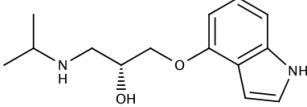
Continued on next page

Table A.1 – continued from previous page

Number	Name	Structure
67	Orbifloxacin	
68	Oxytetracycline	
69	Papaverine	
70	Pen G	
71	Phenanthroline	

Continued on next page

Table A.1 – continued from previous page

Number	Name	Structure
72	Phenazopyridine	
73	Phenobarbital	
74	Phenylbutazone	
75	Phthalic acid	
76	Pindolol	

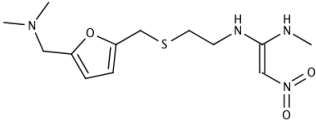
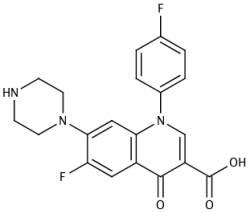
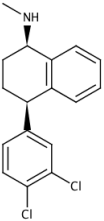
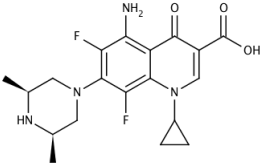
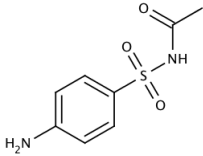
Continued on next page

Table A.1 – continued from previous page

Number	Name	Structure
77	Piroxicam	
78	Procainamide	
79	Procaine	
80	Propranolol	
81	Quinine	

Continued on next page

Table A.1 – continued from previous page

Number	Name	Structure
82	Ranitidine	
83	Sarafloxacin	
84	Sertraline	
85	Sparfloxacin	
86	Sulfacetamide	

Continued on next page

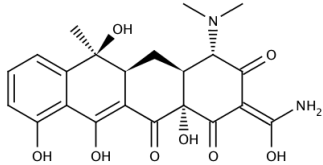
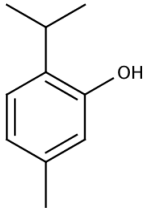
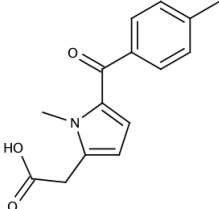
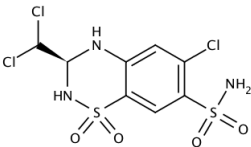
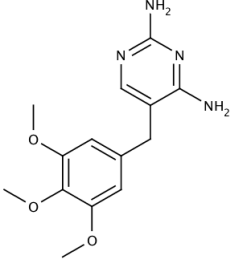


Table A.1 – continued from previous page

Number	Name	Structure
87	Sulfamethazine	
88	Sulfasalazine	
89	Sulfathiazole	
90	Sulindac	
91	Tetracaine	

Continued on next page

Table A.1 – continued from previous page

Number	Name	Structure
92	Tetracycline	
93	Thymol	
94	Tolmetin	
95	Trichlormethiazide	
96	Trimethoprim	

Continued on next page

Table A.1 – continued from previous page

Number	Name	Structure
97	Trimipramine	
98	Tryptamine	
99	Verapamil	
100	Warfarin	

Table A.2: Solubility test set compounds

Number	Compound name	Structure
1	Acebutolol	
2	Amoxicillin	
3	Bendroflumethiazide	
4	Benzocaine	
5	Benzthiazide	

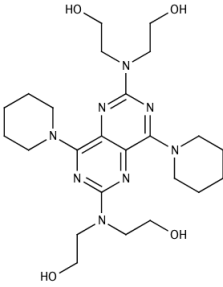
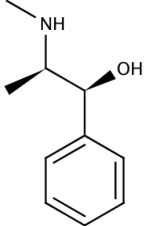
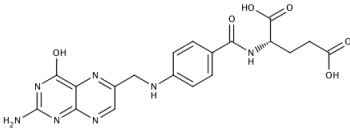
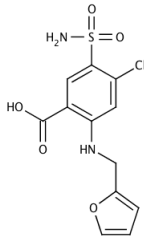
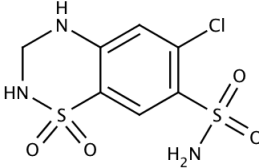
Continued on next page

Table A.2 – continued from previous page

Number	Name	Structure
6	2-chloromandelic acid	
7	Clozapine	
8	Dibucaine	
9	Diethylstilbestrol	
10	Diflunisal	

Continued on next page

Table A.2 – continued from previous page

Number	Name	Structure
11	Dipyridamole	
12	Ephedrine	
13	Folic Acid	
14	Furosemide	
15	Hydrochlorothiazide	

Continued on next page

Table A.2 – continued from previous page

Number	Name	Structure
16	Imipramine	
17	Indomethacin	
18	Ketoprofen	
19	Lidocaine	
20	Marbofloxacin	

Continued on next page

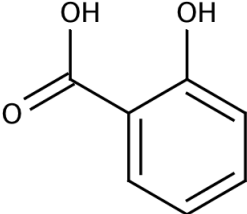
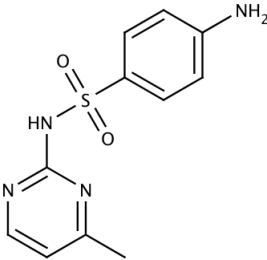
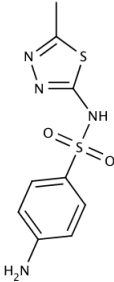
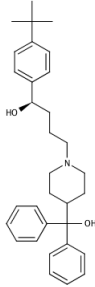
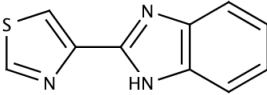
Table A.2 – continued from previous page

Number	Name	Structure
21	Meclofenamic acid	
22	Naphthoic acid	
23	Probenecid	
24	Pseudoephedrine	
25	Pyrimethamine	

Continued on next page

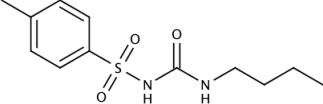
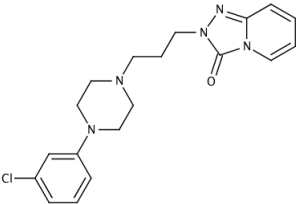


Table A.2 – continued from previous page

Number	Name	Structure
26	Salicylic acid	
27	Sulfamerazine	
28	Sulfamethizole	
29	Terfenadine	
30	Thiabendazole	

Continued on next page

Table A.2 – continued from previous page

Number	Name	Structure
31	Tolbutamide	
32	Trazodone	

---

---