

**UNIVERSIDADE DE LISBOA**  
**Faculdade de Ciências**  
**Departamento de Biologia Animal**



**MACHINE LEARNING ALGORITHMS TO PREDICT  
BLOOD-BRAIN BARRIER PERMEABILITY OF  
DRUG MOLECULES**

**Inês Filipa dos Santos Martins**

**MESTRADO EM BIOINFORMÁTICA E BIOLOGIA  
COMPUTACIONAL**  
Especialização em Bioinformática

2011



**UNIVERSIDADE DE LISBOA**  
**Faculdade de Ciências**  
**Departamento de Biologia Animal**



**MACHINE LEARNING ALGORITHMS TO PREDICT  
BLOOD-BRAIN BARRIER PERMEABILITY OF  
DRUG MOLECULES**

**Inês Filipa dos Santos Martins**

**DISSERTAÇÃO**

Dissertação orientada pelo Prof. Doutor André Osório e Cruz de Azerêdo Falcão

**MESTRADO EM BIOINFORMÁTICA E BIOLOGIA  
COMPUTACIONAL**  
Especialização em Bioinformática

2011



## Acknowledgments

First of all, I would like to thank my supervisor Prof. Doutor André Falcão that aroused my interest to bioinformatics with his course in computational biochemistry still during my degree in biochemistry, and that led to my entry into the Master of bioinformatics and computational biology. Who knew that after a while I would contact you again to search for proposals for Master's theses. Once again, my interest was piqued with the presentation of the theme of this thesis, which above any obligation to complete the Master, I really like to develop. So, thanks for the dedication, the hours of weekly meetings, the fresh ideas when everything did not seem to work, and the support throughout the project.

I am also thankful to Ana Teixeira, colleague at bio-XLDB for their time and precious help in the constitution of the final dataset and their contribution for the write of this thesis.

A "thank you" to David Raposo that indirectly open my eyes to a all new world of informatics that I did not know, and that I enjoy to know, and to guide me in a critical phase of my life course, with a shift to computer science.

I would also like to thank Marta Ribeiro and Inês Torcato from Instituto de Medicina Molecular for their contributions with important references.

To Angelo, Daniela, David, Luis, Patrícia and Pedro for the friendship and share of moments that were important to my academic and personal life. And, for all the others that I did not mention here, but that were part of my track over the five years that I spent in the Faculty of Sciences of the University of Lisbon, my appreciation.

And last, but not least, my grateful to António Broega by the wonderful person you are, by friendship, and by love. For all the support to this thesis and all steps of my life, and clear thought in moments of doubt. Thanks for being you.



*To my father for all the support and dedication  
to my education and to Noémia, the woman  
that not being my mother, is a mother to me.*





## Resumo

A incidência de doenças ligadas ao sistema nervoso central (SNC) aumenta exponencialmente depois dos 65 anos e o aumento da esperança média de vida vem aumentar a população mundial com mais de 65 anos. Depressões, dor crónica, epilepsia e enxaquecas são algumas condições clínicas (distúrbios do SNC) que apresentam tratamento no entanto, podem ser consideradas exceções perante a maioria dos distúrbios do SNC, que incluem doenças neurodegenerativas e que têm muito poucas opções de tratamento. Alguns destes casos são a doença de Alzheimer, doença de Parkinson, esclerose amiotrófica lateral, esclerose múltipla, cancro do cérebro, entre outras. O aumento do número de casos com doenças neurodegenerativas veio então aumentar a necessidade de descoberta e desenvolvimento de novos fármacos que combatam e curem estas doenças. Por exemplo, a doença de Alzheimer afecta actualmente 18 milhões de pessoas a nível mundial e estima-se que em 2025 o número de casos chegue aos 34 milhões. Este pequeno grande exemplo demonstra a preocupação e a necessidade de novos fármacos que dêem esperança de cura a estes e muitos outros casos de doenças neurodegenerativas. No entanto, esta é uma tarefa com dificuldades acrescidas face ao desenvolvimento de fármacos com alvos noutras partes do organismo que não o cérebro. O tempo necessário para que um fármaco com alvos no SNC chegue aos mercados pode ir de 12 a 16 anos, enquanto que um fármaco que não actue no SNC vai de 10 a 12 anos. Esta grande diferença prende-se com a complexidade do cérebro, a tendência desses fármacos para causar efeitos secundários e a existência da barreira hemato-encefálica (BHE).

A BHE separa o fluído cerebroespinal do sangue em circulação, impedindo a livre passagem da grande maioria das substâncias (mesmo as pequenas moléculas) da corrente sanguínea para o cérebro. De entre estas substâncias encontram-se os fármacos, estimando-se que cerca de 95% destes não consigam atravessar a barreira, deixando-nos apenas uma ínfima parte susceptível de atravessá-la. Isto significa que fármacos que têm efeito terapêutico noutras partes do organismo, são inúteis para alvos no SNC pela baixa permeabilidade da BHE, tornando muito difícil o tratamento de infecções bacterianas e virais no SNC através dos antibióticos convencionais. Muitas das doenças neurodegenerativas apresentadas acima poderiam ser tratadas com fármacos, enzimas ou genes já descobertos, infelizmente estas não conseguem atravessar a BHE.

Além desta clara dificuldade imposta pela fisiologia existe ainda outra relacionada com a experimentação. A permeabilidade de um composto na BHE é convencionalmente medida pelo logaritmo do coeficiente de partição sangue-cérebro, tratando-se de uma medida da lipofilicidade. No entanto, a sua aquisição por ensaios experimentais é muito dispendiosa e o processo moroso ainda mais quando estão em análise milhares de compostos. Este processo integra a primeira fase de desenvolvimento de um fármaco que pode demorar até dois anos e em que são analisados, em média, 10000 compostos. São necessários até 16 anos para que um medicamento que actue do SNC chegue aos mercados e seja comercializado, nesse sentido, é objectivo deste trabalho contribuir com metodologias que visem a diminuição do tempo da primeira fase de desenvolvimento de um fármaco e por conseguinte, a diminuição do número de anos final.

Para tal entramos no domínio da chamada quimio-informática, uma área na fronteira entre a Química e a Informática, que tenta retirar informação útil contida nos compostos químicos pela utilização e desenvolvimento de sistemas automatizados de recolha e processamento de dados, algoritmos, técnicas computacionais e métodos estatísticos. A utilização desta informação permite estabelecer correlações que só são possíveis com recurso aos computadores, às metodologias computacionais e à inteligência artificial. Estes recursos disseminaram-se fortemente na indústria química e farmacêutica que fazem investigação de novos produtos utilizando no processo métodos computacionais. A procura de metodologias que contribuam e favoreçam o desenvolvimento de novos compostos é emergente havendo uma grande procura de especialistas nesta área.

Este trabalho, de forte componente computacional, faz uso de algoritmos de aprendizagem automática para determinar com melhor precisão se um determinado fármaco pode passar a barreira hemato-encefálica. Outros estudos têm-se baseado sobretudo na utilização de metodologias de aprendizagem *standard* usando como variáveis preditivas propriedades físicas e químicas dos compostos em análise. Neste trabalho utilizam-se descritores e impressões digitais moleculares (*fingerprints*) para analisar a diversidade estrutural existente num conjunto de compostos permitindo estabelecer medidas de semelhança estrutural entre moléculas e com isso prever o comportamento de um composto "desconhecido".

Assim, no âmbito do projecto foram desenvolvidas três metodologias baseadas no algoritmo do vizinho mais próximo, nas redes neuronais artificiais e nas *Random forests* (floresta aleatória). Foram constituídos três conjuntos de dados (conjunto de dados I, conjunto de dados II e conjunto de dados III) com 628 (374 BHE+/254 BHE-), 729 (466 BHE+/263 BHE-) e 950 (466 BHE+/484 BHE-) moléculas, respectivamente. Os dois primeiros algoritmos testados foram o algoritmo do vizinho mais próximo e as redes neuronais, os quais seguem uma mesma lógica baseada no uso das impressões digitais moleculares. A partir destes, são geradas matrizes de similaridade onde é calculada uma medida de semelhança entre cada par de moléculas. Segue-se a análise em coordena-

das principais que reduz a matriz num vector de coordenadas com  $n$  dimensões, que são os descritores moleculares destes dois métodos. O algoritmo do vizinho mais próximo foi treinado e testado com o conjunto de dados I e 30 descritores tendo sido aplicada a validação cruzada com o método *leave-one-out*. Este algoritmo consegue prever correctamente a classificação dos compostos em 81.75%, 76.57% e 79.78% para BHE+ (compostos que passam a BHE), BHE- (compostos que não passam a BHE) e globalidade das duas classes, respectivamente. A metodologia que implementa as redes neuronais também foi treinada com o conjunto de dados I, tendo sido usados 40 descritores moleculares. O modelo foi validado com validação cruzada 5-fold, obtendo-se os valores de 77.16%, 81.45% e 79.68% para BHE+, BHE- e globalidade das duas classes, respectivamente.

O modelo das *Random Forests* foi construído com base em 1051 descritores moleculares, de entre as impressões digitais moleculares foram também utilizados a massa molecular, massa molecular média, número de anéis, número de ligações em anéis e número de ligações possíveis para cada átomo de um subconjunto definido (C, N, O, S, Br, I, Cl, F e Na). Além da utilização de descritores moleculares diferentes dos que têm sido até agora utilizados na literatura, a metodologia apresentada neste trabalho é nova e traz inovação a esta área de desenvolvimento na medida em que faz uso da diferença percentual entre as duas classes de compostos, 5% para BHE+ e 95% BHE-. A maior parte dos estudos analisados na literatura apresentam conjuntos de dados populados maioritariamente por compostos que atravessam a BHE, no entanto esta situação não transparece a realidade. Nesse sentido, neste trabalho foi desenvolvido uma metodologia que efectua amostragens diferenciadas usando uma estratégia de *oversampling* (sobre-amostragem) nos compostos BHE- para melhor reflectir a realidade de modo a representar 95% de compostos BHE- e apenas 5% de compostos BHE+. Os resultados demonstram-se bastante promissores, conseguindo a *Random Forest* prever correctamente a classificação dos compostos em 92%, 77.2% e 82.6%, para BHE+, BHE- e globalidade das duas classes, respectivamente. Esta metodologia apresenta os melhores resultados, ficando mesmo acima da maior parte dos valores apresentados na literatura.

Com estes valores os principais objectivos desta tese são concretizados, tendo sido feito um contributo bastante positivo para a área, principalmente para uma previsão mais correcta da passagem de um composto pela barreira hemato-encefálica, o qual é um ponto chave no desenvolvimento de fármacos para o tratamento de doenças ao nível do SNC, que precisam obrigatoriamente de passar a BHE. Neste trabalho apenas foram apresentadas as três metodologias que se revelaram mais promissoras e com os melhores resultados embora ao longo do projecto tenham sido desenvolvidas e testada outras metodologias.

Face aos resultados obtidos é proposta de trabalho futuro a concretização de uma aplicação que implemente a metodologia da *Random Forest* com vista a que seja um contributo à Ciência e principalmente à área de desenvolvimento de novos fármacos, contribuindo para uma análise rápida de grandes bibliotecas de compostos, restringido o leque

de compostos com necessária intervenção de ensaios experimentais, diminuindo também o custo associado à investigação de compostos na primeira fase de desenvolvimento de um fármaco.

**Palavras-chave:** Aprendizagem automática, Algoritmo do vizinho mais próximo, Barreira hemato-encefálica, Floresta aleatória, Pré-triagem de fármacos *in silico*, Químio-informática, Redes neuronais



## Abstract

The increasing number of cases with neurodegenerative diseases has increased the need to discover and develop new drugs to combat and cure these diseases. However, this is a difficult task due to the existence of the blood-brain barrier (BBB), which prevents the free passage of most substances from the bloodstream to the brain. Drugs are among these substances and it is estimated that about 95% of these fail to cross the barrier, leaving us only a small fraction likely to cross it.

Besides this clear difficulty imposed by physiology, logBB (blood-brain permeation coefficient), a coefficient used to measure the ability of a compound to cross the BBB, is difficult to acquire by experimental process, being very expensive and time consuming even more when there are thousands of compounds under analysis. This process includes the first phase of developing a drug that can take up to two years and could have in analysis an average of ten thousands compounds. Thus, one of the objectives is to contribute with methodologies to improve current methodologies for in silico compound pre-screening, so that a reasonable level of confidence in the effectiveness of an unknown molecule passing the BBB can be ascertained.

This work has a strong computational component and makes use of state-of-the-art machine learning algorithms, namely the k-nearest neighbor algorithm, neural networks and random forests to determine more accurately whether a particular drug can pass the BBB. Molecular descriptors and molecular fingerprints are used to examine the structural diversity existing in a number of compounds allowing to establish measures of structural similarity between molecules and thereby predict the behavior of an "unknown" compound.

The methodology that uses random forests provides the best results, predicting correctly the classification of compounds in 92%, 77.2% and 82.6%, referring these values to BBB+, BBB- and overall accuracies, respectively. Results achieved are on par and even slightly better compared to current results described in literature.

**Keywords:** Blood-brain barrier, Cheminformatics, in silico drug pre-screening, k-Nearest Neighbor, Neural Networks, Random Forests







# Contents

<b>List of Figures</b>	<b>xvii</b>
------------------------	-------------

<b>List of Tables</b>	<b>xix</b>
-----------------------	------------

<b>1 Introduction</b>	<b>1</b>
1.1 Drug development . . . . .	1
1.2 Problem . . . . .	2
1.3 Motivation . . . . .	3
1.4 Objective . . . . .	3
1.5 Structure of the thesis . . . . .	4
<b>2 State of the Art</b>	<b>5</b>
2.1 CNS barriers . . . . .	5
2.2 Blood-Brain Barrier . . . . .	6
2.3 Algorithms for <i>in silico</i> BBB permeability . . . . .	9
<b>3 Methods</b>	<b>13</b>
3.1 Tools . . . . .	13
3.1.1 R . . . . .	13
3.1.2 Python . . . . .	13
3.1.3 OpenBabel . . . . .	14
3.2 Molecular representation . . . . .	15
3.2.1 SMILES - Simplified Molecular Input Line Entry Specification . . . . .	15
3.2.2 Molecular descriptors . . . . .	17
3.2.3 Fingerprints . . . . .	18
3.3 Molecular similarity . . . . .	20
3.4 Statistical methods . . . . .	22
3.4.1 Principal Coordinates Analysis . . . . .	22
3.4.2 Statistical measures . . . . .	24
3.4.3 Cross-Validation . . . . .	26
3.5 Learning methods . . . . .	27
3.5.1 k-Nearest Neighbor . . . . .	27

3.5.2	Neural Networks . . . . .	28
3.5.3	Random Forest . . . . .	31
<b>4</b>	<b>Data</b>	<b>35</b>
4.1	Drug molecules dataset . . . . .	35
4.2	Data preparation and computational approach . . . . .	36
4.3	Implementations . . . . .	37
4.3.1	Principal Coordinates Analysis . . . . .	37
4.3.2	k-NN . . . . .	37
4.3.3	Neural networks . . . . .	38
4.3.4	Random Forest . . . . .	39
<b>5</b>	<b>Results and discussion</b>	<b>43</b>
5.1	k-Nearest Neighbour . . . . .	43
5.2	Neural Networks . . . . .	46
5.3	Random Forests . . . . .	49
5.3.1	Approach A . . . . .	50
5.3.2	Approach B . . . . .	50
5.3.3	Approach A <i>versus</i> Approach B . . . . .	51
5.3.4	Variable importance analysis . . . . .	52
5.4	Discussion . . . . .	54
<b>6</b>	<b>Conclusion</b>	<b>59</b>
<b>A</b>	<b>Methodologies</b>	<b>61</b>
A.1	Obtaining SMILES from compound names . . . . .	61
A.2	Obtaining molecular descriptors: MW and LogP . . . . .	62
A.3	Similarity matrices for PCoA analysis . . . . .	62
A.4	Principal Coordinates Analysis . . . . .	65
A.5	k-Nearest Neighbor . . . . .	66
A.6	Neural Networks . . . . .	68
A.7	Random Forests . . . . .	70
<b>B</b>	<b>Molecular descriptors</b>	<b>73</b>
<b>C</b>	<b>Results</b>	<b>75</b>
	<b>Glossary</b>	<b>78</b>
	<b>Bibliography</b>	<b>83</b>

# List of Figures

1.1	Drug development process . . . . .	1
2.1	Barriers at central nervous system . . . . .	6
2.2	General and brain capillaries . . . . .	7
2.3	Transport via blood-brain barrier . . . . .	7
2.4	Estimated density of molecular weight for BBB+ and BBB- molecules . .	9
3.1	Pybel representation of FP2 fingerprint bits . . . . .	19
3.2	Molecules representation in a two dimensional space . . . . .	23
3.3	Classification with k-NN algorithm . . . . .	28
3.4	Feed-forward network with a single hidden layer . . . . .	30
4.1	Oversampling process . . . . .	39
5.1	Phi measures for tested Tanimoto coefficient variants with k-NN algorithm	44
5.2	Phi measures for all the evaluated coefficients with k-NN algorithm . . .	45
5.3	Phi value in order to the number of neighbors . . . . .	46
5.4	Phi measures for tested Tanimoto coefficient variants with NN algorithm .	47
5.5	Phi measures for all the evaluated coefficients with NN algorithm . . . .	48
5.6	Phi value in order to the number of neurons . . . . .	49
5.7	Importance variable analysis . . . . .	52
5.8	Number of the bonds in rings and its distribution among BBB- and BBB+ molecules . . . . .	53
5.9	Distribution of oxygens with single bond among BBB- and BBB+ molecules	54
C.1	Importance variables analysis and its molecular structures . . . . .	75
C.2	Analysis of less important variables . . . . .	76



# List of Tables

2.1	Prediction accuracies and phi for BBB+ and BBB- compounds from different studies reported in the literature . . . . .	10
3.1	Multiplicities as molecular descriptors . . . . .	18
3.2	Contingency table for similarity measures . . . . .	21
3.3	Confusion matrix . . . . .	24
4.1	Molecular descriptors for random forest model . . . . .	39
5.1	Phi measures for evaluated coefficients with k-NN algorithm . . . . .	45
5.2	k-NN classification results for several statistical measures . . . . .	47
5.3	Phi measures for evaluated similarity coefficients with NN algorithm . . . .	48
5.4	NN classification results for several statistical measures . . . . .	49
5.5	Random Forest results with approach A . . . . .	50
5.6	Random Forest results with approach B . . . . .	51
5.7	Approaches A and B: results . . . . .	52
5.8	Molecular weight and its average (AMW): comparison and distribution in BBB+ and BBB- molecules . . . . .	53
5.9	Methodologies and studies comparison . . . . .	55
5.10	Classes of molecular descriptors used in studies reported in literature and in this work . . . . .	57
B.1	Classes and molecular descriptors . . . . .	73



# Chapter 1

## Introduction

### 1.1 Drug development

The process of drug development until it reaches the market can take at least 10 years. In the worst case, it can take up to 16 years. Many are the compounds studied in the early stages of the developing process, but very few can reach the final stages of the process.

Successful drugs have to fulfill some criteria, dependent of physicochemical properties and pharmacokinetic profile, including how the compound is absorbed, distributed, metabolized, and excreted (ADME).

The drug development process can be divided into three major phases [Alzheimer's Drug Discovery Foundation, 23-07-2011] (see Figure 1.1):

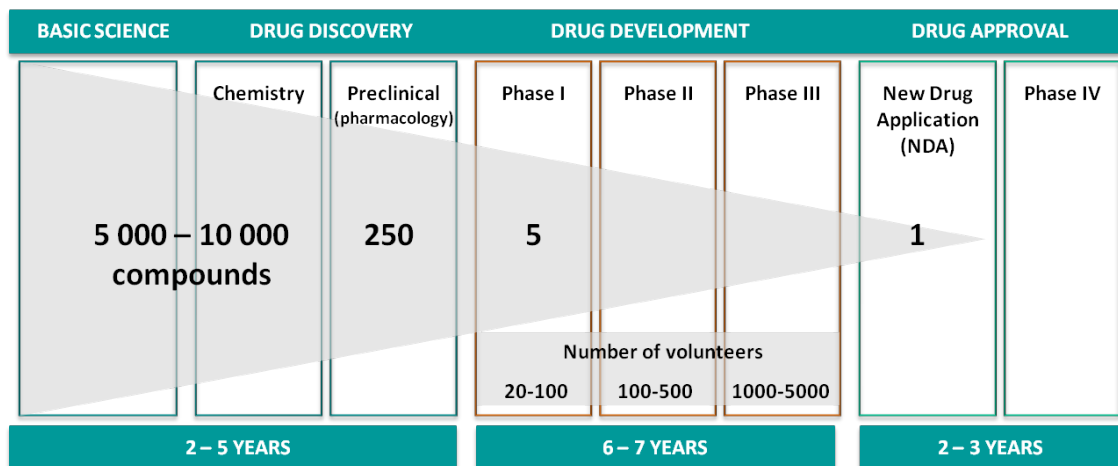


Figure 1.1: Stages of drug development process, including compounds, volunteers and time in each stage. Adapted from Alzheimer's Drug Discovery Foundation [23-07-2011].

**Basic science** - at this level, the aim is to discover information about individual chemicals and biological processes.

**Drug discovery** - develop specific chemicals and study their effects on identified disease

targets. When a potentially viable drug candidate is identified, the compound proceeds to laboratory testing against specific targets to determine its effect. This phase requires a more sophisticated chemistry and it is performed tests on animal models of disease. In study can be between 5000 to 10000 compounds, and it estimated that only 250 will progress to pre-clinical development with animals.

**Drug development** - as outlined before, many of the chemical compounds do not reach this phase. Nevertheless, when a chemical compound has an impact on a specific target, human testing is conducted through chemical trials to show the safety and efficacy so it may be approved as a drug and marketed. The drug development phase is then divided into three phases which differs the number of volunteers take to test the compounds. In phase I is study the effects on healthy human subjects (20 to 100 people), in the next phase are made a limited study of patients (100 to 500 patients), and finally, in phase III, are made comparative studies on a large number of patients (1000 to 5000 patients). Only 5 of the 250 compounds reach the phase I. However, just one will survive over the remaining phases and become an approved drug.

The compound numbers presented above represent an average success rate among all therapeutic areas.

It can be identified a fourth phase pointed out in Figure 1.1 as drug approval, that corresponds to the authorisation process of the new drug and to the continued comparative studies (phase IV of clinical trials).

## 1.2 Problem

It was described the overall perspective of drug development process, including compound numbers and an average of the success rate for all therapeutic areas, but when referring to Central Nervous System (CNS) drugs, they have only an 8% [Alzheimer's Drug Discovery Foundation, 23-07-2011] success rate of being approved and receive the status of drug. This percentage is clearly lower when comparing with areas like the cardiovascular, which has a 20% success rate. Among this low chance of success, the time taken to bring a new CNS drug can reach 16 years compared with non-CNS drugs (12 years). This is attributable to various features, including the complexity of the brain, the tendency to cause side effects on CNS, and the existency of the Blood-Brain Barrier (BBB) that CNS drugs have to cross [Alavijeh et al., 2005].

Most organs of the body are perfused by capillaries with endothelial cells that have small pores that allow the movement of small molecules from the circulation into the organ. Although in brain capillary the diffusion is inhibited by the presence of tight junctions between the endothelium cells, this property prevents most of the molecules of



crossing the barrier leaving many of the CNS disorders without a therapy [Alavijeh et al., 2005].

The features of BBB represent a *problem* in CNS drug development, when 95 percent of molecules show poor or none permeability to BBB [Pardridge, 2005, King, 2011, Tsaïoun et al., 2009]. Therefore, BBB penetration is one of the keys that are taken into account in chemical toxicological studies and in drug design [Zhang et al., 2008]. BBB permeability of drug molecules is commonly measured by logBB which represent the brain-to-blood (BB) concentration ratio. It is defined as the ratio of a drug concentration in brain tissue to the drug concentration in blood. Although the measurement of logBB is the most direct approach to know whether a molecule cross the BBB, the experiments are very expensive and time consuming [Zhao et al., 2007] and constitute a hindrance when are involved more than 5000 compounds.

## 1.3 Motivation

The Blood-Brain Barrier (BBB) is a membrane that separates circulating blood and the brain extracellular fluid in the CNS. Some of main functions of this barrier comprises the protection of the brain from *foreign substances* in the blood that may injure it, protection against hormones and neurotransmitters in the rest of the body and maintenance of a constant environment for the brain. Therefore, the BBB has special features that make it almost impenetrable to most drugs. It has a selective permeability and the molecules that generally cross are: nonpolar and lipophilic, properties related with high octanol/water partition coefficient (logP) and low molecular/molar weight - smaller molecules are more likely to pass through the BBB.

Due to the low permeability of BBB the process of discovering substances that cross the membrane and are active on CNS is hard and slow, and requires, in the early stages, a large number of chemical compounds.

The prediction of drug molecules permeability would be an useful tool to assist the experimental drug discovery process [Doniger et al., 2000], decreasing the time of the earliest stages and therefore, the time required for a drug reaches the market and us would be considerably less.

Computacional approaches are needed and have been introduced as pre-screening tools for large chemical databases to reduce the cost and enhance the speed of BBB permeability analysis [Li et al., 2005b, Zhang et al., 2008].

## 1.4 Objective

To address the problem of drug permeability of BBB a robust comparison of different methodologies is needed. This thesis proposes the use of chemical information present in

molecules in an attempt to produce a novel computational approach with better predictive rates. Thus, the main assumption in this thesis, is that molecules sharing a similar structure should have a similar behavior through the barrier and so, if a molecule crosses the blood-brain barrier, a similar one must also cross the barrier.

Our aim is to test different approaches based on the structural similarity of molecules for predicting BBB permeability of drug molecules, with high and balanced accuracy rates for both BBB+ and BBB- molecules.

The final methodology intends to be a contribution for the drug development process, at the early stages of compounds discovery, reducing the time expended in screening large numbers of pre-clinical candidates compounds.

## 1.5 Structure of the thesis

The work is divided into six chapters. Starting with the introductory chapter (the first), it is presented the pillars on which this thesis develops, with respect to the problem of low permeability of BBB that leads to a further delay in the development of drugs that can act on CNS, passing for the contributions that work intends to give for this research field, being presented the objectives.

The second chapter is dedicated to the presentation of CNS barriers and mainly the BBB, its characteristics and the functions that make it unique in the body, ending with a background, where it is exposed some of the previous works in literature, with relevance in the area.

Chapter three delineates the methods underlining the approaches developed, the machine learning algorithms, the statistical methods and also the tools used throughout the work.

Chapter four focuses on data presentation and building of computational approaches, with emphasis for datasets constitution and algorithms implementation.

Moving on to results and discussion, they are topic for chapter five. It is presented the results for each of the developed approaches, followed by comparison of results between approaches and with the previous work presented in chapter two.

In the last chapter, beyond the conclusions of the work, some proposals are presented for future work that could be constructed upon this project.

The appendices present the relevant computer program used both for data processing, algorithm development and result analysis.

# Chapter 2

## State of the Art

This chapter is divided into three sections. Section 2.1 characterizes the barriers that are part of CNS. Next, Section 2.2 focuses on the blood-brain barrier, being presented its properties and functionalities, and Section 2.3 summarises some of the previous studies.

### 2.1 CNS barriers

The CNS has two controlling barriers, that protects the brain interstitial fluid (ISF) from blood circulating substances. The first and the largest is the BBB, and the second is the blood-cerebrospinal fluid barrier (BCSFB). The BBB has the greatest control on the brain homeostasis, acting as the first control line. More barriers could be identified, however only this two are mentioned as they has a role in permeability of substances that circulates in bloodstream [Cipolla, 2009]. The Figure 2.1 shows the localization of both barriers.

The BBB is formed by endothelial cells, astroglia, pericytes and basal lamina connecting the cellular systems. Endothelial cells form an almost *impenetrable* barrier by the presence of tight junctions, while the other components contributes for the integrity of the barrier. The next section focuses on the role and properties of this complex barrier.

The BCSFB regulates the passage of molecules from the blood to cerebrospinal fluid (CSF), since it can exchange molecules with ISF. The barrier is then located between the blood and the CSF in the epithelium of the choroid plexus. This along with arachnoid membrane form a barrier between the blood and CSF. The passage of substances from the blood through the arachnoid membrane is prevented by tight junctions. The arachnoid membrane is generally impermeable to hydrophilic substances, and its role is forming the Blood-CSF barrier is largely passive. The choroid plexus forms the CSF and actively regulates the concentration of molecules in the CSF [Misra et al., 2003]. While endothelium cells of choroid plexus capillaries are highly permeable to hydrophobic substances, the epithelial cells are glued with tight junctions, although they are more permeable than those find in BBB.

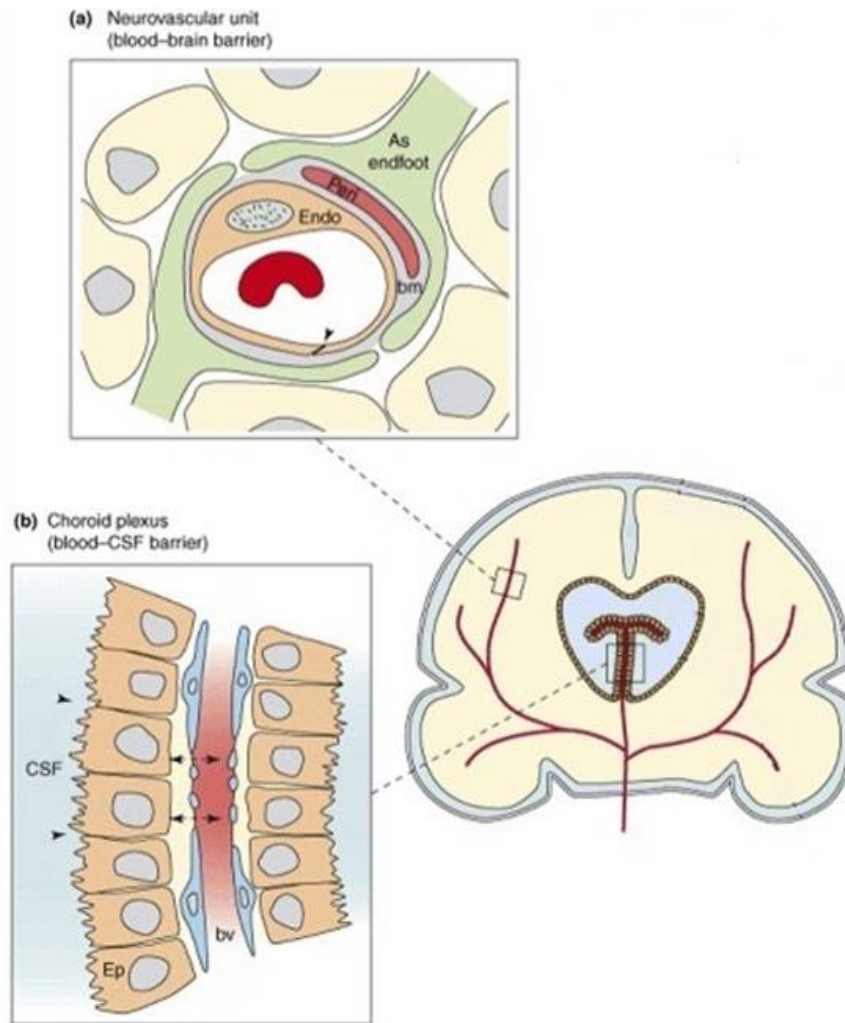


Figure 2.1: Barriers at central nervous system: a) blood-brain barrier, and b) blood-cerebrospinal fluid barrier. Endo - endothelial cells; bm - basement membrane; Peri - pericytes; As endfoot - nearby astrocytes; bv - blood vessels; Ep - epithelial cells. Adapted from Cipolla [2009].

## 2.2 Blood-Brain Barrier

The BBB represents 400 miles [Begley et al., 2000] of capillaries that separates the brain from the bloodstream and it is composed of a continuous layer of endothelial cells sealed by tight junctions, this characteristic distinguishes BBB of blood vessels in other parts of the body that have small pores for the movement of small molecules into the organ interstitial fluid, Figure 2.2 shows the difference between a general capillary and a capillary located in the brain. It is responsible by maintain the homeostasis of CNS, preventing the entry of substances that can disturb the function of neurons. It prevents the entry of the majority of molecules, except those that are small and lipophilic or those that enters via a mediated transport, and are needed for the proper functioning of the brain. The BBB protects

the brain from fluctuations in ionic composition that could disturb synaptic and axonal signalling [Abbott et al., 2006].

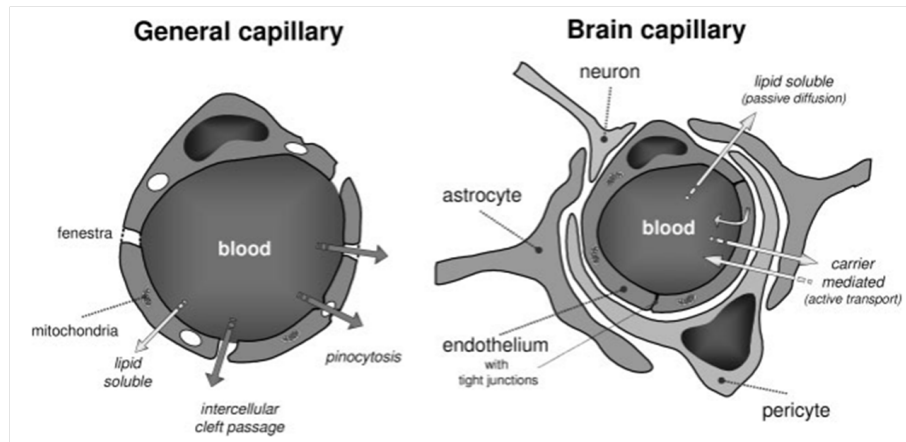


Figure 2.2: Differences between general and brain capillaries [Dermietzel et al., 2006].

The transport through the BBB can occur via five different ways [Abbott et al., 2006, Alavijeh et al., 2005, Cipolla, 2009] and are represented in Figure 2.3:

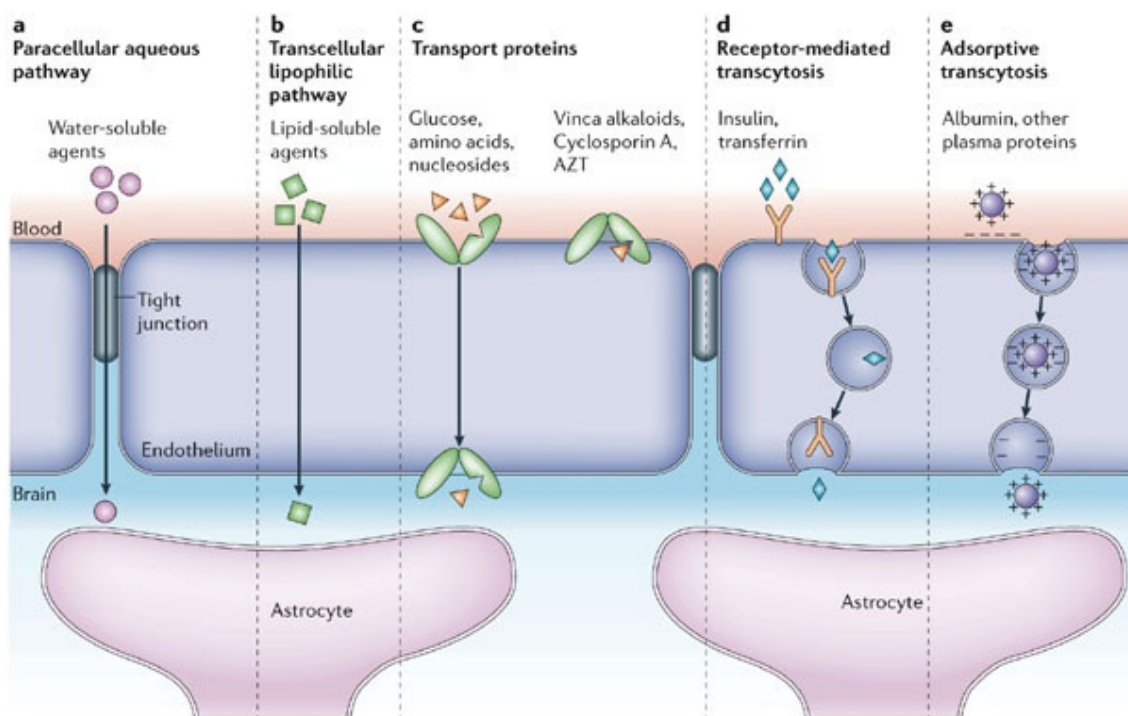


Figure 2.3: The five routes of transport via blood-brain barrier [Abbott et al., 2006].

**Paracellular pathway** - represents the diffusion of polar solutes through tight junctions.

**Transcellular lipophilic pathway** - correspond to a passive diffusion of lipid-soluble agents, this depends on the physicochemical properties of molecules.

**Mediated proteins transport** - influx transport systems responsible for the entry of nutrients and endogenous compounds, such as amino acids, monocarboxylic acids, amines, hexoses, thyroid hormones, purine bases and nucleosides. Some transporters act as active efflux pumps, they accounts for poor BBB permeability of certain drugs, this transporters are able to expel a large number of very different compounds from brain interstitial fluid. Among them is the transporter phosphorylated glycoprotein (P-gp), which is responsible for inhibit the entry of hydrophobic drugs into the brain.

**Receptor-mediated transcytosis** - some proteins (as insulin and transferrin) enter in brain by specific receptor-mediated endocytosis and transcytosis.

**Adsorptive transcytosis** - native plasma proteins, as albumin, are surrounded by positive charges (cationization) to increase the transport via adsorptive-mediated endocytosis and transcytosis.

In the development of CNS-drugs it is not enough to cross the BBB, the molecules have to be in brain long enough to produce the desirable effect, and that depends of its ADME properties. Considering the development of CNS-drugs, they are absorbed when enter the blood capillaries and reach the brain, crossing the BBB. The transport can be done through one of the five processes describe above, and in part depends of lipophilicity and solubility of the drug. It can be thought that the greater the lipophilicity, greater will be the permeability, however this is not necessarily true. High lipophilicity also shows high affinity for metabolic enzymes, leading to a quick degradation of drug molecules. After enter in bloodstream, a drug molecule distributes across the body. At the same time, the elimination process start as a consequence of the metabolism and excretion. Therefore, a drug has to be administrated in a proper dose, enough to reach its target and produce its effect.

In short, the BBB is a very selective membrane that prevents the entry of most molecules, making the most of drug molecules useless for CNS disorders. In CNS drug development, the ability to permeate across the BBB is required, but nonetheless it does not mean that will have a therapeutic effect. On the other hand, it is expected that non-CNS drug molecules do not show ability to cross the BBB, preventing from potential side effects [Zhang et al., 2008]. Figure 2.4 shows the distribution of compounds in order to molecular weight, in plot A is considered that 50% of the molecules cross the BBB and 50% of the molecules are not able to cross it (same ratio of compounds), on the other hand plot B considers prior probabilities of compounds classes, therefore only 5% of the compounds cross the BBB and 95% of the compounds can not cross it. Looking at first plot, it can be seen that below 500 Da most of the compounds are BBB+ compound and then the probability of passing is higher, and above that threshold most of the compounds are BBB-, and then they do not cross the BBB. It is expected since the small molecules are much

more capable of passing the BBB [Pardridge, 2005] than "heavy" molecules. However this is an inaccurate perspective since only 5% of molecules can cross the barrier, that is what plot B intends to show. In fact, before the prior probabilities, it can be verified that even lower than 500 Da most of the small molecules can not cross the barrier, and those that get through are a small part.

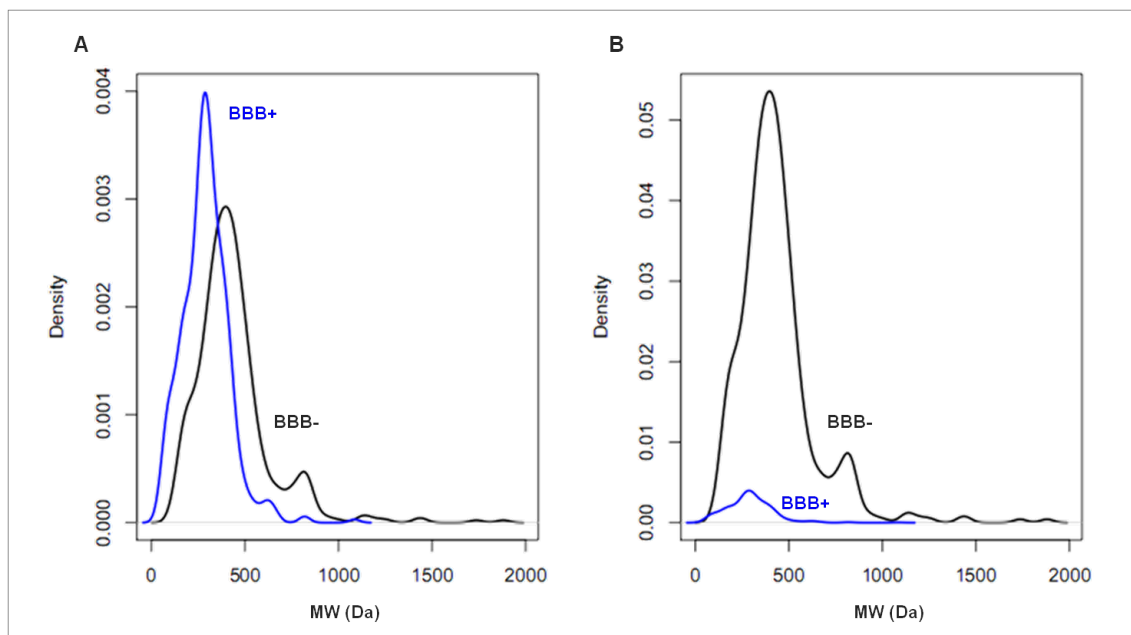


Figure 2.4: Estimated density of molecular weight (MW) for BBB+ compounds (blue) and BBB- compounds (black): A) considering the same ratio for both classes of compounds, B) considering prior probabilities (95% BBB- and 5% BBB+).

## 2.3 Algorithms for *in silico* BBB permeability

This section attempts to present some of the previous work that contributed to this research area: the prediction of BBB permeability of drug molecules. Several computational methods have been developed for the prediction of the BBB-penetrating (BBB+) and -nonpenetrating (BBB-) agents to use as pre-screening tools to reduce the cost and enhance the speed of the analysis. Table 2.1 summarises some of the previous work, pointing out the relevant aspects for terms of comparison with our study case. For each of the works was retained, when available, the number of descriptors, methods (being presented for the most of the cases the method that contribute for the best results), number of molecules for the training and test sets, as well as available classification statistics.

Table 2.1: Prediction accuracies and phi for BBB+ and BBB- compounds from different studies reported in the literature.

Year	Study	Desc <sup>c</sup>	Methods <sup>a</sup>	no. molecules (p/n)		Accuracy (%)			Phi
				Training	Test	BBB+	BBB-	Overall	
1999	Ajay	173	BNN	9000	139/136	92	71	81.8	-
2000	Crivori	72	PCA	46/64	49/71	90	65	74.8	-
2000	Cruciani	-	PCA	46/64	35	-	-	>75	-
2001	Trotter	72	SVM	81/91	256/48	78.9	60.4	76	-
2002	Doniger	9	SVM	154/120	25/25	82.7	80.2	81.5	-
2004	Adenot	67	PCA, PLS	1336/360	20/62	90	92	91	-
2005	Hu Li	199	SVM	276/139	-	88.6	75.0	83.7	0.645
2007	Zhao	19	R, PLS	832/261	451/49	98.2	87.8	97.2	-
2007	Zhao	19	R, PLS	1283/310	267/130	80.1	63.1	74.6	-
2008	Zhang	832 <sup>b</sup>	k-NN, SVM	124/20	99+267	-	-	82.5-100	-

<sup>a</sup> BNN(Bayesian neural network), PCA (principal component analysis), SVM (support vector machine), PLS (partial least squares), k-NN (k-Nearest neighbors), R (recursive partitioning models). <sup>b</sup> Descriptors are divided into three classes and are tested separately. <sup>c</sup> Number of descriptors

Ajay and Murcko [1999] used a Bayesian neural network (BNN) to develop a model that can distinguish between molecules that are CNS-active and CNS-inactive. The model was conducted using 173 descriptors, of them 7 are non-structural (such as MW, number of donors, number of acceptors) and 166 are structural (for example, the presence or absence of certain functional groups), and it was tested with an independent validation set of 275 molecules of known CNS activity. The cross-validation results show an overall accuracy of 81.8%, and 92% and 71% accuracies for CNS-active and CNS-inactive molecules, respectively. It should be noted that this study is based on the prediction of CNS-active and CNS-inactive, whereas in other studies, including the presented in this thesis are based on BBB-permeability and BBB-nonpermeability. The difference, although slight, exists since a molecule that is CNS-active is obliged to pass the BBB, but a CNS-inactive molecule may or may not pass the BBB.

PCA (principal component analysis) and discriminant PLS (partial least squares) were used by Crivori et al. [2000] to correlate data and build a model to access the BBB permeability of drug molecules. 72 simpler molecular descriptors were calculated from 3D molecular interaction fields with recourse to VolSurf program, some of them refer to molecular size and shape, to size and shape of hydrophobic and hydrophilic regions, hydrogen bonding, and others. The model was trained with 110 models and tested through cross-validation with an independent set of 120 molecules, and the results show an overall accuracy of 74.8%, and 90% and 65% accuracies for BBB+ and BBB- molecules, respectively. This work serves as base for Cruciani et al. [2000] that used the same training set of the previous work ([Crivori et al., 2000]) to predict the BBB penetration, and the same program (VolSurf) to calculate the molecular descriptors. The model was built



based on PCA analysis and tested through cross-validation with an independent set of 35 molecules. Results show values above 75% for the overall accuracy, the other measures are not specified.

Trotter et al. [2001] used 172 molecules, 72 molecular descriptors and SVM to build a model. It was tested with 304 molecules, and the results show an overall accuracy of 76%, and 78.9% and 60.4% accuracy for BBB+ and BBB- molecules, respectively.

Neural networks (NN) and support vector machine (SVM) were tested by Doniger et al. [2000] to distinguish molecules capable of crossing the BBB from those that can not cross it. The model was trained with 324 molecules and 9 molecular descriptors, namely logP, MW, volume, surface area, percent of hydrophilic surface area, hydrogen bond donors/acceptors, and 3D hydrogen bonding. To validate the model, 30 different validation sets of 50 molecules were constituted through a "bootstrapping approach". According to this approach, molecules are selected at random from the complete dataset. The best results were obtained with SVM: 82.7%, 80.2% and 81.5% for BBB+, BBB- and overall accuracies, respectively.

PCA and PLS were also used by Adenot and Lahana [2004] to build a model. To train the model, about 1696 molecules and 67 descriptors were used. Some of the descriptors are number of atoms, number of heteroatoms, MW, molecular volume, polar surface area, and others. The model was tested through cross-validation with a validation set of 82 molecules, and the results show an overall accuracy of 91%, and 90% and 92% accuracy for BBB+ and BBB- molecules, respectively.

Li et al. [2005b] perform several models in their study: LR (logistic regression), LDA (linear discriminate analysis), C4.5 DT (C4.5 decision tree), k-NN (k-nearest neighbor), PNN (probabilistic neural network) and SVM. It was used 415 molecules and 199 descriptors of the next classes: simpler molecular properties, molecular connectivity and shape, electrotopological state, quantum chemical properties and geometrical properties, and it was performed a 5-fold cross validation. Also in this study, the highest values were obtained with SVM models with an overall accuracy of 83.7%, and 88.6% and 75% accuracies for BBB+ and BBB- molecules, respectively.

Zhao et al. [2007] used recursive partitioning and PLS, and two different training sets to build models. These authors used 19 descriptors namely, excess molar refraction, overall hydrogen-bond acidity/basicity, MW, polar surface area, logP, number of hydrogen bonding donors/acceptors, and others. The first training set with 1093 molecules was tested through a cross-validation with independent validation set of 500 molecules, and the results show an overall accuracy of 97.2%, and 87.8% and 97.2% accuracies for BBB+ and BBB- molecules, respectively. The second training set with 1593 molecules was cross-validate with an independent validation set of 397 molecules, and the results show an overall accuracy of 74.6%, and 80.1% and 63.1% accuracies for BBB+ and BBB- molecules, respectively.

Finally, Zhang et al. [2008] developed two methodologies including k-NN and SVM with 144 molecules. It was also applied an applicability domain (AD) that defines the area of the descriptors space in which models can accurately predict the target properties. Considering that, some of the molecules, that are not included in the chemical space of the modeling set, could not be predicted its classification. To test the models two independent validation sets, with 99 and 267 molecules were used. Without applying the AD, the test set achieves an overall accuracy of 82.5% and 59.0%. After applying the AD, the overall accuracy increases to 100% and 83.3%, however there is a reduction of the number of molecules for which the prediction could be made, mainly for the subset of BBB- molecules.

The implementation of an SVM methodology seems to be generalized in most of the studies described and most of the times the highest values of accuracy are associated with it. Although the comparison of results could be wrong since datasets are usually different. It is noteworthy that none of the studies take into account the imbalanced ratio between both classes of molecules.

# Chapter 3

## Methods

In this chapter, the fundamentals of molecular representation and encoding are presented, as well as the algorithms and statistical methods used. In Section 3.2 is presented the molecular representation used to encode molecules structure and all the molecular descriptors used to retain properties about molecules. Section 3.3 introduces the idea of similarity between molecules, presenting some of the coefficients used to measure this similarity. Section 3.1 makes a brief introduction to the tools used in this work to code and implement approaches. Moving to methods part, in Section 3.4 are presented the statistical methods and measures used to assess performance of the approaches. Finally, Section 3.5 comprises the three machine learning algorithms used to build the three approaches presented in this work: k-Nearest Neighbor (k-NN), Neural Networks (NN) and Random Forests (RF).

### 3.1 Tools

#### 3.1.1 R

R is an open source programming language and software environment for statistical computing and graphics, and it can be run in Windows. It is widely used for data analysis, and it has the advantage of being highly extensible, easy and offers a wide variety of functions. For this work it was installed R version 2.10.1. Principal Coordinates Analysis (PCoA), k-NN, NN and RF models were coded using available implementation (packages) of R.

#### 3.1.2 Python

Python is a high-level programming language [Python, 23-10-2010], often used as a scripting language. It is free to use, with an open source license. For this work, it was used Python version 2.7 to code and obtain SMILES representation of molecules of constituted dataset, similarity matrices and molecular descriptors.

Python is widely used for bioinformatics and cheminformatics purposes [O’Boyle

et al., 2008], due to its simple syntax and code readability, that make it an easy programming language to learn, compared with other programming languages.

### 3.1.3 OpenBabel

OpenBabel is an open-source C++ toolkit with interfaces in Python. Scripting languages like Perl, Python and Ruby are ideally to solve common cheminformatics tasks, although they can be slower than compiled languages. For this reason, cheminformatics toolkits are often implemented in compiled languages, as C++ [O’Boyle et al., 2008]. This toolkit can deal with many molecular file formats containing thousands of molecules, manipulate molecular data, supports molecular fingerprints, calculus of molecular descriptors, as LogP, iteration over molecules and many others functions. OpenBabel has interfaces to other programming languages (Perl, Ruby and Java), however the Python interface seems to be the most used [O’Boyle et al., 2008]. It can be accessed from two Python modules: (1) the standard Python bindings and (2) the Pybel.

1. Python bindings are generated using SWIG (Simplified Wrapper and Interface Generator) [Beazley, 1996]. SWIG is a software development tool that connects programs written in C and C++ with a variety of high-level programming language. In this way, it allows to use scripting languages with their C/C++ programs without worrying about the underlying implementations details of each language. The file generated by SWIG is then compiled and linked with the Python development libraries and OpenBabel, creating a Python extension module, *openbabel*. The bindings provide access to the majority of the OpenBabel interfaces, namely the base classes as OBMol, OBAtom, OBBond, and OBResidue.
2. Pybel module wraps the bindings automatically generated by SWIG to give a more Pythonic<sup>3</sup> way to access OpenBabel [O’Boyle et al., 2008]. This module makes easier to access features of the OpenBabel libraries from Python and a good feature is the possibility of using both modules at the same time, by converting Atom and Molecule classes of Pybel to and from the OBAtom and OBMol of *openbabel* module, even because Pybel does not have all of the methods present in OpenBabel library.

Each module (*openbabel* and *pybel*) can be imported into a Python script using the *import openbabel* and *import pybel* statement, respectively.

Reading and writing molecule files is one of the tasks where using Pybel is advantageous. Pybel makes use of iterators from Python to iterate over collections of objects,

---

<sup>3</sup>It refers to a neologism: use Python language conventions and idioms, everything in the smallest number of code lines.

which simplifies a lot the processes and in a few lines of code we can read and write molecule files, iterate over each atom in a molecule and some other features.

To use OpenBabel and Pybel it is necessary to download and install the binary installer for Windows (OpenBabelGUI, version 2.3.0), which includes command-line tools and a graphical user interface (GUI). Only then it is possible to install the OpenBabel Python bindings, version 1.6, to Python 2.5, 2.6, 2.7 and 3.1. With OpenBabel library, custom scripts can be coded without using a GUI or the command-line.

Some of the scripts were tested with Python version 3.1 and it was verified that some features of OpenBabel did not work.

## 3.2 Molecular representation

### 3.2.1 SMILES - Simplified Molecular Input Line Entry Specification

Brown [2009] refers that despite their apparent simplicity, "molecules are complicated real-world objects". SMILES (Simplified Molecular Input Line Entry Specification) language is a specification for describing the structure of chemical molecules using short ASCII strings and it contains the same informations that can be found in other representations (as in extended connection table, InChI, InChIKey) however it is more human-readable and quite compact than the others. With this representation, molecules could be encoded by more than one SMILES string, so it could not be used as a unique identifier of a molecule structure. Nevertheless, in 1960s was proposed an algorithm that overcome this feature and provides a canonical ordering of atoms in a molecule.

#### Canonicalization

The canonicalization generates one generic SMILES among all equally valid SMILES, known as the "unique SMILES". Canonical SMILES are unique for each structure but it depends of the canonicalization algorithm used to generate it. This type of SMILES are generated regardless of the order of the atoms in the structure. Generic SMILES termed refers to labeled molecular graph (atoms and bonds). SMILES that also describe isotopic and chiral information are known as "isomeric SMILES", and an "absolute SMILES" represent a unique isomeric SMILES.

#### SMILES Specification Rules

There are rules that guide SMILES representation, corresponding to specification of atoms, bonds, rings, branches, aromaticity, and disconnected structures [OpenSMILES, 12-05-2011, Weininger, 1988, Brown, 2009].

**Atoms** - are represented by their atomic symbol, in square brackets. The symbol has the same representation as in Periodic Table: first character is uppercase and the

second, if exist, is lowercase. This rule is not applied if the atom is aromatic, for this case, the first character is lowercase. Some elements, B, C, N, O, P, S, F, Cl, Br and I are grouped in an "organic subset", and do not require brackets. When represented without brackets, this elements have the following properties: number of attached hydrogens conforms to the lowest normal valence is consistent with explicit bonds, the atom's charge is zero, and the atom has no isotopic and chiral specification. The atoms inside brackets must have hydrogens and charge specified. The charge is specified by a "+n" or "-n", where n is a number.

**Bonds** - adjacent atoms are assumed to have single or aromatic bonds, other type of bond has to be specified. Single, double, triple and aromatic bonds are represented by '-', '=', '#' and ':' respectively.

**Rings** - the first and the last atoms of a ring are labeled with numeric suffix to indicate connectivity between non-adjacent atoms. If there is more than one ring in the molecule structure, the label will be 2, and so on.

**Branches** - an atom with three or more bonds is called a branched atom, and is described using parentheses.

**Aromaticity** - can be represented in the Kekulé form using alternating single and double bonds with atoms in uppercase, or using the "aromatic atoms" (in lowercase), with no need of bond symbols.

**Disconnected Structures** - the '.' (dot) symbol is used to indicate not bonded atoms, and represent disconnected and ionic compounds.

### Isomeric SMILES

Describes rules to specify stereochemistry, which covers the cis/trans configuration around a double bond and the chiral configuration of specific atoms in a molecule.

- The cis/trans configuration around double bonds is specified using the symbols "\" and "/"; these symbols appear in pairs indicating relative directionality between the connected atoms - "same side" (cis) or "opposite side" (trans).
- The configuration at tetrahedral carbon is specified by '@' or '@@'. Considering the chiral center, the remaining three atoms are written from left to right, that means, in anticlockwise order using the '@' symbol. To describe atoms in a clockwise order, then '@@' symbols are used.

### 3.2.2 Molecular descriptors

Molecular descriptors have been used as input for statistical studies as a source to make assumptions in very different studies that treat the permeability of BBB to drug molecules [Li et al., 2005a]. They aim to be descriptions of molecules that capture relevant aspects of them, mainly quantitative descriptions of structural and physicochemical properties of molecules.

In this work only structural properties of molecules (physicochemical properties) were used as molecular descriptors. This class of descriptors provides values of physical properties of molecules [Brown, 2009]. Below are presented all the molecular descriptors used in the context of this work.

**Molecular Weight (MW)** - represent a summation according to the number and type of atoms that constitute the molecule.

**Average Molecular Weight (AMW)** - is a variation of MW, representing the mean of MW by the total number of atoms present in the molecule.

**LogP** - represent the octanol-water partition coefficient and it is widely used in quantitative structure-activity relationship (QSAR) methods as a descriptor of cell permeability, providing a measure of solubility or lipophilicity of a molecule [Wildman and Crippen, 1999], therefore it is an important parameter in the study of new drug molecules [Brown, 2009]. Nevertheless, its measurements are time-consuming, expensive and difficult to get. To overcome these difficulties were developed methods to calculate logP based on molecular structure. There are many different approaches to perform this calculations but it can be highlighted two main approaches, regarding to fragments methods and atom-based approach. LogP obtained through these methods is often called ClogP [Wildman and Crippen, 1999, Daylight, 2-10-2011].

$$\log P = \log\left(\frac{[solute]_{octanol}}{[solute]_{water}^{un-ionized}}\right) \quad (3.1)$$

**Number of Rings (NRing)** - represent the number of rings in a given molecule.

**Number of Bonds in Rings (NBRing)** - represent the number of bonds in a ring, ignoring the type of bond.

**Multiplicity** - for each of the following atoms, C, N, O, S, Br, I, Cl, F, and Na it was considered the number of possible bonds that atoms can do, ignoring possible hydrogen bonds. Considering the carbon atom for example we could have: C-1, C-2, C-3, C-4, that means, carbon bonded to one substituent atom<sup>1</sup> and bonded to three

---

<sup>1</sup>For the context of multiplicity, a substituent represent an atom or group of atoms substituted in place of a hydrogen on the considered atom.

hydrogens, carbon bonded to two substituent atoms and bonded to two hydrogens and so on. For each atom are considered all possible multiplicities, and for each of the multiplicities are counted the number of occurrences in a molecule. So, we can consider each multiplicity as a molecular descriptor (table 3.1).

**Fingerprints** – is a vector representation of molecules, in each molecule is encoded in a binary string that each bit represents the presence or absence of a substructure. Each bit of fingerprints, in a total of 1023, will be considered a molecular descriptor in order to represent the presence of a particular *piece* in molecules, and it will retain the number of occurrences of the substructure encoded in the bit. Due to the complexity of fingerprints, the next section is dedicated to present fingerprints and their characteristics.

Table 3.1: Multiplicities used as molecular descriptors.

Element	Atomic number-number of bonds
Carbon (C)	6-1, 6-2, 6-3, 6-4
Nitrogen (N)	7-1, 7-2, 7-3, 7-4
Oxygen (O)	8-0, 8-1, 8-2
Fluorine (F)	9-1
Sodium (Na)	11-0
Phosphorus (P)	15-4
Sulfur (S)	16-1, 16-2, 16-3, 16-4
Chlorine (Cl)	17-0, 17-1
Calcium (Ca)	20-0
Bromine (Br)	35-0, 35-1
Iodine (I)	53-1

### 3.2.3 Fingerprints

Molecular fingerprints are used to encode molecular structure in a series of binary digits (bits) that represent the presence or absence of particular substructures in the molecule. Fingerprints can be divided into two types: structure-key and hash-key fingerprints.

#### Structure-Key Fingerprints

The structure-key fingerprint is based on a dictionary of defined substructures to generate a binary (0 or 1) string where each bit equates one substructure in the dictionary. So, in this kind of fingerprints an on-bit (1) in seventh fingerprint position (fp<sup>2</sup>) corresponds to a defined substructure in the dictionary.

<sup>2</sup>Abbreviation to refer to fingerprint position will be done in lowercase to distinguish from FP of False Positives.



This type of fingerprints was the start point to a new generation of fingerprints, the hash-key fingerprints, since the oldest ones have several disadvantages. Some of them are:

- Lack of generality. It is necessary to choose in advance the patterns to be included in the "dictionary" of keys, it reflects the nature of the research area in which dictionary will be applied. So, because of that, it can hardly be used in other types of research areas.
- Mostly zeros. As the dictionary of substructure keys are build by users, a typical molecule will contain very few of the patterns represented by the structural key's bits.
- Patterns are not overlapped. Although a molecule shares portions with other patterns, this type of fingerprints does not consider this. So, more complex molecules are not accurately characterized.

### Hash-Key Fingerprints

In hash-key fingerprints there are no predefined database of substructures, instead the keys are generated based simply on the molecule. In this type of fingerprints, there is no assigned meaning to each bit. This fingerprint enumerates multiple atom-bond paths from each molecule, generally with a path length of 0 to 7.

With hashing method, the same bit can map several substructures, because of that a match does not guarantee the presence of a particular substructure, on the contrary if a fingerprint indicates the missing of a substructure then it certainly is.

Pybel provides four types of fingerprints, FP2 corresponds to a hash-fingerprint definition and it is the most common type [Babel, 4-11-2010].

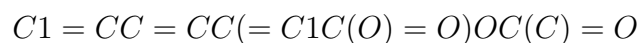
The FP2 fingerprint type generates bits from 0 to 1024. For each molecule, pybel returns an output record (figure 3.1) [dalke scientific, 6-7-2011].

```
>CCCCC
00000000 01000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 40000000
00000000 00000000 00000000 00000000 00000000 00000000
00002000 00000001 00000000 00000000 00000000 00000010
00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000
```

Figure 3.1: Pybel representation of bits for a FP2 fingerprint.

This output contains a fingerprint with 6 groups of 5 lines, with exception for two of the groups that have two more lines. Each group has 8 hexadecimal values and each of these are 4 bits, so in total we should have 1024 bits for each fingerprint  $((5 \times 6 + 2) \times 8 \times 4 = 1024)$ .

Open Babel GUI gives a list of the chemical fragments used to set bits [Babel, 21-09-2010]. Below is an example of an output showing the on-bits and respective fragments for acetylsalicylic acid (commonly named Aspirin) represented by



```
0 6 1 6 <670>
0 6 1 6 5 6 1 8 1 6 1 6 <463>
0 6 1 8 1 6 <105>
0 6 1 8 1 6 1 6 <655>
0 6 1 8 1 6 5 6 1 6 <104>
0 6 5 6 <81>
0 6 5 6 1 6 <473>
0 6 5 6 1 8 1 6 <568>
0 6 5 6 1 8 1 6 1 6 <1018>
...
```

The first digit of each line of the output indicates whether the fragment is linear (0) or cyclic (1) and the remaining digits indicate the atomic number and bond order alternatively. For example, the first bit of the output (670) represent a carbon (atomic number 6) bonded to another carbon by a single bond (1). If we have digit 2 instead of 1, it means that carbons are bonded with a double bond, and so on.

### 3.3 Molecular similarity

Most of the times, molecular similarity is calculated with a coefficient that compares two molecular fingerprints. Once we got the fingerprints representation of a molecule it is easy to calculate a measure of similarity between two molecules. A similarity measure based on fingerprints can be seen as a mathematical approach, in which on-bits in the fingerprints are compared between two molecules. Molecules are described by the presence or absence of substructures, so four terms a, b, c, and d are considered, which are represented in a two-by-two contingency table (Table 3.2).

Where:

- a - count of bits on in object A but not in object B
- b - count of bits on in object B but not in object A
- c - count of bits on in both object A and object B
- d - count of bits off in both object A and object B
- n - total number of bits (a + b + c + d)

Table 3.2: Contingency table representing the four terms that describe presence and absence of features between two molecules.

		Molecule B		
		0	1	Total
Molecule A	0	d	b	d + b
	1	a	c	a + c
	Total	a + d	b + c	n

A number of distance metrics can be calculated for binary fingerprints. All the measures applied in this work are actually similarity measures and thus represent the reverse of a distance metric and so the distance can be obtained by subtracting the value from 1.0.

Similarity measures are often classified into two categories:

**measures of co-occurrence** - range from 0 to 1 and their numerator usually consists of a + d.

**measures of association** - range from -1 to 1 and their numerator usually contain ad-bc.

### Similarity measures

Next we define all the measures [Daylight, 3-12-2010] used in this work:

**Cosine** is the ratio of the bits in common to the geometric mean of the number of on bits in the two molecules.

$$\text{Cosine} = \frac{c}{\sqrt{(a + c)(b + c)}} \quad (3.2)$$

**Dice** is the ratio of the bits in common to the arithmetic mean of the number of on bits in the two molecules.

$$\text{Dice} = \frac{2 \cdot c}{a + b + 2 \cdot c} \quad (3.3)$$

**Euclidean** is function of d, so it is dependent on the double zeros.

$$\text{Euclidean} = \sqrt{\frac{c + d}{a + b + c + d}} \quad (3.4)$$

### Forbes

$$\text{Forbes} = \frac{c(a + b + c + d)}{(a + c)(b + c)} \quad (3.5)$$

### Hamman

$$\text{Hamman} = \frac{(c + d) - (a + b)}{a + b + c + d} \quad (3.6)$$

**Kulczynski** represent the mean of the individual substructures similarities.

$$Kulczynski = 0.5\left(\frac{c}{a+c} + \frac{c}{b+c}\right) \quad (3.7)$$

**Manhattan**

$$Manhattan = \frac{a+b}{a+b+c+d} \quad (3.8)$$

**Matching**

$$Matching = \frac{c+d}{a+b+c+d} \quad (3.9)$$

**Pearson**

$$Pearson = \frac{c.d - a.b}{\sqrt{(a+c) * (b+c) * (a+d) * (b+d)}} \quad (3.10)$$

**Rogers-Tanimoto**

$$Rogers - Tanimoto = \frac{c+d}{2.a + 2.b + c + d} \quad (3.11)$$

**Russel-Rao**

$$Russel - Rao = \frac{c}{a+b+c+d} \quad (3.12)$$

**Simpson** is the best of the individual substructures similarities.

$$Simpson = \frac{c}{\min((a+c), (b+c))} \quad (3.13)$$

**Tanimoto** is independent of d. It represents the proportion of the on-bits shared.

$$Tanimoto = \frac{c}{a+b+c} \quad (3.14)$$

**Yule**

$$Yule = \frac{c.d - a.b}{c.d + a.b} \quad (3.15)$$

## 3.4 Statistical methods

### 3.4.1 Principal Coordinates Analysis

Principal Coordinates Analysis (PCoA), also known as Classical Multidimensional Scaling (CMDS), is a statistical technique that provides a graphical representation of the pattern of proximities in a given set of objects.

PCoA is an unsupervised statistical learning method and it can be seen as a clustering method that determines natural groupings of objects based solely on their independent variables, such as molecular descriptors. The supervised statistical learning methods use

a different approach, that consists in using a priori information regarding the classes to which the objects in a training set belongs, as k-NN and NN.

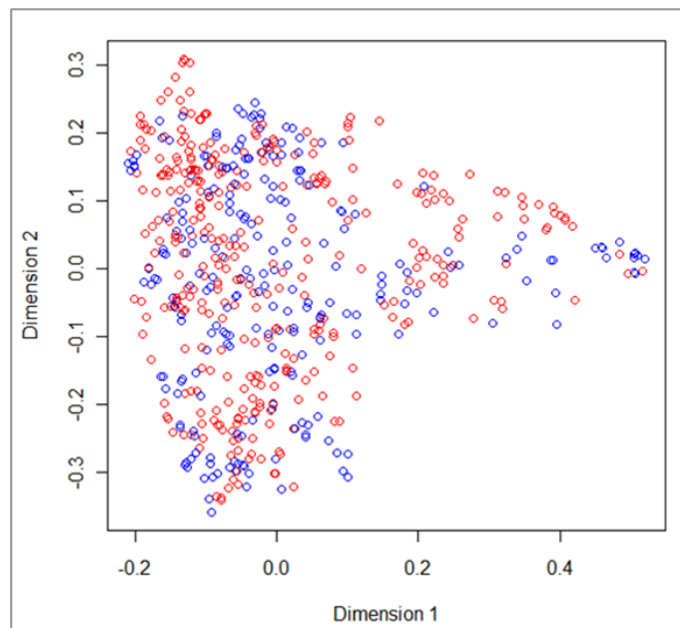


Figure 3.2: Molecules representation in a two dimensional space, after applying `cmdscale` function, considering Tanimoto measures. The red and blue circles represent molecules able to cross the barrier and not able to cross it, respectively.

PCoA objects are represented as points in a low dimensional (typically two or three) Euclidean space, where distances between the points match the original distances as well as possible. This technique is based on an eigenvalue equation where a symmetric matrix  $M$  of  $n$  objects ( $n \times n$ ) with a  $(i, j)$ th entry,  $d_{ij}$ , gives a measure of the relationship between the  $i$ th and  $j$ th objects. The basic idea is that the symmetric matrix contains distance informations and the goal is to obtain a set of  $n$  points in  $p$  dimensions whose interdistances approximate the matrix. Usually, PCoA is used to represent a set of objects in a two-dimensional space, however a low number of dimensions may prove to be insufficient to properly represent data. In this case, the number of dimensions should be increased. The result is useless if the only purpose is to have a visual representation of data. Although, PCoA can be used as a dimension reduction method, transforming the original matrix into a simple and more small distance matrix. A simplified view of the method is as follow:

1. Assign points to arbitrary coordinates in a  $n$ -dimensional space;
2. Compute euclidean distances among all pairs of points;
3. Adjust coordinates of each point to get the best representation.

Principal Coordinates Analysis can be done in R by using the `cmdscale` function [CRAN-R, 29-10-2010]. `cmdscale()` takes a  $n \times n$  distance matrix  $M$ , and returns a  $n$

$n \times p$  matrix  $Y$ , in which a row represents the coordinates of  $n$  points in  $p$ -dimensional space for some  $p < n$ . The input has to be square and symmetric indicating relationships between objects, in our case, between molecules.

This technique gives, for each molecule, a vector of coordinates, allowing their simple representation in space, that transmits similarities between molecules.

For our work, we are mainly interested in the classification of molecules in their ability to cross the BBB. The original symmetrical matrix contains distances between molecules, according to its structural similarity.

Figure 3.2 shows the molecules representation, in a two-dimensional space, where red circles are molecules that can penetrate the BBB and blue circles molecules that can not penetrate the BBB. Of the figure, some little clusters of just one color can be identified, meaning that molecules with a classification tend to be closer of molecules with the same classification.

### 3.4.2 Statistical measures

In order to classify the results obtained from the algorithm's analysis various statistical measures are available [Carugo, 2007, Hripcsak and Rothschild, 2005, Baldi et al., 2000].

The degree of reliability of each prediction was estimated with the confusion matrix. The matrix represents a classification of features in positive and negative. In our work, the ability to cross the BBB is considered as the positive feature, so we refer them as "positive molecules"; on the other hand, the absence of the ability to cross the BBB is considered the negative feature, so molecules are referred as "negative molecules".

From the confusion matrix we can define the four following quantities:

True Positives (TP) = number of positive molecules that are correctly predicted;

True Negatives (TN) = number of negative molecules that are correctly predicted;

False Positives (FP) = number of negative molecules that are (incorrectly) predicted to be positive; and

False Negative (FN) = number of molecules that are predicted to be negative despite they are positive.

The matrix is formed by two rows and two columns that report the number of the four quantities previously explained. Each column of the matrix represents the instances in the predicted class, while each row represents the instances observed in the class (Table 3.3).

Table 3.3: A confusion matrix.

		Real	
		+	-
Predicted	+	TP	FP
	-	FN	TN

### Precision

Precision is defined as the proportion of the true positives against all the positive results. It gives a rate of the "positive molecules" (TP) that are correctly classified among the total of molecules classified as "positive molecules" (the sum of TP and FP).

$$Precision = \frac{TP}{TP + FP} \quad (3.16)$$

### Recall

Recall, or sensitivity, is a measure of the ability of a system to recognize all positive cases, it is defined as the number of TP divided by the total number of elements that actually belong to the positive class (the sum of TP and FN). This measure gives us a proportion of "positive molecules", but, unlike precision, it is according to the total of molecules that really are positive.

$$Recall = \frac{TP}{TP + FN} \quad (3.17)$$

Precision and Recall are widely used for evaluate the correctness of a pattern recognition algorithm. Precision can be seen as a measure of exactness or fidelity, whereas recall is a measure of completeness. Both of them may range from 0 to 1, the latter value being associated with perfect predictions.

### Accuracy

Accuracy computes the fraction of instances for which the correct result is returned, that means, the proportion of true results (both TP and TN). Its values range between 0 and 1, where 1 means that the predicted values are exactly the same as the given values. With this measure both "positive molecules" and "negative molecules" are considered, in the global of possible labels ("positive" and "negative molecules" correct or incorrect classified).

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad (3.18)$$

The measures explained so far do not take the true negative rate into account, so it is necessary to resort to other measures to assess the performance of a binary classifier. Moreover, none of the measures presented previously can alone describe perfectly a confusion matrix. Even giving a measure of correctness values for both classes (positive and negative), accuracy can sometimes fail in the true quality of a system, mainly when classes are unbalanced. This lead us to another measure: the phi measure.

## Phi

Phi, or Matthews Correlation Coefficient (MCC), is a correlation coefficient between the observed and predicted binary classifications. It takes into account true and false positives and negatives and is generally regarded as the more faithfully measure which can be used even if the classes are unbalanced. Phi measure is generally regarded as being one of the best measures to describe the confusion matrix, and consequently is widely used in machine learning algorithms as a measure of the quality of binary classifications. This measure returns values in the interval  $[-1,1]$ , where -1 means that algorithm fails all predictions, so is just need to change our interpretation to have a perfect predictor, 0 a random prediction and 1 a perfect prediction.

$$Phi = \frac{TP.TN - FP.FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \quad (3.19)$$

## BBB+ and BBB-

Defined by us, BBB+ represents the positive predictive rate, while BBB- represents negative predictive rate.

$$BBB+ = \frac{TP}{TP + FP} = Precision \quad (3.20)$$

$$BBB- = \frac{TN}{TN + FP} \quad (3.21)$$

### 3.4.3 Cross-Validation

Cross-Validation (CV) is a statistical technique to evaluate and compare learning algorithms. It is widely used in predictive purposes, in order to assess performance of the model. The idea behind CV is to split data into two subsets, one to train the model and another one to test it. Then, successive rounds are performed using different subsets for train and test, and at the end of the performance results are averaged over the rounds. The performance is tracked using statistical measures, as accuracy. There are many types of CV, in this document is only considered two of them: leave-one-out cross-validation (LOOCV) and k-fold cross-validation.

To perform a k-fold cross-validation the initial dataset is divided into k equally sized subsets. In each iteration (k), k-1 subsets are chosen to train the model, and subsequently the trained model will make predictions in the remaining subset used to test the model. In this way, each of the subsets are crossed against the others. The present work comprise a binary classification, therefore at the time of split data in k subsets it is common that each of the subsets have around half the instances, in this way is ensured a good representativity



of the dataset. Most of the times, the number of  $k$  equals 10, this value is chosen as being a good compromise [Refaeilzadeh et al., 2009].

LOOCV works as a  $k$ -fold cross validation, where  $k$  equals the size of the dataset. In each iteration ( $k$ ), each instance of dataset is used to test the model, and the remaining instances to train it, that means that each instance is successively "left out" from the dataset and used for validation. This type of CV is time-consuming when the number of instances in dataset is high, due to the number of times that model would be trained.

## 3.5 Learning methods

### 3.5.1 k-Nearest Neighbor

k-Nearest Neighbor algorithm is a simple classification algorithm. It uses a space representation of instances to classify other instances of unknown classes, so no model or learning is performed. In k-NN, it can be identified two phases: the training and testing phases. The training phase requires a training set used to populate a sample of the search space with instances labeled with a known class. Based on the training phase, the testing phase is performed, where it is possible to make class predictions for unknown instances. k-NN measures the Euclidean distances between instances to be classified and each instance of the training set, expressed in the following equation:

$$D = \sqrt{\|x - x_i\|^2} \quad (3.22)$$

After computing the distances measures for all possible pairs, k-NN sorts the distances in increasing numerical order and "saves" the  $k$  (number of neighbors) elements. Next, the most frequent class is returned.

The following example is an illustration of the issues here discussed. In Figure 3.3 the red circles represents molecules that can not cross BBB (BBB-), blue circles molecules that can cross the barrier (BBB+) and green circle is a molecule of unknown class that we want to know the classification based on their neighbor molecules. So, considering the three closest neighbor molecules, the "green molecule" will be assigned as passing the BBB, because there are more "blue molecules" than red; but, considering five closest neighbor molecules, the "green molecule" will be classified as not passing the BBB because there are more "red molecules".

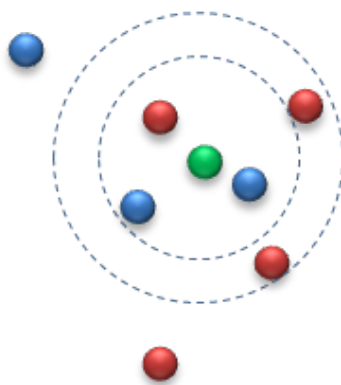


Figure 3.3: Two-dimensional representation of molecules for classification using k-NN algorithm.

The choice of the neighbor number depends upon the data, it is necessary to search and choose the value that better fits to the data.

During an article reading, I found a phrase that perfectly depicts our approach: "The k-NN method works in line with the central premise of medicinal chemistry in that structurally similar molecules have similar biological activities" [Dmitry A. Konovalov and Heyden, 2007].

### 3.5.2 Neural Networks

A Neural Network (NN) is a computational structure composed of processing units (neurons) that are connected by links and organized in layers. Neurons communicate with each other sending signals through connections and information processing is made as occur simultaneously in all neurons. Each neuron has input and output connections, and each connection has a weight associated with it [Santos and Azevedo, 2005, Gentleman, 2009]. Then NN is a non-linear statistical data modeling used for classifications tasks, by finding patterns and unlock relationships between input information and output. It makes use of interconnected artificial neurons to process information that changes through an iterative process, where weights between neurons are successively corrected. NN are highly sensitive to the data and generally they have reduced ability to extrapolate beyond the limits of the input variables.

Each NN has a architecture that determines how neurons are organized. Therefore, three types can be distinguished:

- multi-layer network - consists of different parallel layers and corresponds to the most widely used NN. The first and last layer are designated by input layer and output layer, respectively and intermediate layers are designated hidden layers.

- totally connected network - each neuron is connected to all neurons of network.
- single layer network - has two layers, input and output layer. Neurons are independent of each other so they can be trained separately.

The architecture of a neural network determines how neurons are connected and they can be subdivided into:

- recurrent network - connections between neurons form a directed cycle, where neurons of the output layer can be connected to the neurons of the input layer.
- feed-forward network - connections are unidirectional (forward), so information moves from the input neurons through hidden neurons and to the output neurons.

In our study case, we are interested in prediction of permeability of drug molecules, based on the knowledge of the permeability of the molecules in training set. Then, NN has to be capable of learning, increasing knowledge through experience.

NN in which correct classification of instances of training set are known have a supervised learning. If the produced classification match prior classification, then no need to change the weights. Otherwise, the difference between the two classifications (value) is used to modify the connection weight in the network. The weights are adjusted incrementally until the training set matches the desired output as closely as possible to a maximum number of defined iterations.

The NN implementation used was the one provided by software R, namely package `nnet` [CRAN-R, 18-10-2010]. This package allows implementation of a feed-forward, multi-layer neural network, with a single hidden layer 3.4.

The `nnet` package implements a NN that has a input layer (Figure 3.4), one hidden layer and an output layer. Next it will be explain how to choose the number of neurons in each of the layers.

#### *Input layer*

This layer represents the variables used for the NN learning process. For example, if the molecular descriptors corresponds to coordinates vector from a PCoA analysis of a similarity matrix, then the number of neurons in the input layer is equal to the number of coordinates of the vector. `nnet()` automatically defines this from input data.

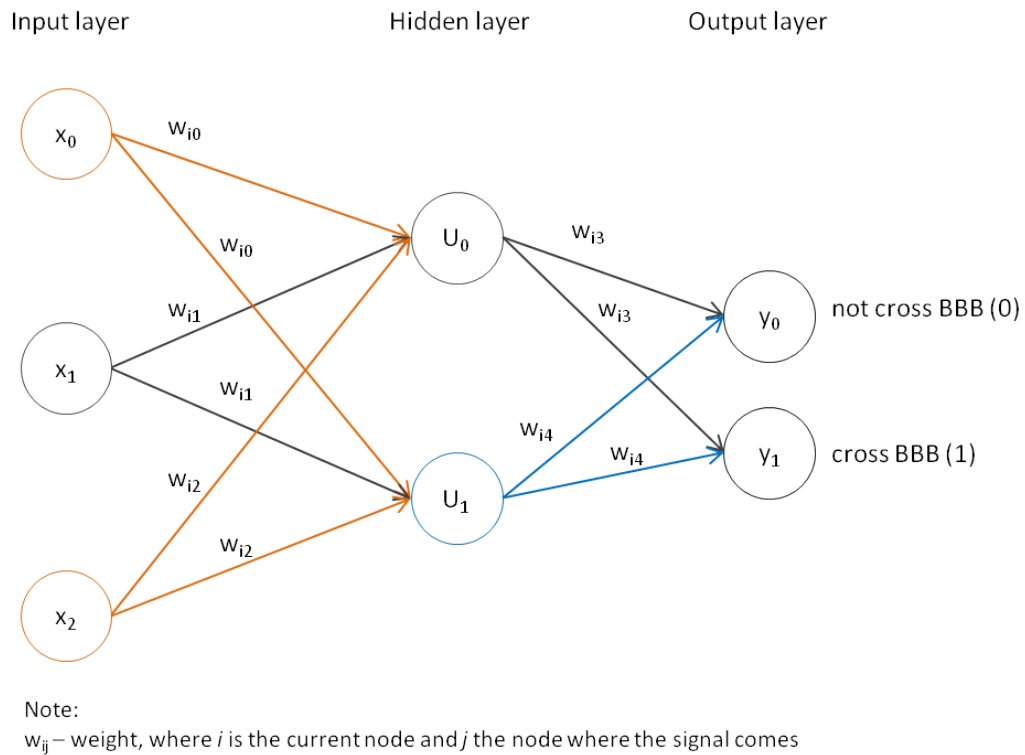


Figure 3.4: Feed-forward network with a single hidden layer. Adapted from Santos and Azevedo [2005]

### *Hidden layer*

For this layer, `nnet()` allows to choose the number of neurons. There are some empirically-derived rules of thumb, of these three are highlighted [Xu and Chen, 2008, Heaton, 2008]:

- The number of hidden neurons should be between the input layer size and the output layer size;
- The number of hidden neurons should never be more than twice as large as the input layer;
- The number of hidden neurons should be  $2/3$  the size of the input layer, plus the size of the output layer.

### *Output layer*

Molecules will be classified in passing or not passing the BBB (binary classification), so the number of neurons in the output layer will be two. This feature is also automatically defined by `nnet()`.

## **3.5.3 Random Forest**

The Random Forest (RF) algorithm [Breiman and Cutler, 18-5-2011] is based on the features of decision trees, but instead of having only one tree, there is a collection of decision trees. The algorithm grows many decision trees, in order to improve predictive accuracy. It classifies one case using each tree in the new "forest", and choose a final predicted outcome by combining the results across all trees using majority vote.

To understand how RF works, first it is necessary to introduce decision trees.

### **Decision Tree**

A Decision Tree (DT) represents a set of rules that follows a hierarchy of classes and values, used to classify instances. An instance is classified by starting to test the attribute specified by the root and then following the branch corresponding to the value of the attribute in the instance. This process is then repeated for the sub-tree with root on the new node. The decision tree algorithm grows recursively trees from a training set, subdividing this data set until the tree is just formed by "pure nodes", in that each node represents only a single class, or the satisfaction of a criterion.

A DT has a defined structure based in the conception of a real tree, which is: root, branch (values of the attributes/variables), internal nodes (the attributes/variables, with two or more sub-trees that represent different paths) and leafs or "pure nodes" (the classes).

There are two types of DT: classification and regression; however, we are just interested in classification DT. When a tree model is applied to a test set, each instance follows a certain path, until the instance reaches a leaf, which will be assigned with the class of the leaf.

DT has a mechanism of pruning, that plays an important role in producing smaller and more accurate trees with more potential to classify new instances. The main goal of pruning is removing parts of the tree that do not contribute to classification accuracy, producing less complex trees.

## Random Forest

In a Random Forest, a DT grows according to the next steps:

- Considering  $N$  cases in the training set, it is necessary to proceed to a random sample  $n$  with replacement. This sample will be the training set for growing the tree.
- A number of  $m$  variables in a total of  $M$  are randomly selected and of these the variable that better splits class instances <sup>4</sup> is chosen to split the node. The variable chosen at each node is held constant.
- Although pruning is a common practice in DT, in Random Forest this feature is not implemented.

In RF there is no need of cross-validation, the reason is that it performs a Out-Of-Bag (OOB) while is training. As RF uses bootstrap samples for construct trees, one-third of the cases of training set are left out and consequently they are not part of trees construction. This feature is very important in small data sets where is not possible to have a large independent test data set without undermine the scope and performance of the training set. The left out instances constitute the OOB sample, which is used to estimate the ensemble prediction performance.

The number of trees and minimum size node are two parameters in Random Forest. There is not a maximum number of trees but 500 trees show to be sufficient in most cases, so this is the default value [Svetnik et al., 2003]. The other parameter, minimum node size, gives us the minimum number of nodes below which no split will be made, having effect on the grown size of trees. In terms of classification, the default value is 1, ensuring that trees grown until is maximum size.

## Variable importance

RF gives a measure of how each variable contributes to the prediction accuracy that is calculated in the course of training. Variable importance can be measured in terms of the number of times each variable is selected by all trees in the RF. Two measures can be defined to assess this variable importance: the mean decrease accuracy and mean decrease Gini, described below.

---

<sup>4</sup>In our study case, class instances refers to the capability of molecules to cross (or not) the BBB.

*Mean decrease accuracy*

This measure gives a value of how much accuracy would decrease if a variable was removed. So, if a variable is irrelevant, its deletion should not have great impact in the accuracy value, consequently, the mean decrease accuracy will be assigned with a very low value. On the other hand, if a variable is important, its deletion has a great impact in accuracy value, so the mean decrease accuracy should be high.

*Mean decrease Gini*

The mean decrease Gini is a weighted mean of the individual trees improvement in the splitting criterion produced by each variable [Strobl et al., 2006]. Gini index is calculated for each variable, based on the node with lowest index. In each, the more informative attribute is the one that has lower Gini index (impurity index). So, a low Gini index (higher decrease in Gini) means that a particular variable plays a greater role in partitioning the data into the defined classes.

**Random Forest versus Decision Tree**

In DT all descriptors are tested for the splitting process at each node, instead of that, RF only tests a small number of the descriptors, making the search faster <sup>5</sup>. Also, in DT usuallys proceed to pruning in order to improve accuracy, which is usually done via cross-validation, however with RF there is no need of pruning since each tree is computed based in a bootstrap sample.

So, RF has several advantages [Breiman and Cutler, 18-5-2011], that make it one of the best choices in very different cases, some of them are:

- high accuracy values compared to DT and other classification algorithms;
- allows thousands of input variables without variable deletion;
- it is possible to know which variables are more importante; and,
- leads well with imbalanced data sets.

RF were used with resorce to randomForest R library [CRAN-R, 10-4-2011]. This package implements Breiman's Random Forest algorithm [Breiman and Cutler, 18-5-2011], based on Breinam and Cutler's original Fortran code for classification and regression.

---

<sup>5</sup>With randomForest package for R, the default is the square root of the number of descriptors.





# Chapter 4

## Data

The current chapter focuses on implementation details. It starts with a description of the drug molecules dataset (Section 4.1) and the way it was constituted (Section 4.2). Chapter ends with Section 4.3 where is highlighted details behind implemented approaches.

### 4.1 Drug molecules dataset

For this work, a drug dataset of 950 molecules, with quantitatively measured logBB, was compiled using data from previous publications discussing BBB permeability of drug molecules. Molecules were divided into BBB+ and BBB- groups according to the assumption pointed out in Li et al. [2005b] and Zhang et al. [2008], where compounds with experimental logBB  $< -1$  were classified as relatively poor penetrators of the BBB barrier (BBB-), while compounds with logBB  $> -1$  were classified as relatively good penetrators of the BBB (BBB+).

The dataset constitution suffered updates over the course of the work, in order to improve the diversity of compounds, to increase the number of compounds and to constitute a dataset that can approach the reality of the ratios between BBB- and BBB+. So, three phases can be distinguished until the final dataset is constituted. Some of the tested models take as input the initial drug dataset instead of the final one, representing the first steps of this work.

#### Phase I - Initial drug dataset

The initial dataset is comprised of 628 molecules, after elimination of replicated molecules, and it have been accumulated from two sources: [Doniger et al., 2000] and [Li et al., 2005b]. Of the total of molecules, there are 374 BBB+ and 254 BBB- molecules.

#### Phase II

In this phase, the initial dataset was updated with more 101 molecules, of which 92 BBB+ and 9 BBB-, taken from the work of [Zhang et al., 2008]. So, the dataset has reach a total

of 729 molecules (466 BBB+ and 263 BBB-).

### Phase III - Final drug dataset

To better account for the verified a priori probabilities of BBB permeability, a third dataset was used for the final tests.

This last phase aims to decrease the lack of BBB- compounds, after have been highlighted that 95 percent of molecules do not show BBB permeability and only 5 percent of the drug molecules cross the BBB [King, 2011, Pardridge, 2005, Tsaoun et al., 2009, Pardridge, 2003]. Our initial dataset was so far of representing the reality of this estimations, since it had much more BBB+ compounds than BBB- compounds. These percentages represent a twist in the work and represent the starting point for a new approach. Thus, the initial dataset suffered a new updated with more 221 BBB- molecules, selected from [Zhao et al., 2007]. This final dataset is constituted by 950 molecules, of which 466 BBB+ and 484 BBB-.

The structure of the final dataset retains the following informations:

- an alphanumeric ID was defined for each molecule.
- the original name of the molecule as referred in the literature;
- a binary classification,  $p$  (pass BBB) or  $n$  (not pass BBB) defined based on the logBB values; and,
- the SMILES representation of the molecules.

## 4.2 Data preparation and computational approach

In this section it will be describe the procedure until we have our data ready to test models. The overall procedure contained the following four major steps:

### Obtaining SMILES

SMILES representation of molecules was obtained using the Chemical Identifier Resolver [CIR, 18-5-2011], an online service, provided by [NCI/CADD, 18-5-2011]. This service allows to convert a given structure identifier, as the common name of the molecule, into another representation, as the SMILES representation, it also provides a structure editor. To obtain the SMILES representation of each molecule, a Python script was developed (Appendix A.1) that uses this web service for each molecule present in the dataset. Some of the molecules were not recognized by the service, and so the original SMILES provided by source articles were used. Next, duplicate SMILES were eliminated, since several compounds had different names and the same SMILES representation.

### Obtaining molecular descriptors

The Python library Pybel was used to generate all the molecular descriptors described in section 3.2.2 (MW, AMW, LogP, NRing, NBRing, multiplicities and fingerprints), based on their SMILES representation. In appendix A.2 is shown the script used to obtain MW and LogP.

The generated fingerprints were the hash-key fingerprints, somewhat similar to the Daylight fingerprints [O’Boyle et al., 2008], taking as input SMILES representation of molecules. Pybel provides four types of fingerprints, we use FP2 that corresponds to a hash-fingerprint definition [Babel, 4-11-2010].

### Obtaining and reducing similarity matrices

As described in section 3.4.2 there are several measures to describe molecular similarity beyond the Tanimoto coefficient, for each of these measures a similarity matrix was generated. The program devised (Appendix A.3) uses Pybel for calculating the FP2 fingerprints, according to a hash-fingerprint definition [Babel, 4-11-2010] and then is calculated the respective measure for each pair of molecules. The matrix uses the drug dataset of the phase I (628 molecules) and it is constructed comparing each molecule with all the others.

The similarity matrices obtained were then reduced using the `cmdscale()` function of software R. The resulting vectors were used to test k-NN and NN algorithms.

## 4.3 Implementations

### 4.3.1 Principal Coordinates Analysis

As mentioned earlier, the principal coordinates analysis was performed by using the `cmdscale()` provided by R software. The script developed to perform this analysis is described in Appendix A.4. This function takes as argument a distance matrix calculated by the application of the metrics discussed. It also receives the classification of each molecules to whether or not they cross the BBB. The result of the function is a vector of n-dimensions that represent the *principal coordinates* necessary to display a molecule in space. For each coefficient, was obtained vectors for 2, 5, 10, 20, 30, 40, and 50 coordinates. Only dataset I was subjected to PCoA.

### 4.3.2 k-NN

The k-NN implementation was the library `knnflex` [CRAN-R, 12-10-2010] which is a CRAN package. Aggregation method (`knnflex` parameter) was changed to majority class,

the option used for classifications, all the other parameters were left with the default value.

To use k-NN for classification a simple R script was developed (Appendix A.5), taking as input arguments the coordinate vectors provenients from a PCoA analysis. Furthermore, MW and LogP were used as complement to principal coordinates, due to the different orders of magnitude, these two properties were normalized according to the equation 4.1, where  $x'$  is the normalized value,  $x_i$  the value to be normalized,  $\bar{x}$  the mean of all values, and  $\sigma$  the standard deviation.

$$x' = \frac{x_i - \bar{x}}{\sigma} \quad (4.1)$$

$$\sigma = \sqrt{\sum (x_i - \bar{x})^2} \quad (4.2)$$

Tests were performed with several dimensions (2, 5, 10, 20, 30, 40, and 50) and a variable number of neighbors (1, 2, 4, and 8). Molecules of dataset I were used to train model and it was validated by a leave-one-out cross validation. The distance matrix was used with all the elements except one, which was tested against it and attributed a class to the most frequent neighboring class. This procedure was repeated for all molecules and final statistics classification were thus determined.

### 4.3.3 Neural networks

Neural Networks were used for classifications taking as input the principal coordinates vectors obtained after a PCoA analysis of similarity matrices. Like in k-NN method, the NN analysis was performed for all the similarity distances described in section 3.3 considering 2, 5, 10, 20, 30, 40, and 50 dimensions for coordinate vectors. Also here, normalized values of MW and LogP were used to complement principal coordinates. Initially, the number of neurons in the middle layer start with 3, however to investigate the dependence of the model in terms of number of neurons, this number was varied (3, 5, 9, 15, 20, 25, 40).

The molecules of dataset I were randomly divided into ten subsets of approximately equal size for conducting a 10-fold cross validation test of the prediction accuracy of the statistical learning method. Each subset represents the test set and the left molecules were used to train the model.

The choice of range and decay parameters of NN must be decided together, since the choice of one will affect the choice of the other. The rang value corresponds to the initial value of weights and this value is 0.1, which means that the values of the weights along the iterations varies between -0.1 and 0.1. The speed at which the initial positive value reaches the final negative value depends on the value of the decay that has an impact on learning process of the neural network. Thus a very low decay value was chosen so that the network can learn from data and the process only ends when it reaches the

maximum number of iterations (1000). The trace parameter allows to choose between tracing optimization or not. In our case, the option was set to false so the NN does not choose only the best path and thus there are no information lost in the learning process.

### 4.3.4 Random Forest

randomForest R package software was used to implement the model (Appendix A.7). The model was trained and tested with the molecules from dataset III and a set of 1051 molecular descriptors, which are listed in Table 4.1. All the molecular descriptors are exclusively based in structural and physicochemical properties of molecules.

Table 4.1: Molecular descriptors used to build random forest model. MW - molecular weight, AMW - average molecular weight, NRing - number of rings, NBRing - number of bonds in rings.

Descriptors	Number
fingerprints	1023
multiplicities	24
MW	1
AMW	1
NRing	1
NBRing	1
total	1051

Due to the very imbalanced dataset where class priors are highly unequal and imbalanced, the idea of oversampling appear in order to solve this problem. The BBB- molecules have a much larger prior probability (95 percent - should be the majority class) than BBB+ molecules (5 percent - should be minority class). Oversampling will balance training class priors, by increasing the class data points of BBB- in the training set by randomly replicating existing class data points, in order to achieve a majority class. Figure 4.1 translates the process of oversampling for dataset III.

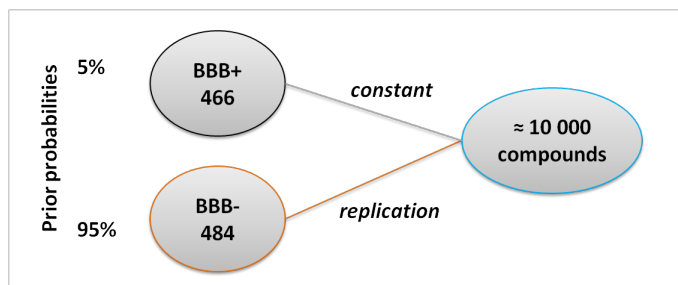


Figure 4.1: Oversampling process for dataset III.

Besides the problem of the imbalanced dataset, it was necessary to find an algorithm

that was robust enough to handle the huge difference between the two classes of compounds and the large amount of molecular descriptors. Random Forest was chosen as capable of yielding exceptional performance with little need to tune parameters. The algorithm is robust to irrelevant parameters, does not overfit and has shown its success on many applications [Sug, 2010].

Since model was trained considering the real distribution of BBB+ and BBB- compounds, appeared the question of how the validation set should be constituted. Then it was considered two strands of validation sets, one considering prior probabilities (approach A) and another without considering prior probabilities (approach B).

### Approach A

This approach decides based on the knowledge of probabilities without data being presented (prior probabilities). Most of the times, these prior probabilities are a subjective assessment of the experience, as in the present work, the prior probabilities of crossing class molecules are an estimation based on existing knowledge of experience and early cases. The classification model assumes that the ratio of molecules (equation 4.3) is skewed in the dataset, and the model is used assuming a fixed probability of 5% BBB+ and 95% BBB-.

$$moleculesratio = \frac{BBB+}{BBB- + BBB+} \quad (4.3)$$

The validation set considering prior probabilities was constituted of 95 molecules. It comprises approximately 91 BBB- molecules and 4 BBB+ molecules randomly selected from BBB- subset and BBB+ subset, respectively. After the selection of this molecules, the left ones are the seeds for oversampling and constitution of training set.

### Approach B

Unlike the previous approach, this approach does not apply the prior probabilities, instead the classification models are used according to the probabilities of occurrence within the dataset.

The validation set is also constituted of 10% of the molecules however each class as approximately 50 percent of BBB- and BBB+ molecules. The selection process is done similarly.

### Validation

As discussed in section 3.5.3, RF already performs an internal cross validation, nonetheless, we perform a 10-fold cross validation to evaluate the consistency and robustness of the model, and evaluate the most and less important descriptors that are used in RF construction and whether they keep constant throughout the iterations. Furthermore, a cross

validation is needed since not all of the molecules can be used to train the model because oversampling can not choose some of the molecules to replicate. Then is necessary validate the model behavior with different training sets after oversampling process.

For 10-fold cross validation, in each iteration the next four steps are performed:

1. Select 10% of all molecules from dataset according to the process describe above for each type of validation set. The remaining are part of training set.
2. Oversampling the training set. All the BBB+ molecules are used to constitute positive part of the dataset (5%) and BBB- are randomly selected (with replacement) and replicated until the final training set reaches the 10000 molecules, in order to simulate 95% of negative part.
3. Construct the RF based on the training set obtained in 2.
4. Model test with the set determined in 1.





# Chapter 5

## Results and discussion

This chapter now shows the results obtained with the methodologies presented and discusses their significance, comparing them to other efforts in the literature.

Many parameters can be used to represent the results of the classification and our choice were made based on the importance of each one. Phi was chosen as the parameter that best represent our results, since it represent an overall measure of the correctness of the model to classify instances.

### 5.1 k-Nearest Neighbour

Starting with k-Nearest Neighbour analysis, a simple test was performed and is based on the following question: "could MW and LogP, separately or together, be used to correctly classify molecules in BBB+ and BBB- molecules?". MW and lipophilicity, as translated by the LogP coefficient, are generally 2 important factors described in the literature that may be able to explain why certain molecules cross the BBB. Figure 5.1 shows that they are not enough to classify molecules, with Phi values being far below expectations (0.05, 0.20 and 0.27 for LogP, MW, MW+LogP, respectively). The presented data represent the results of validation sets according to a leave-one-out cross validation (LOOCV). From these results arise the need to add more variables that can explain more and better the chemical information contained in the molecules. A similarity coefficient was used for fingerprints comparison. The most popular similarity measure for comparing chemical structures is the Tanimoto coefficient [Godden et al., 2000]. PCoA brings dimension reduction of the Tanimoto similarity matrix (628x628) necessary because of the large size that extends the model run time. Testing different dimensions is necessary to assert the amount of structural information present in the molecules (and discriminated by Daylight fingerprints) that may be able to contribute to predict BBB permeability. Besides that, MW and LogP were added to the data sets to verify if their presence increased the predictive power of the models (figure 5.1).

Tanimoto coefficient shows good results (phi value of 0.55) comparing to individually

analysis of MW, LogP and MW+LogP meaning that principal coordinates catch most of the information constant in these 2 properties (figure 5.1). The Tanimoto phi value only gets better when molar weight is added, with a number of dimensions above 20. Any other addition of parameters (tanimoto+LogP and tanimoto+MW+LogP) causes the decrease in the number of phi, and so this kind of approach was not continued.

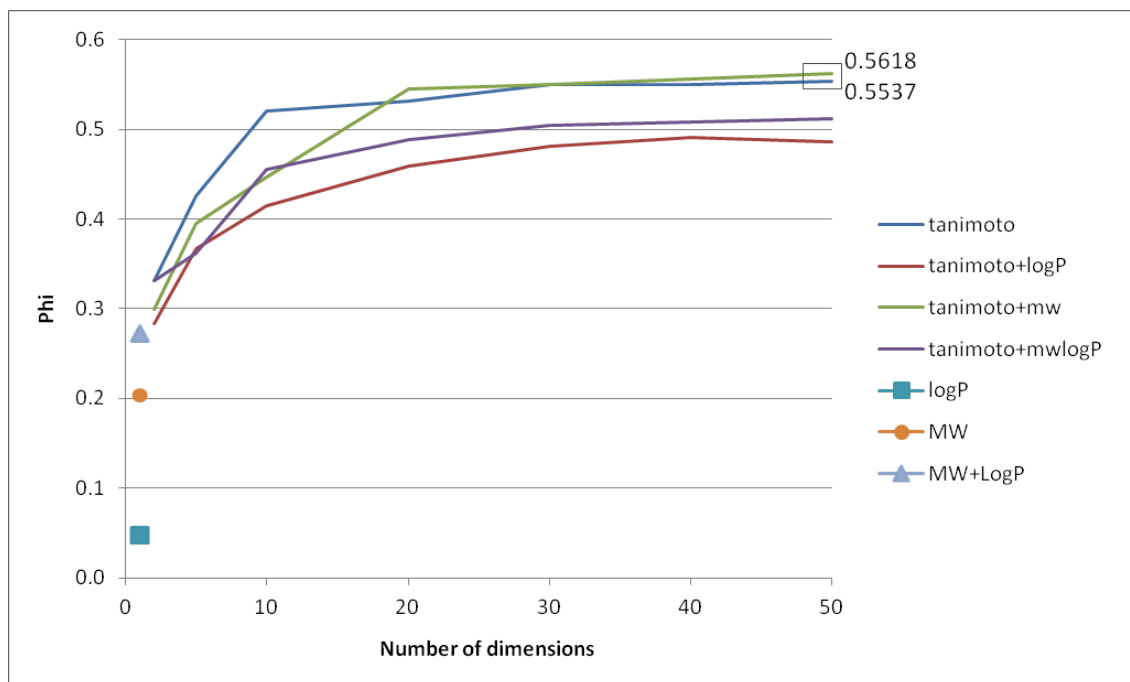


Figure 5.1: Measures of Phi for tested Tanimoto coefficient variants with k-NN algorithm, according to a LOOCV.

Although many other similarity measures can be applied for chemical structures comparison represented by means of fingerprints (Section 3.3). Thus another coefficients were evaluated, showing that higher values of phi can be achieved. Taking as reference the Tanimoto coefficient, Pearson coefficient reaches the highest phi value (figure 5.2) with 30 dimensions. Further, many of the used coefficients exceeds the highest Tanimoto value (for 50 dimensions) with fewer dimensions (figure 5.2 and table 5.1).

It can be noted that coefficients like Cosine, Euclid, Kulczynski, Pearson and Simpson overpass those obtained with Tanimoto. The value obtained with Pearson coefficient shows that 30 dimensions are enough to explain molecules and an increase of dimensions does not mean increasing the explicative power, since the phi value tends to decrease after the 30 dimensions.

In general the results were equivalent between coefficients, requiring a high number of dimensions, with the exception of the coefficient of Forbes.

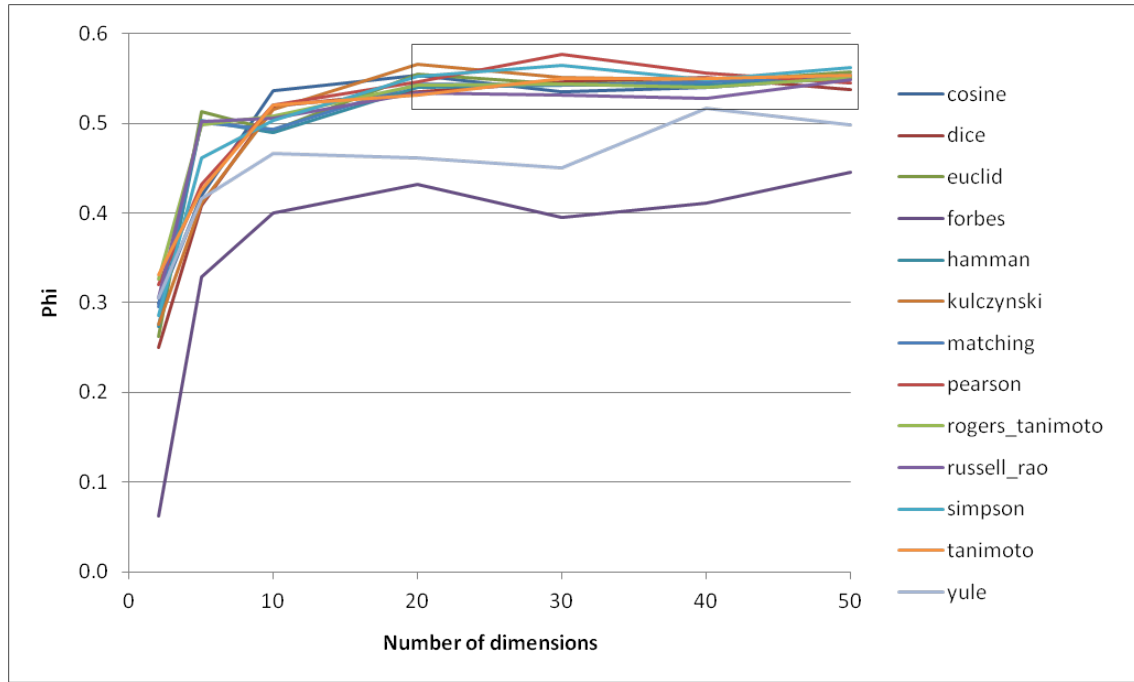


Figure 5.2: Measures of Phi for all the evaluated coefficients with k-NN algorithm, according to a LOOCV.

Table 5.1: Details of measures to Phi for some of the coefficients shown in Figure 5.2, for k-NN algorithm. The best value obtained with the algorithm is highlighted in orange, and for each coefficient is highlighted its best value in blue.

Similarity Measure	No. Dims			
	20	30	40	50
Cosine	0.5541	0.5346	0.5404	0.5506
Dice	0.5346	0.5463	0.5506	0.5382
Euclid	0.5550	0.5439	0.5463	0.5573
Hamman	0.5404	0.5431	0.5428	0.5533
Kulczynski	0.5661	0.5510	0.5484	0.5515
Matching	0.5435	0.5431	0.5463	0.5530
Pearson	0.5459	0.5769	0.5556	0.5448
Rogers-Tanimoto	0.5428	0.5439	0.5400	0.5506
Russel-Rao	0.5337	0.5318	0.5275	0.5481
Simpson	0.5530	0.5642	0.5484	0.5621
Tanimoto	0.5316	0.5494	0.5502	0.5537

Besides the number of dimensions, k-NN depends on the number of neighbors. Based on this number, the algorithm classifies an object by a majority vote of its neighbors, with the object being assigned to the class most common amongst its k nearest neighbours. Figure 5.3 shows the dependency of phi value in order to the number of neighbors for the coefficients and dimensions with highest values up to Tanimoto, namely Cosine and Kulczynski with 20 dimensions, Euclid and Tanimoto with 50 dimensions, and Pearson

and Simpson with 30 dimensions. Regarding to which number of neighbors is better, it seems to exist a tendency inside each group of dimensions, however it is highly dependent of which measure is used. For 20 dimensions, both Cosine and Kulczynski show higher phi values for 1 and 2 than for 4 and 8 neighbors, being this separation higher for the last coefficient. Nevertheless, in both cases the highest value is achieved with 2 neighbors, then may be this value is the best choice for the 20 dimensions. Considering now the 50 dimensions, an improvement of the results with just 1 neighbor is evident, with a certain distance of the values obtained with the other neighbors. Although, when the number of dimensions is 30, the choice of how many neighbors should be used is not so obvious. Simpson shows best results with 1 or 2 neighbors, however when the case is the Pearson there is a clear bias in favor of 4 neighbors, corresponding this to the highest phi value obtained by the algorithm. The results show that it is possible to make some assumptions but they do not exclude the required analysis coefficient to coefficient.

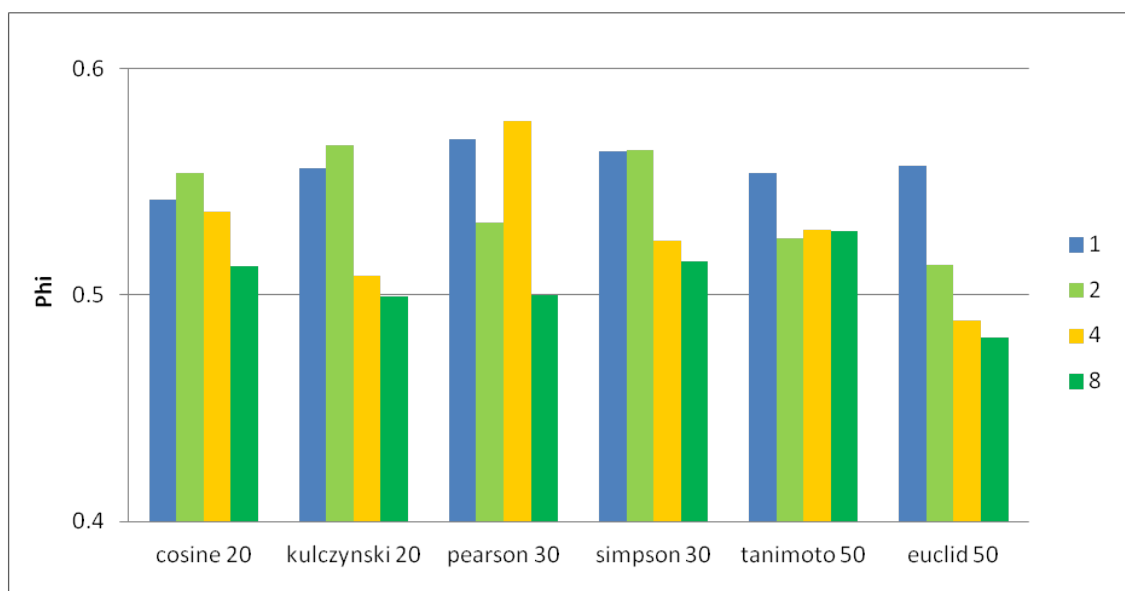


Figure 5.3: Dependency of phi value in order to the number of neighbors.

## 5.2 Neural Networks

Like in k-NN approach, first of all it was analyzed the ability of MW and LogP to predict the molecules class. Although in k-NN the validation sets were subject to a LOOCV, in NN the sets were validated through a 10-fold cross validation. Once again, the phi values are low (0.07, 0.21 and 0.31 to LogP, MW and MW+LogP, respectively) reinforcing the idea that just these properties are not sufficient for a correct prediction (figure 5.4). Following the same procedure, fingerprints comparison with Tanimoto coefficient were used as molecular descriptors after applying a dimension reduction of the original matrix,

Table 5.2: k-NN classification results for several statistical measures. The algorithm achieves its best phi value with Pearson coefficient (orange) and in blue are highlighted the values higher than its measures.

Coefficient	Dims	Accuracy	Precision	Recall	F	BBB+	BBB-	Phi	TP	TN	FP	FN
Cosine	20	0.7866	0.8109	0.8369	0.8237	0.8109	0.7479	0.5541	313	181	73	61
Euclid	50	0.7882	0.8114	0.8396	0.8252	0.8114	0.7510	0.5573	314	181	73	60
Kulczynski	20	0.7898	0.8306	0.8128	0.8216	0.8306	0.7328	0.5661	304	192	62	70
Pearson	30	0.7978	0.8175	0.8503	0.8336	0.8175	0.7657	0.5769	318	183	71	56
Tanimoto	50	0.7866	0.8093	0.8396	0.8241	0.8093	0.7500	0.5537	314	180	74	60
Simpson	30	0.7898	0.8253	0.8209	0.8231	0.8253	0.7383	0.5642	307	189	65	67

resulting in an improvement of the ability to predict (phi value of 0.5792). To Tanimoto vector coordinates were add MW, LogP and MW+LogP, but it was not achieve a significant improvement in the phi value, 0.5792 to 0.5811 (figure 5.4), moreover is necessary an increase in dimension number passing from 40 to 50. Since, no relevant improvements were made with addition of MW and LogP, also here it no longer be applied.

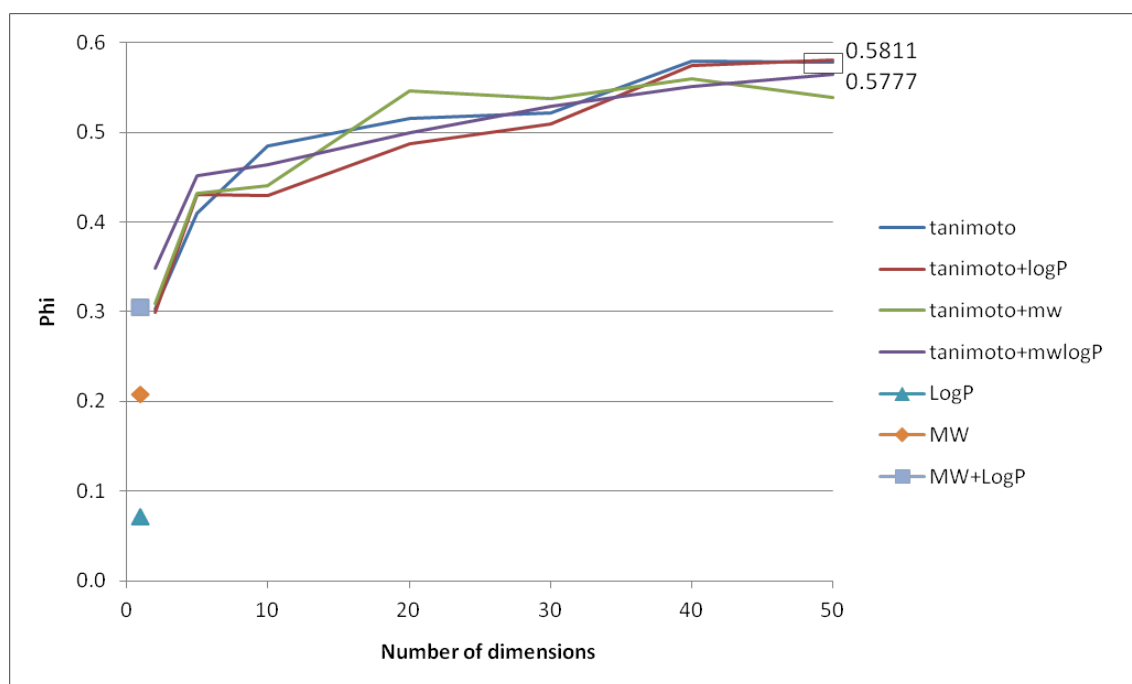


Figure 5.4: Measures of Phi for tested Tanimoto coefficient variants with NN algorithm, according to a 10-fold cross validation.

The next step includes analysis of all the other coefficients referred in section 3.3. The measure of similarity that provides a better value of phi is the Simpson coefficient with 40 dimensions (0.5818). It is noted that, for the same number of dimensions, other coefficients stay very close to this value, as the case of Tanimoto and Pearson. Table 5.3 discriminates the small differences for some of the coefficients.

Results show an increasing of phi value as the number of dimensions increases. A high number of dimensions (tested up to 50 dimensions) translates a better representation of molecules, and it seems to be important to get a good result with NN algorithm. Furthermore, the results between coefficients are similar and show the same tendency, with exception of the Forbes distance that remains significantly below other measures.

Table 5.3: Details of measures to Phi for some of the coefficients shown in Figure 5.5, for NN algorithm. The best value obtained with the algorithm is highlighted in orange, and for each coefficient is highlighted its best value in blue.

Similarity Measure	No. Dims		
	30	40	50
Cosine	0.5311	0.5260	0.5753
Dice	0.5340	0.5460	0.5672
Kulczynski	0.5365	0.5373	0.5363
Pearson	0.5335	0.5738	0.5797
Russell-Rao	0.5224	0.5373	0.5377
Simpson	0.5113	0.5818	0.5789
Tanimoto	0.5212	0.5792	0.5777

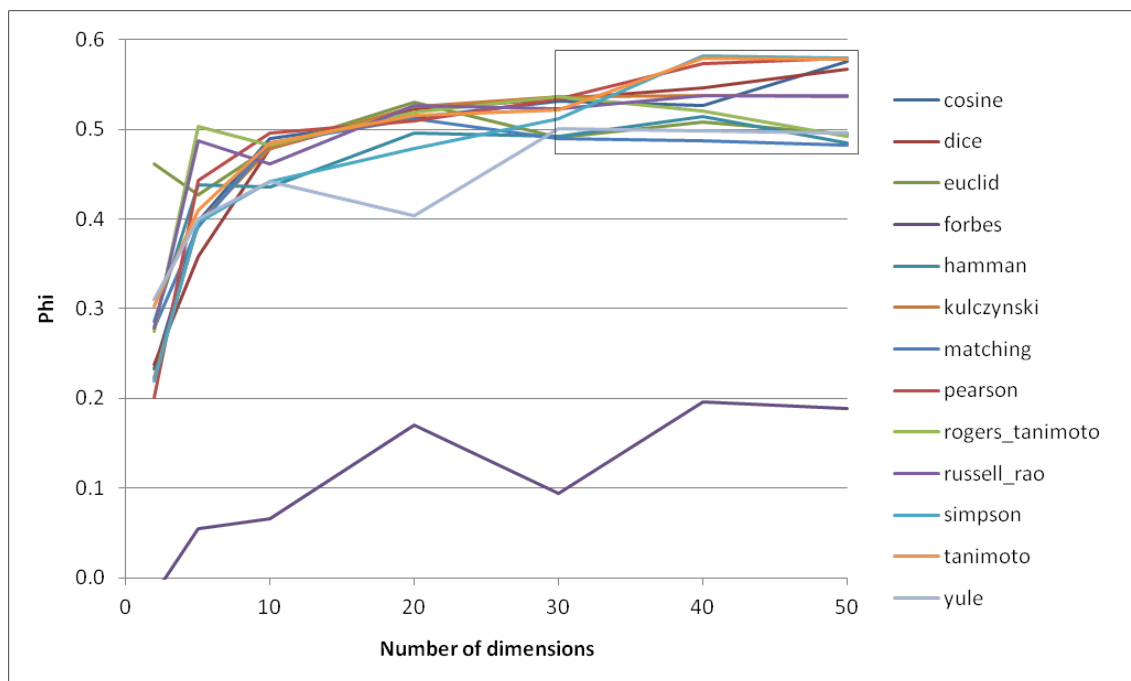


Figure 5.5: Measures of Phi for all the evaluated coefficients with NN algorithm, according to a 10-fold cross validation.

In this approach the number of neurons of middle layer was variable to assess its impact in increasing the predictive power. Considering the 5 highest phi values, it was

Table 5.4: NN classification results for several statistical measures. The algorithm achieves its best phi value with Pearson coefficient (orange) and in blue are highlighted the values higher than its measures.

Coefficient	Dims	Accuracy	Precision	Recall	F	BBB+	BBB-	Phi	TP	TN	FP	FN
Pearson	50	0.7984	0.7361	0.7595	0.7463	0.7361	0.8399	0.5797	18.5	31	6.6	5.9
Simpson	40	0.7968	0.7716	0.7403	0.7532	0.7716	0.8145	0.5818	19.4	30	5.8	6.8
Tanimoto	40	0.7952	0.7474	0.7555	0.7480	0.7474	0.8316	0.5792	18.8	30.5	6.5	6.2

analyzed the number of neurons in the middle layer (figure 5.6). Overall, an increase in the number of neurons results in an increase of phi value, however the number of neurons does not seem to have a bias inside a group of dimensions. For 40 dimensions, Simpson has achieved its best value with 40 neurons, whereas Tanimoto just needs 25 neurons to achieve the best results. Moreover, with 50 dimensions, Pearson gets its best value with 25 neurons, while Cosine gets it with 40 neurons, and Dice shows that the number of neurons in middle layer is highly dependent on the coefficient, being necessary only 9 neurons. It is further noted that although the phi value increases with an increase of the number of neurons, the classification process becomes much slower.

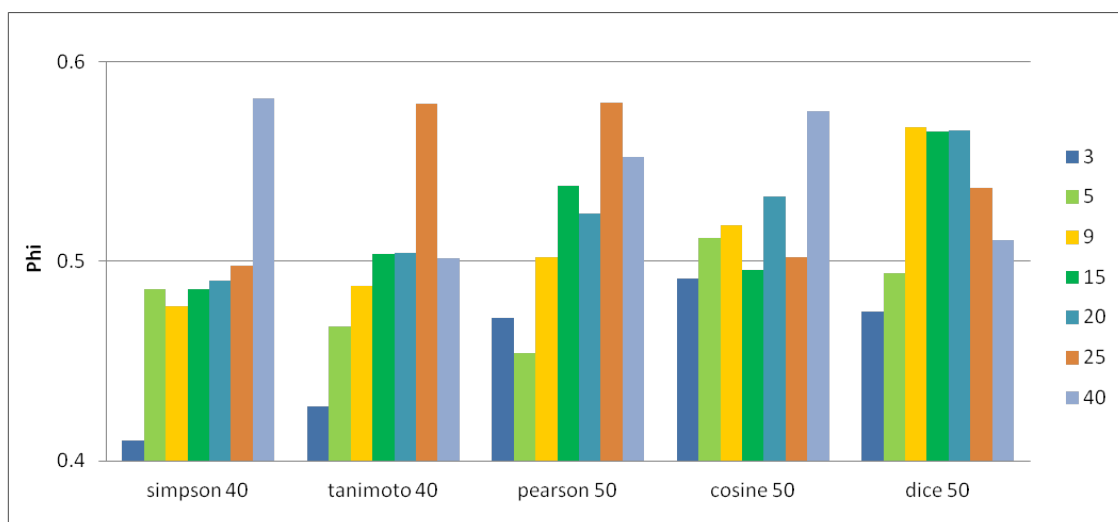


Figure 5.6: Dependency of phi value in order to the number of neurons in middle layer.

### 5.3 Random Forests

RF emerged as a new approach, completely different of the used with the previous algorithms. It makes use of the assumption that 95 percent of the molecules can not reach the brain, to build a model that uses a oversampling of the dataset III to simulate such percentages.

As our dataset is highly imbalanced, it was needed to test two different constitution of validation sets, in which pretends to be a mirror of the BBB+/BBB- ratio in training set. The goal is to verify with which validation set a better classification is achieved, the next three sections show the results obtained for each approach and side by side the results of the two validation sets for the most relevant measures. Both validation sets were validate according to a 10-fold cross validation, being presented for each of the 10 iterations the results for several statistical measures besides their mean.

### 5.3.1 Approach A

Moving through the analysis of results, in table 5.5 are presented the values obtained for each measure for each iteration of the 10-fold cross validation process from the evaluation of the validation set without considering prior probabilities, the final row represent the mean of the iterations. It is presented the mean accuracy, precision, BBB+, BBB- and Phi of 82.6%, 92.0%, 92.0%, 77.2%, and 67.4%, respectively.

Table 5.5: Random Forest results represented in several statistical measures, for approach A. The validation process follows a 10-fold cross validation, being represented the results obtained at each validation iteration, and its mean.

Iteration	Accuracy	Precision	Recall	F	BBB+	BBB-	Phi	TP	TN	FP	FN
1	0.8737	0.3889	0.8750	0.5385	0.3889	0.9870	0.5305	7	76	11	1
2	0.8842	0.3077	0.6667	0.4211	0.3077	0.9756	0.4003	4	80	9	2
3	0.8211	0.1579	0.7500	0.2609	0.1579	0.9868	0.2883	3	75	16	1
4	0.8526	0.1765	1.0000	0.3000	0.1765	1.0000	0.3868	3	78	14	0
5	0.8947	0.3077	0.8000	0.4444	0.3077	0.9878	0.4548	4	81	9	1
6	0.8842	0.1667	0.6667	0.2667	0.1667	0.9880	0.2937	2	82	10	1
7	0.9263	0.4167	1.0000	0.5882	0.4167	1.0000	0.6199	5	83	7	0
8	0.8632	0.2143	0.6000	0.3158	0.2143	0.9753	0.3010	3	79	11	2
9	0.9263	0.3750	0.6000	0.4615	0.3750	0.9770	0.4378	3	85	5	2
10	0.8526	0.2000	0.6000	0.3000	0.2000	0.9750	0.2858	3	78	12	2
mean	0.8779	0.2711	0.7558	0.3897	0.2711	0.9853	0.3999	3.7	79.7	10.4	1.2

### 5.3.2 Approach B

In table 5.6 the mean accuracy, precision, BBB+, BBB- and Phi are of 87.8%, 27.1%, 27.1%, 98.5%, and 40%. For the first case, the values of Phi are in the range of 55.1% 78.5%, and to the second are in the range of 28.6% 62.0%, both cases indicates that there are some worst RF pulling down the Phi value.



Table 5.6: Random Forest results represented in several statistical measures, with approach B. The validation process follows a 10-fold cross validation, being represented the results obtained at each iteration, and its mean.

Iteration	Accuracy	Precision	Recall	F	BBB+	BBB-	Phi	TP	TN	FP	FN
1	0.8632	0.9268	0.7917	0.8539	0.9268	0.8148	0.7347	38	44	3	10
2	0.8947	0.8781	0.8781	0.8781	0.8781	0.9074	0.7855	36	49	5	5
3	0.8316	0.9429	0.7021	0.8049	0.9429	0.7667	0.6846	33	46	2	14
4	0.8632	0.9000	0.8000	0.8471	0.9000	0.8364	0.7281	36	46	4	9
5	0.7895	0.9032	0.6222	0.7368	0.9032	0.7344	0.5987	28	47	3	17
6	0.7579	0.9000	0.5745	0.7013	0.9000	0.6923	0.5507	27	45	3	20
7	0.8105	0.9167	0.6875	0.7857	0.9167	0.7458	0.6428	33	44	3	15
8	0.8421	0.9167	0.7333	0.8148	0.9167	0.7966	0.6930	33	47	3	12
9	0.8632	0.9512	0.7800	0.8571	0.9512	0.7963	0.7415	39	43	2	11
10	0.7474	0.9697	0.5818	0.7273	0.9697	0.6290	0.5774	32	39	1	23
mean	0.82632	0.92052	0.71512	0.80070	0.92052	0.77196	0.67369	33.5	45	2.9	13.6

### 5.3.3 Approach A versus Approach B

Analysing the table 5.7 we can get, just by considering the Phi value, that is necessary a balanced validation set to proper validate the model. The values BBB+ and BBB- (probability of hitting the classification for BBB- and BBB+) in the validation set considering prior probabilities, 98.5% and 27.1%, respectively, show that BBB+ molecules presented are not sufficient to evaluate the predictive potencial of the generated RFs, since it can classify equally well both classes of molecules when considering a validation set without prior probabilities.

randomForest package of R allows saving RFs, and in case of choosing just one tree is necessary a careful analysis that can not be based only on the Phi value. The choice should reflect the analyse of the other measures that are as important for the study case, even if the phi value may decrease a bit. As the passage of molecules through the BBB is an exception, it is more important to correctly classify the minority class, so there is a higher cost for misclassifying the minority class than misclassifying the majority class. For example, in table 5.6, the second iteration is the high value of Phi (78.5%) however, it has the lowest probability of hitting the classification of BBB+ molecules (87.8%). On the other hand, the lower value of Phi (57.7%) has the higher probability of hitting the classification of BBB+ with 97%. This example shows the importance of analysing all the measures and not being restricted to just one.

From now on, it will be only considered for analysis the results obtained for the validation set that not consider the prior probabilities (Approach B).

Table 5.7: Comparison of the accuracy, precision, BBB+, BBB-, and Phi for the validation sets considering approaches A and B.

Approach	Measures				
	Accuracy	Precision	BBB+	BBB-	Phi
A	0.8779	0.2711	0.2711	0.9853	0.3999
B	0.8263	0.9205	0.9205	0.7720	0.6737

### 5.3.4 Variable importance analysis

Moving on to importance analysis of molecular descriptors, as mentioned earlier, it was performed an analyse of the most important variables (Figure 5.7) by selecting in each iteration the twenty more important, and for the less important variables it was select the twenty-five less important (figure C.2 of appendice). The presented value of mean decrease accuracy for each variable is a mean of all iteration where variable appear. This is only valid for the more important variables, for the less important the value of the mean decrease accuracy is zero. In figure 5.7 is also presented the frequency of the variables, it was scaled (multiplied by a factor of 0.001) just to appear at the same plot, then a frequency of 0.01 in reality corresponds to 10 and then it means that the variable are present in all the iterations (10).

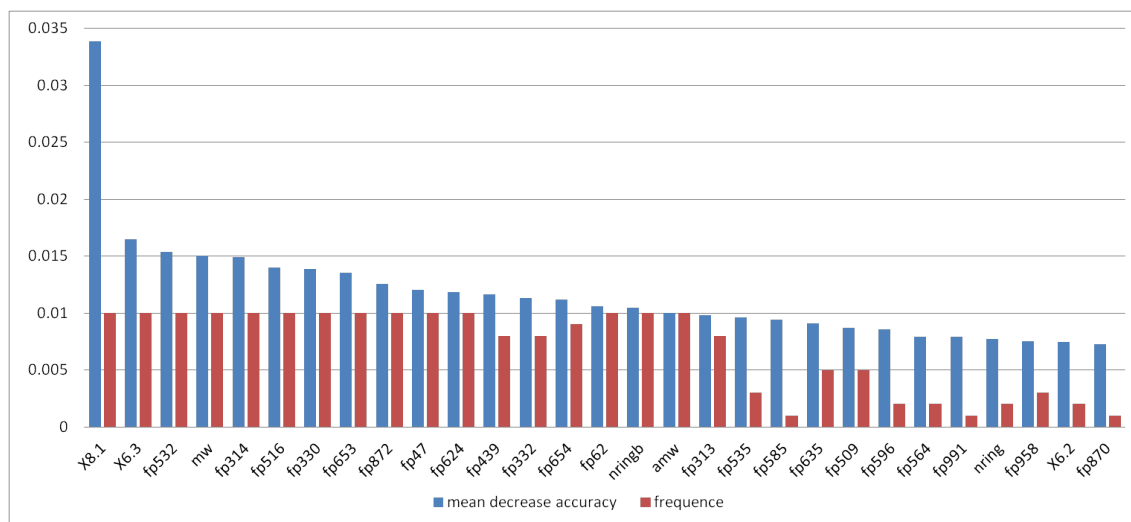


Figure 5.7: Mean decrease accuracy and frequency of the twenty more important variables.

Starting with the more important variables, of the 29 variables collected from the iterations, 14 (about 48%) are presented in all iteration and represent the highest values of mean decrease accuracy, further only 3 (about 1%) of them are presented in one iteration.

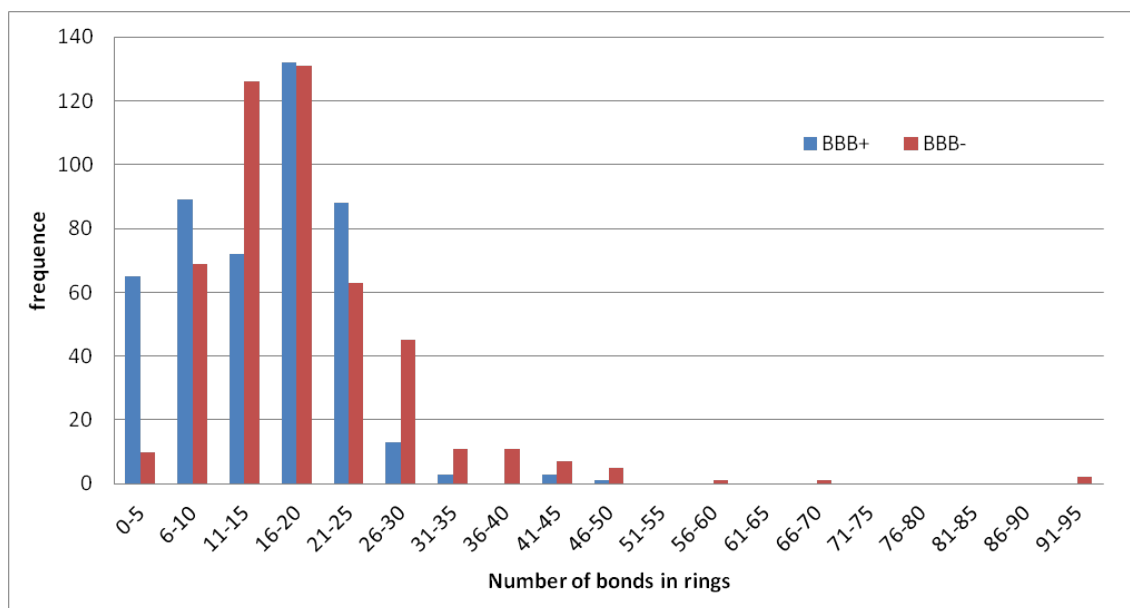


Figure 5.8: Number of the bonds in rings and its distribution among BBB- and BBB+ molecules.

As each fingerprint encodes a substructure it is easy to extract from these descriptors information about which specific molecular characteristics are important for the BBB+ and BBB- predictions. Therefore, in the figure C.1 (Appendix C) are shown the 14 more important variables and the 10 less important presented in all iterations with the respective molecular substructure, when applied.

Figure 5.8 shows the frequency of the number of bonds in rings in relation with both classes of compounds. 52% of the BBB+ compounds have between 0 to 10 and 21 to 25 bonds in rings, it is important to highlight that 14% of the molecules do not have rings (zero bonds in rings) and only 0.2% of the BBB- compounds do not have rings. Regarding BBB- compounds, 43% of them have 11 to 15 and up to 26 bonds in rings. With 16 to 20 bonds in rings both classes has approximately the same representativity. Considering the last range as a threshold, above it compounds have a higher probability of being classified as BBB-, below it compounds have a higher probability of being classified as BBB+. The values are concordant with an increase in hydrophobicity of molecules as the number of rings increase.

Table 5.8: Molecular weight and its average (AMW): comparison and distribution in BBB+ and BBB- molecules.

label	MW		AMW	
	min	max	min	max
n	109.13	1879.66	5.38	21.93
p	42.08	1085.15	4.24	24.67

Random Forests choose oxygen with one single bond as the most important variable to distinguish between BBB+ and BBB- compounds. Figure 5.9 represents the distribution of the number of oxygens with a single bond in both classes of molecules. Unlike the other molecular descriptors analyzed, with this variable there is a bigger separation between the number of appearances between BBB+ and BBB- compounds. 86% of the BBB+ compounds have 0 to 2 oxygens with single bond, whereas 72% of the BBB- compounds have above 3 oxygens with single bond (until a maximum of 24).

An interesting fact is that AMW is an important molecular descriptor and its values are similar in both classes, nonetheless, the maximum of AMW is a bit higher for BBB+ compounds than BBB- compounds, this can be explained by the larger number of atoms in higher MW compounds.

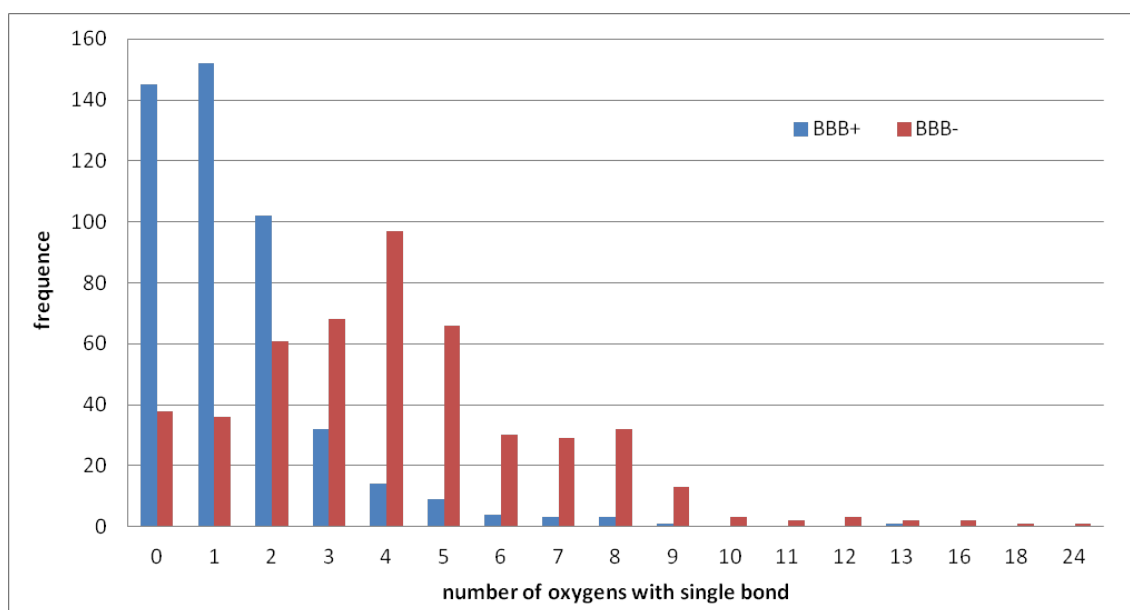


Figure 5.9: Distribution of oxygens with single bond among BBB- and BBB+ molecules.

## 5.4 Discussion

Table 5.9 resumes the principal results obtained for the three methodologies presented in this work and the results for some of the studies presented at section 2.3. It shows some similarities between k-NN and NN algorithms, with similar values of overall accuracy and phi. The difference remains in which class of compounds is predicted with highest accuracy: the k-NN predicts better BBB+ compounds, whereas the NN predicts better the BBB- compounds. The higher predictive power of BBB+ compounds with k-NN is expected since the dataset I has a larger number of BBB+ than BBB- compounds, which will have more representativity in chemical space of training set. The NN is better able to handle with an imbalanced dataset (dataset I) having a good predictive power for BBB-

compounds, even as these were a minority. Regarding the number of dimensions and the overall accuracy, k-NN algorithm has better results with a lower number of dimensions, which is advantageous. The literature signals that MW and logP are two important variables for BBB studies, although, the presence of them does not bring an improvement for classification when the fingerprint data is being used for classification, suggesting that those variables can, in any way, be implicitly derived from the fingerprints. The addition of informative variables conjugated with coordinates from an PCoA analysis seems to be a good representation for compounds and retain the necessary information for a good prediction of classes, being worse with informative variables.

The search for better results leads us to an approach that uses Random Forests. Then, adding RF results to the analyse, it shows a higher phi value, which is in accordance with a more robust approach, that can handle a highly imbalanced dataset (95% of BBB- and 5% of BBB+ compounds) getting a higher predictive capability for the minority class of compounds (BBB+), an important. That means that the used variables, namely some fingerprints, MW, AMW, the number of times that appear oxygen with single bond and the number of bonds in rings, are important as molecular descriptors and allow a correct classification in BBB+ or BBB-.

Table 5.9: Methodologies and studies comparison. The best results with k-NN, NN and RF.

Molecules (p/n)	Descriptors	Methods	Accuracy (%)			
			BBB+	BBB-	Overall	Phi
628 (374/254)	30	k-NN	81.75	76.57	79.78	0.5769
628 (374/254)	40	NN	77.16	81.45	79.68	0.5818
950 (466/484)	1051	RF	92.05	77.2	82.63	0.6737
Studies	Ajay and Murcko [1999]		92	71	81.8	-
	Doniger et al. [2000]		82.7	80.2	81.5	-
	Li et al. [2005b]		88.6	75.0	83.7	0.645
	Zhang et al. [2008]		-	-	82.5-100	-
	Adenot and Lahana [2004]		90	92	91	-
	Zhao et al. [2007]		98.2	87.8	97.2	-
	Zhao et al. [2007]		80.1	63.1	74.6	-

Next, it will be made a comparison between the results obtained with RF (since they are the best results) with the presented results from previous works (section 2.3). It should be noted that a direct comparison may not be appropriate because of the differences between methodologies, namely the different algorithms and descriptors used to build up the models and the way as it is assessed the correctness of the models. Most studies [Ajay and Murcko, 1999, Crivori et al., 2000, Cruciani et al., 2000, Zhao et al., 2007, Zhang et al., 2008] validate their models using external (and independent) validation sets, whereas in our approach the RF model is validated through a more stringent 10-fold cross

validation process, being the obtained result the average of all 10 iterations. Of the ten studies presented, six of them deserve comments, the remaining four show values lower than ours.

Ajay and Murcko [1999] work has the particularity of being a little bit different of the remaining works, including the presented. The main goal is to predict which compounds are CNS-active or CNS-inactive, instead of the prediction in BBB+ or BBB-. However, it is a good case for comparison since it shows similarities with our methodology that implements RF. It uses a very large dataset of compounds to train his model (9000 compounds) and obtain good results and in our case, the methodology makes use of oversampling to constitute a dataset of 10000 "compounds", the results are similar showing that our approach can reproduce a real data set of compounds and can be used as a valid data set.

The starting point of this project was Doniger et al. [2000] study, having been used all the compounds presented by him in our datasets. When comparing with this study, RF methodology obtained higher predictive values for BBB+ and overall accuracies, and for BBB- accuracy it loses in -0.3%. This apparent difference in classifying correctly the BBB- molecules is compensated in the present approach with a significant increase in the classification of BBB+ molecules.

Li et al. [2005b] study is the only one that presents a phi value for comparison, that is just a slightly lower than ours, further BBB+ and BBB- accuracies has higher values. Only the overall accuracy is slightly greater than our. These values show that only one statistical measure is not enough to classify a model, looking at phi values, they do not show significant differences between models, although accuracies values highlight improvements with RF methodology. The results are from 5-fold cross validation process, therefore a comparison of RF model with this should be more appropriate, than with others.

The highest overall accuracy value is presented by Zhang et al. [2008]. These authors value is only slightly smaller than the presented study. Further, when it is used the applicability domain (AD), the overall accuracies increases to 100%. This value would be perfect was not the decrease of the capability to predict the classification of only 60% of the compounds. These authors's methods appear inefficient and cannot be used alone, being always necessary the help of other tools that can give a classification for all compounds. It also should be note that values for BBB+ and BBB- accuracies are not given, and so it is not possible to make a more deep comparison.

The methodology proposed by Adenot and Lahana [2004] shows good results for all analysed accuracies. Although our methodology can reach a slightly better predictive classifications for BBB+ compounds.

Finally, Zhao et al. [2007] presents accuracies values for two training datasets. One of them (the lowest) with 1093 compounds shows the highest predictive power when comparing with all the previous studies and our methodology. However, the results obtained with the second training set with 1593 compounds deserve some comments. The increase

of the number of compounds leads to a huge decrease in the predictive power to -18.1, -24.7 and -13.5 in BBB+, BBB- and overall accuracies, respectively. The results are surprising since it should lead to more accurate results with an improvement in variability of chemical space. This however is not verified, quite the contrary, all accuracies decrease, mainly the BBB- accuracy, showing that this methodology may not appear very robust to changes in the variables domain. The second training set appears too biased towards a strong emphasis on BBB+ molecules. This could be the origin for a worst predictive power for BBB- compounds, impairing all the other measures, and is concordant with some of the present study results obtained with a skewed dataset.

Table 5.10: Classes of molecular descriptors used in studies reported in literature and in this work.

Study (reference)	Classes of molecular descriptors
Ajay and Murcko [1999]	simple molecular properties, structural properties
Doniger et al. [2000]	simple molecular properties, geometrical properties
Li et al. [2005b]	simple molecular properties, molecular connectivity and shape, electrotopological state, quantum chemical properties, geometrical properties
Zhang et al. [2008]	simpler molecular properties, molecular connectivity and shape, electrotopological state, geometrical properties
Adenot and Lahana [2004]	simple molecular properties, geometrical properties
Zhao et al. [2007]	simple molecular properties, geometrical properties, quantum chemical properties
this work	simple molecular properties, structural properties

Li et al. [2005b] presents the molecular descriptors divided into 5 classes, (table B.1 of Appendix B) which was completed with one more class, named structural properties, in order to include the molecular descriptors used in this work.

Moving on to an analysis of the molecular descriptors used in the 6 studies analyzed earlier, it was made an attempt to categorize them according to the several classes of molecular descriptors (Table 5.10). All the studies build up models based on simpler molecular descriptors ([Ajay and Murcko, 1999, Li et al., 2005b, Zhao et al., 2007, Zhang et al., 2008, Doniger et al., 2000, Adenot and Lahana, 2004]). The importance of this class of descriptors seems to be a generalized idea among the studies, and it is also concordant with the results obtained for RF model, in which some of the descriptors of this class were present in importance variable analysis (namely MW, AMW, number of bonds in rings, and others). Ajay and Murcko [1999] study is the only one that uses the same classes of molecular descriptors of the RF model, although in a much smaller number, 166 of the 173 descriptors are structural whereas in this work it was used 1023 structural descriptors. As mentioned before, the goal of the studies is different, although the

comparison serves to demonstrate that good results are achieved considering these two classes of descriptors: simpler molecular and structural properties, besides that Li et al. [2005b], Zhao et al. [2007] and Zhang et al. [2008] works using a wider range of descriptors do not show a much higher and generalized improvement in accuracies values. This shows that structural descriptors retain and can represent most of the others descriptors in BBB+/BBB- classification. Nevertheless, the addition of descriptors belonging to other classes of descriptors could be a good test for Random Forests to show their relevance in an importance analysis.

The number of compounds used to train logBB models is very limited in most of the studies. Therefore, BBB models developed from these limited datasets and used for high-throughput screening of drug compounds may result in large predictive errors, since many groups of compound will be outside the chemical space of the compounds used to set up the models. It should exist an effort for the constitution of larger datasets.

Random Forests seem to be a good approach, handling prior probabilities and highly imbalanced dataset, giving very good results whereas other approaches have so far not proved as robust.



# Chapter 6

## Conclusion

Several works identified the failure to pay sufficient attention to the prediction and assessment of the ability of compounds to cross the BBB [Alavijeh et al., 2005] as a restriction in the CNS drugs development. The number of cases with CNS disorders has experienced a huge growth over the years and the time taken for CNS drugs to reach the markets is bigger than other therapeutic areas, so there is a need for more research that can lead with CNS therapeutic needs.

Thus, computational approaches have been increasingly used to assist in the discovery of useful information on the huge quantity of chemical information produced nowadays by the pharmaceutical industry. This work demonstrates how these approaches can be used to assist in the search for compounds that are able to cross the blood-brain barrier, in a simple, fast and inexpensive way, and tries to be a contribution for this important area of research.

At the beginning of this work it was pointed out a premise that is the basis of the present study: similar molecules have similar behaviors. Previous studies have proven the correctness of this idea, and once again the idea is supported by the computational methodologies developed in this work, which make use of the structural information retained in molecules to perform comparisons and classify molecules based on the classification of similar molecules. Of the three statistical learning methods tested, Random Forests appear to give a higher prediction accuracy than other methods, mainly for molecules that are able to cross the BBB.

The use of the structural features of the molecules, as molecular fingerprints, seems to be a good approach and quite enough to correctly classify and label molecules in BBB+ and BBB- with high probability, assessed by several statistical measures. For Random Forest, the most relevant measures (accuracy and phi) reach 92.1%, 77.2%, and 82.6% for BBB+, BBB-, and overall accuracies, and 0.674 for phi, which corresponds to the mean classification after a 10-fold cross validation. The phi value is not yet a common measure, and so it was only found a value in the literature to compare. The result obtained with Random Forests approach is higher, however is still far away of being a perfect classifier

(1.0).

Further, the use of prior probabilities of BBB+ and BBB- molecules seems to contribute positively for the increase of the accuracy and phi values when comparing with simpler statistical methods based on k-Nearest Neighbours and Neural Networks methodologies. Even with a very small percentage of positive molecules in the oversampled training set, the model can predict and hit the classification of these molecules with high probability. To conclude, Random Forests seem to be the best approach of the ones tested. It produces a highly accurate classifier, when compared to k-NN, NN and other tested algorithms not presented in this document. It can handle thousands of input variables without variable deletion, and can deal with imbalanced datasets still providing good classification results.

The use of Random Forests in chemistry, even for this same problem, is not new, but this approach brings novelty by the used variables and their number, the oversampling and application of the prior probabilities, such that Random forest implementation of software R states the option although this has not yet been implemented.

In the future, it would be interesting to develop a tool that implements the methodology that uses Random Forests and apply it to real world problems, for example in a drug development project to determine if study molecules are able to cross the BBB. A possible extension of this work would also include an update of the current data set to form a set of real data without the negative replication of molecules as in this work, and the use of other variables and molecular descriptors, known as having influence in the BBB permeability. The dataset update would imply a much larger dataset. Thus, a much more chemically diverse dataset would be available publicly aiming for better prediction models and understanding of the chemical nature of the Blood-Brain Barrier.

# Appendix A

## Methodologies

The current appendix begins the series of two appendices. This first comprises the code samples developed during the project and they were used to perform some tasks and implement the approaches. The first two sections represent python scripts to obtain SMILES representation for the molecules of the constituted dataset. The second script allows to get the similarity matrices, which is a representation of how similar the molecules are. The remaining four correspond to implementations in R for a PCoA analysis of similarity matrices, a k-NN, NN and Random Forest analysis, constituting the three methodologies developed.

### A.1 Obtaining SMILES from compound names

Python script to obtain SMILES representation of molecules at section 4.2. It is used as a first step of this work, where SMILES will then be used to get molecular descriptors.

```
import urllib
compounds = open('article_smiles.txt')
output = open('article_smiles.csv', "w")
erro = '<h1>Page not found (404)</h1>'
except = 'not found'
# 1. use http://proxy.di.fc.ul.pt:3128 for http proxying
proxies = {'http': 'http://proxy.di.fc.ul.pt:3128'}
# 2. iterate over compounds names at file
for l in compounds.readlines():
    comp = l.strip()
    # 3. passing compound name via URL
    _url = 'http://cactus.nci.nih.gov/chemical/structure/' + comp + '/' +
        'SMILES'
    # 4. open URL for reading
    filehandle = urllib.urlopen(_url, proxies=proxies) # or filehandle
        = urllib.urlopen(_url), when it is not necessary to specify
        proxies
    # 5. retrieve the corresponding SMILES from URL
    smiles = filehandle.read().strip()
    filehandle.close()
    if smiles == erro :
```

```
        output.write(comp + ', ' + excep + '\n')
    else :
        # 6. write SMILES to output file
        output.write(comp + ', ' + smiles + '\n')
compounds.close()
output.close()
```

## A.2 Obtaining molecular descriptors: MW and LogP

Python script to obtain molecular weight and LogP from SMILES representation of molecules at section 4.2. They were used as coordinates for k-NN and NN approach to assess its contribution for the prediction of molecules class.

```
import pybel
fichIn = open('smiles.txt')
fichMW = open('mw.txt', "w")
fichLogp = open('logp.txt', "w")
# 1. read SMILES
for l in fichIn.readlines():
    smiles = l.strip()
    # 2. read molecule from a string
    mol = pybel.readstring("smi", smiles)
    # 3. write molecular weight of molecule
    fichMW.write(str(mol.molwt) + '\n')
    # 4. dictionary containing descriptor value: LogP
    descvalues = mol.calcdesc(["LogP"])
    # 5. write LogP
    fichLogp.write(str(descvalues) + '\n')
fichIn.close()
fichMW.close()
fichLogp.close()
```

## A.3 Similarity matrices for PCoA analysis

Python script to get the similarity matrices for all the similarity measures described in section 3.3. Matrices complete the data preparation phase (section 4.2).

```
import pybel
from math import sqrt
import csv

# 1. equations that define which of the similarity measures
# a - number of 1 bits in object A but not in object B
# b - number of 1 bits in object B but not in object A
# c - number of 1 bits in both objects
# d - number of 0 bits in both objects

def tanimoto(a,b,c,d):
    return c / (a + b + c)
def cosine(a,b,c,d):
```

```

    return c / sqrt((a + c) * (b + c))
def dice(a,b,c,d):
    return (2.0 * c) / (a + b + 2 * c)
def euclid(a,b,c,d):
    return sqrt((c + d) / (a + b + c + d))
def forbes(a,b,c,d):
    return c * (a + b + c + d) / (a + c) * (b + c)
def hamman(a,b,c,d):
    return ((c + d) - (a + b)) / (a + b + c + d)
def kulczynski(a,b,c,d):
    return 0.5 * (c / (a + c) + c / (b + c))
def manhattan(a,b,c,d):
    return (a + b) / (a + b + c + d)
def matching(a,b,c,d):
    return (c + d) / (a + b + c + d)
def pearson(a,b,c,d):
    return (c * d - a * b) / sqrt((a + c) * (b + c) * (a + d) * (b + d))
def rogers_tanimoto(a,b,c,d):
    return (c + d) / (2 * a + 2 * b + c + d)
def russell_rao(a,b,c,d):
    return c / (a + b + c + d)
def simpson(a,b,c,d):
    return c / min((a + c), (b + c))
def yule(a,b,c,d):
    return (c * d - a * b) / (c * d + a * b)

# 2. generates similarity matrices for the methods defined above, it
# receives a list with compound names, a list with corresponding
# SMILES and method name
def dist_matrix(smi_names, smi_list1, smi_list2, method) :
    objA = []
    objB = []
    a = 0.0
    b = 0.0
    c = 0.0
    d = 0.0
    dist = 0.0
    if method == 'tanimoto' :
        dist = tanimoto
    elif method == 'cosine' :
        dist = cosine
    elif method == 'dice' :
        dist = dice
    elif method == 'euclid' :
        dist = euclid
    elif method == 'forbes' :
        dist = forbes
    elif method == 'hamman' :
        dist = hamman
    elif method == 'kulczynski' :
        dist = kulczynski
    elif method == 'manhattan' :
        dist = manhattan
    elif method == 'matching' :
        dist = matching

```

```
elif method == 'pearson' :
    dist = pearson
elif method == 'rogers_tanimoto' :
    dist = rogers_tanimoto
elif method == 'russell_rao' :
    dist = russell_rao
elif method == 'simpson' :
    dist = simpson
elif method == 'yule' :
    dist = yule
# 3. output file name
name_matrix = 'matriz_' + method + '.csv'
matrix = open(name_matrix, "wt")
matrix.write(' , ')
for i in smi_names :
    matrix.write(i + ', ')
matrix.write('\n')
for i,j in zip(smi_names,smi_list1):
    matrix.write(i + ', ')
    for l in smi_list2:
        smiles = [j,l]
        # 4. create a list of the two molecules to be compared
        mols = [pybel.readstring("smi", x) for x in smiles]
        # 5. calculate their fingerprints
        fps = [x.calcfp() for x in mols]
        # 6. obtain their on-bits
        objA = fps[0].bits
        objB = fps[1].bits
        # 7. calculus of a, b, c and d parameters
        for _i in objA :
            for _j in objB:
                if _j == _i :
                    c += 1
        a = len(objA) - c
        b = len(objB) - c
        d = 1024 - c - a - b
        # 8. passing parameters for method
        mydist=dist(a,b,c,d)
        # 9. calculus of coefficient for the method
        coefficient = 1 - mydist
        print (coefficient)
        matrix.write(str(coefficient) + ', ')
        c = 0.0
    matrix.write('\n')
matrix.close()

# 10. run function to generate similarity matrices
def run():
    import pybel
    from math import sqrt
    import csv
    fich = open ("file.dat", "r")
    names = []
    list1 = []
    list2 = []
    reader = csv.reader(fich, delimiter="," )
```

```
for r in reader :
    names.append(str(r[0]))
    list1.append(str(r[1]))
    list2 = list1
fich.close()
# 11. create list with methods, of which we want to obtain
# similarity matrix
met=[]
met.append('tanimoto')
met.append('cosine')
met.append('dice')
met.append('euclid')
met.append('forbes')
met.append('hamman')
met.append('kulczynski')
met.append('manhattan')
met.append('matching')
met.append('pearson')
met.append('rogers_tanimoto')
met.append('russell_rao')
met.append('simpson')
met.append('yule')

# 12. for each method, call function that retrieves similarity
# matrix
for m in met:
    dist_matrix(names, list1, list2, m)

if __name__=="__main__":
    run()
```

## A.4 Principal Coordinates Analysis

R script to perform a principal coordinates analysis (PCoA) of the similarity matrices calculated (section 4.3.1).

```
#script: principal coordinates analysis
# 1. function to perform the principal coordinates analysis, it
# receives the number of dimensions (k), the similarity matrix (
# dat_matrix), the file with classification (p or n) for each
# molecule (file_classification), and the name of the output file
pcoa <- function(k, dat_matrix, file_classification, output){
  # 2. read similarity matrix
  dat <- read.table(dat_matrix,header=T,sep=",")
  # 3. reduce original matrix to a k-dimensional vector
  result <- cmdscale(dat,k)
  # 4. read file with classification for each molecule
  nomes <- read.csv(file_classification, header = TRUE, sep = ";",
    quote="\"", dec=".")
  # 5. each label (n and p) is assigned with a color
  cols <- c("blue","red")[nomes$np]
  # 6. plot molecules in a 2-dimensional space
```

```

plot(result, col=cols, xlab="Dimension 1", ylab="Dimension 2",main
      ="cmdscale()")
# 7. write k-dimensional vector to file
write.table(result, file=output, sep=",", row.names=T, quote=FALSE)
}

# call pcoa function
pcoa(40, "matriz_tanimoto.txt", "dadosNP.csv", "cmdscaleK40_tanimoto.
      txt")

# run pcoa function: build string name of the different files with
# similitiy matrix, and for each file run the pcoa function with
# each of the dimensions defined
runPCOOA <- function() {
  metodos<-c("tanimoto", "cosine", "tanimoto+mwlogP", "tanimoto+mw", "
    tanimoto+logP", "dice", "euclid", "forbes", "hamman", "
    kulczynski", "manhattan", "matching", "pearson", "
    rogers_tanimoto", "russell_rao", "simpson","yule")
  for (met in metodos) {
    for (K in c(2, 5, 10, 20, 30, 40, 50)) {
      matriz <- paste("matriz","_",met,".txt", sep="")
      output<-paste("cmdscaleK",K,"_",met,".txt", sep="")
      pcoa(K, matriz, "dadosNP.csv", output)
    }
  }
}
runPCOOA()

```

## A.5 k-Nearest Neighbor

R script to implement a k-Nearest Neighbor approach, receiving coordinates vectors from PCoA analysis (section 4.3.2).

```

#script: knn analysis
# 1. function to perform the knn analysis, it receives the file with
# vector of coordinates (file), the number of dimensions of the
# vector (ndim), the file with classification (p or n) for each
# molecule (file_classification), the number of neighbors (k)
library(knnflex)
k_nn<-function(file, ndim, namedist, file_classification, k) {
  n_oks <- 0; n_fails <- 0; t_pos <- 0; t_neg <- 0; f_pos <- 0; f_neg
    <- 0
  # 2. read coordinates vector
  results <- read.table(file,header=T,sep=",", dec=".")
  # 3. read file with classification for each molecule
  nomes <- read.csv(file_classification, header = TRUE, sep = ";",
    quote="\"", dec=".")
  cl <- nomes$np
  siz <- nrow(results)
  vec <- (1:siz)
  # 4. leave-one-out cross validation
  for(i in 1:siz) {
    # 5. definition of training and test sets

```



```

train<-vec[-i]
test<-vec[i]
# 6. calculus of symmetric matrix with the distances to be used
# for k-NN predictions
kdist <- knn.dist(results)
# 7. prediction of test set
pred <- knn.predict(train, test, cl, kdist, k=k, agg.meth="
majority")
cat(pred, "\n")
if(cl[test]==pred){
  n_oks <- n_oks+1
  if(pred == "p")
    t_pos <- t_pos + 1
  else
    t_neg <- t_neg + 1
}
else {
  n_fails <- n_fails+1
  if (pred == "p")
    f_pos <- f_pos + 1
  else
    f_neg <- f_neg + 1
}
}
# 8. calculus of measures
bbb_p <- t_pos / (f_pos + t_pos)
bbb_n <- t_neg / (f_neg + t_neg)
accuracy <- n_oks / (n_oks + n_fails)
precision <- bbb_p
recall <- t_pos / (t_pos + f_neg)
F <- 2 / (1 / precision + 1 / recall)
Phi=(t_pos * t_neg - f_pos * f_neg) / sqrt((t_pos + f_pos) * (t_pos
+ f_neg) * (t_neg + f_pos) * (t_neg + f_neg))
cat(ndim, "kNN", namedist, k, accuracy, precision, recall, F, bbb_p,
bbb_n, Phi, t_pos, t_neg, f_pos, f_neg, "\n", sep=",")
# 9. write measures to file
cat(ndim, "kNN", namedist, k, accuracy, precision, recall, F, bbb_p,
bbb_n, Phi, t_pos, t_neg, f_pos, f_neg, "\n", sep=",", file="
output_kNN.txt", append = TRUE)
flush.console()
}

# run k_NN function: build string name of the different files with
# coordinates vectors, and for each file run the k_NN function with
# each of the dimensions and neighbors defined
run<-function() {
  metodos<-c("tanimoto+mwlogP", "tanimoto+mw", "tanimoto+logP", "
tanimoto", "cosine", "dice", "euclid", "forbes", "hamman", "
kulczynski", "manhattan", "matching", "pearson", "
rogers_tanimoto", "russell_rao", "simpson", "yule")
  for (met in metodos) {
    for (d in c(2,5,10,20,30,40,50)) {
      for (k in c(1,2,4,8)) {
        fname<-paste("cmdscaleK",d,"_",met,".txt", sep="")
        k_nn(fname,d,met,"dadosNP.csv",k)
      }
    }
  }
}

```

```

    }
  }
}
run()

# call k_NN function
k_nn("cmdscaleK30_pearson.txt", 30, "pearson", "dadosNP.csv", 4)

```

## A.6 Neural Networks

R script to implement a neural network, which also receives coordinates vectors representing chemical information present in molecules (section 4.3.3).

```

#script: NN analysis
# 1. function to perform the NN analysis, it receives the file with
#    vector of coordinates (file), the number of dimensions of the
#    vector (ndim), the file with classification (p or n) for each
#    molecule (file_classification), the number of iterations to perform
#    n-fold cross validation (n=10), and the number of neurons in
#    middle layer (size)
library(nnet)
cval_nnet<-function(file, ndim, namedist, file_classification, n, size,
  decay=1e-5, maxit=1000) {
  # 2. read coordinates vector
  file_input <- read.table(file,header=T,sep=",", dec=".")
  # 3. read file with classification for each molecule
  classif <- read.csv(file_classification, header = TRUE, sep = ";",
    quote="\"", dec=".")
  cl <- classif$np
  len <- nrow(file_input)
  smp <- sample(1:len,len)
  dat <- file_input[smp,]
  cls <- cl[smp]
  s_tpos <- 0; s_tneg <- 0; s_fpos <- 0; s_fneg <- 0; s_accuracy <- 0;
  s_precision <- 0; s_recall <- 0; s_F <- 0; s_Phi <- 0; s_bbb_n
  <- 0
  # 4. 10-fold cross validation
  for(i in 1:n) {
    # 5. randomly select 1/10 of the dat for test
    _sample <- seq(((len*0.1)*i-(len*0.1)+1), (len*0.1)*i) # amostra
    <- seq((62*i-62+1), 62*i)
    test <- dat[_sample,]
    train <- dat[-_sample,]
    targets <- class.ind(cls)
    # 6. build neural network
    network <- nnet(train, targets[-_sample,], size=size, rang=0.1,
      decay=decay, maxit=maxit, trace=FALSE, MaxNWts = 3000)
    print(predict(network,test))
    # 7. build confusion matrix
    conf_matrix <- table(max.col(targets[_sample,]), max.col(predict(
      network, test)))
    tpos <- conf_matrix[1,1]; tneg <- conf_matrix[2,2]; fpos <-
    conf_matrix[1,2]; fneg <- conf_matrix[2,1]
  }
}

```

```

total <- sum(conf_matrix)
# 8. calculus of measures for each iteration
accuracy <- (tpos + tneg) / total
precision <- tpos / (tpos + fpos)
recall <- tpos / (tpos + fneg)
F <- 2 / (1 / precision + 1 / recall)
Phi=(tpos * tneg - fpos * fneg) / sqrt((tpos + fpos) * (tpos +
    fneg) * (tneg + fpos) * (tneg + fneg))
bbb_n <- tneg / (fneg + tneg)
# 9. measures sum
s_tpos <- s_tpos + tpos
s_tneg <- s_tneg + tneg
s_fpos <- s_fpos + fpos
s_fneg <- s_fneg + fneg
s_accuracy <- s_accuracy + accuracy
s_precision <- s_precision + precision
s_recall <- s_recall + recall
s_F <- s_F + F
s_Phi <- s_Phi + Phi
s_bbb_n <- s_bbb_n + bbb_n
}
# 10. calculus of measures mean
m_tpos <- s_tpos / n
m_tneg <- s_tneg / n
m_fpos <- s_fpos / n
m_fneg <- s_fneg / n
m_accuracy <- s_accuracy / n
m_precision <- s_precision / n
m_recall <- s_recall / n
m_F <- s_F / n
m_Phi = s_Phi / n
m_bbb_p <- m_precision
m_bbb_n <- s_bbb_n / n
cat(ndim, "NN", namedist, size, m_accuracy, m_precision, m_recall,
    m_F, m_bbb_p, m_bbb_n, m_Phi, m_tpos, m_tneg, m_fpos, m_fneg,
    "\n", sep=",")
# 11. write measures to file
cat(ndim, "NN", namedist, size, m_accuracy, m_precision, m_recall,
    m_F, m_bbb_p, m_bbb_n, m_Phi, m_tpos, m_tneg, m_fpos, m_fneg,
    "\n", sep=",", file="output_nnet.txt", append = TRUE)
}

# run NN function: build string name of the different files with
# coordinates vectors, and for each file run the NN function with
# each of the dimensions and number of neuros in middle layer defined
run<-function() {
  metodos<-c("tanimoto+mwlogP", "tanimoto+mw", "tanimoto+logP", "
    tanimoto", "cosine", "dice", "euclid", "forbes", "hamman", "
    kulczynski","manhattan","matching", "pearson", "rogers_tanimoto
    ", "russell_rao", "simpson", "yule")
  for (met in metodos) {
    for (d in c(2, 5, 10, 20, 30, 40, 50)) {
      for (size in c(3,5,9,15,20,25,40)) {
        fname<-paste("cmdscaleK",d,"_",met,".txt", sep="")
        cval_nnet (fname,d,met,"dadosNP.csv",10,size)
      }
    }
  }
}

```

```
    }  
  }  
}  
run()  
  
# call NN function  
cval_nnet("cmdscaleK20_russell_rao.txt",20,"russell rao","dadosNP.csv"  
          ",10,5)
```

## A.7 Random Forests

R script to implement the Random Forest approach following the methodology of section 4.3.4.

```
library(randomForest)  
randomforest <- function(n_it, n_val){  
  dat<-read.table("rf_descriptors.txt")  
  # 1. molecules classification  
  bbb = dat[,1]  
  the_data<-1:nrow(dat)  
  # 2. assign array wts with probabilities for each molecule: if  
  # molecule does not cross the BBB (n) has a probability of 0.95,  
  # else has a probability of 0.05  
  wts<-c()  
  for( i in 1:nrow(dat)) {  
    if(dat[i,1]=='n') {  
      wts<-c(wts,0.95)  
    }  
    else {  
      wts<-c(wts,0.05)  
    }  
  }  
  s_tpos <- 0; s_tneg <- 0; s_fpos <- 0; s_fneg <- 0; s_accuracy <- 0;  
  s_precision <- 0; s_recall <- 0; s_F <- 0; s_Phi <- 0; s_bbb_n  
  <- 0  
  
  available_inds<-the_data  
  # 3. perform n_it-fold cross validation  
  for (i in 1:n_it){  
    # choose one of the lines between 4. or 5.  
    # 4. select validation set indices without considering prior  
    # probabilities  
    val_inds <- sample(available_inds,n_val)  
    # 5. select validation set indices considering prior  
    # probabilities  
    val_inds<-sample(available_inds, n_val, prob=wts, replace=TRUE)  
    # 6. select training set indices obtained by subtracting the  
    # validation set to all dataset  
    train_inds <- the_data[-val_inds]  
    available_inds<-available_inds[-val_inds]  
    # 7. select weights for training set  
    wts_train_set<-wts[train_inds]  
    # 8. select validation set from the previously defined indices
```

```

val_set<-dat[val_inds,]
# 9. perform oversampling on training set indices, considering
# probabilities
train_inds2<-sample(train_inds,10000,prob=wts_train_set, replace=
TRUE)
# 10. select training set from the indices defined in 9.
train_set<-dat[train_inds2,]
# 11. random forest construction with training set
bbb_rf <- randomForest(bbb ~ ., data = train_set, importance=TRUE
)
# 12. predict classification for validation set
bbb_pred <- predict(bbb_rf, val_set)
tabela <- table(observed = dat[val_inds, "bbb"] , predicted =
bbb_pred)
# 13. write importance variable to csv file
out_importance <- paste("imp_it",i,".csv", sep="")
importance <- bbb_rf$importance
write.csv(importance, file = out_importance)

tpos <- tabela[2,2]; tneg <- tabela[1,1]; fpos <- tabela[1,2];
fneg <- tabela[2,1]
total <- sum(tabela)
# 14. calculus of measures for each iteration
accuracy <- (tpos + tneg) / total
precision <- tpos / (tpos + fpos)
recall <- tpos / (tpos + fneg)
F <- 2 / (1 / precision + 1 / recall)
Phi <- (tpos * tneg - fpos * fneg) / sqrt((tpos + fpos) * (tpos +
fneg) * (tneg + fpos) * (tneg + fneg))
bbb_n <- tneg / (fneg + tneg)
bbb_p <- precision
# 15. measures sum
s_tpos <- s_tpos + tpos
s_tneg <- s_tneg + tneg
s_fpos <- s_fpos + fpos
s_fneg <- s_fneg + fneg
s_accuracy <- s_accuracy + accuracy
s_precision <- s_precision + precision
s_recall <- s_recall + recall
s_F <- s_F + F
s_Phi <- s_Phi + Phi
s_bbb_n <- s_bbb_n + bbb_n
cat(accuracy, precision, recall, F, bbb_p, bbb_n, Phi, tpos, tneg
, fpos, fneg, "\n", sep=",")
}
# 16. calculus of measures mean
n = n_it
m_tpos <- s_tpos / n
m_tneg <- s_tneg / n
m_fpos <- s_fpos / n
m_fneg <- s_fneg / n
m_accuracy <- s_accuracy / n
m_precision <- s_precision / n
m_recall <- s_recall / n
m_F <- s_F / n
m_Phi = s_Phi / n

```

```
m_bbb_p <- m_precision
m_bbb_n <- s_bbb_n / n
cat(m_accuracy, m_precision, m_recall, m_F, m_bbb_p, m_bbb_n, m_Phi,
    m_tpos, m_tneg, m_fpos, m_fneg, "\n", sep=",")
}
randomforest(10,95)
```

# Appendix B

## Molecular descriptors

In this appendix is comprised the different classes and respective molecular descriptors that are presented in several studies reported in literature and in the present work.

Table B.1: Classes and molecular descriptors used in studies reported in literature and in this work. Adapted from Li et al. [2005b].

Class		Molecular descriptors
simpler	molecular properties	LogP, molecular weight, average molecular weight, numbers of rings, number of bonds in rings, rotatable bonds, H-bond donors, and H-bond acceptors, atom counts, bond counts
	molecular connectivity and shape	molecular connectivity indices, valence, path, cluster, path/cluster molecular connectivity indices, kappa molecular shape indices, flexibility index, differential connectivity indices, Kier and Hall connectivity, graph's radius and diameter, counts of different vertices, counts of paths and edges between different types of vertices
	electrotopological state	electrotopological state indices, topological, and atom type electrotopological state indices, Wiener index, Platt index, Shannon and Bonchev-Trinajstic information indices, centric index, Altenburg index, Balaban index, Harary number, Schultz index, PetitJohn R2 index, PetitJohn D2 index, mean distance index, PetitJohn I2 index, information Wiener, Balaban RMSD index, graph distance index
quantum	chemical properties	polarizability index, hydrogen bond acceptor basicity (covalent HBAB), hydrogen bond donor acidity (covalent HBDA), molecular dipole moment, absolute hardness, softness, ionization potential, electron affinity, chemical potential, electronegativity index, electrophilicity index, most positive charge on H, C, N, O atoms, most negative charge on H, C, N, O atoms, most positive and negative charge in a molecule, sum of squares of charges on H,C,N,O and all atoms, mean of positive charges, mean of negative charges, mean absolute charge, relative positive charge, relative negative charge
	geometrical properties	length vectors (longest distance, longest third atom, 4th atom), molecular van der Waals volume, solvent accessible surface area, molecular surface area, van der Waals surface area, polar molecular surface area, sum of solvent accessible surface areas of positively charged atoms, molecular rugosity, molecular globularity, hydrophilic region, hydrophobic region, capacity factor, hydrophilic-hydrophobic balance, hydrophilic intery moment, hydrophobic intery moment, amphiphilic moment
	structural properties	molecular fingerprints (fp1 to fp1023), presence/absence of functional groups





# Appendix C

## Results

This appendix holds some of the results, that are part of the analysis and discussion of the thesis but for being less important are recorded here. Figure C.1 represent the variables importance analysis, made in the context of the random forest approach. It shows the 14 most important variables, that contributes for a higher decrease in accuracy in case of variable deletion, and 9 of the less important variables. Next figure shows all the less important variables collected from each iteration and they frequency among these. For example, variables with a frequency of 10 means that are present in all the iterations.


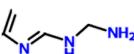

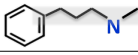
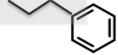

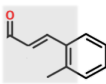


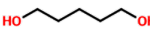
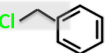
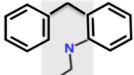
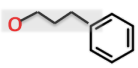
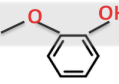
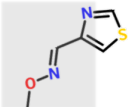
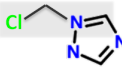
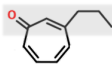
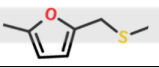

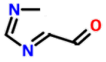
More important	Less important
X8.1 	fp 662 
X6.3 	
fp 532 	fp 749 
MW	
fp 314 	fp 1022 
fp 516 	
fp 330 	fp 603 
fp 653 	fp 963 
fp 872 	
fp 624 	fp 968 
fp 47 	fp 876 
fp 62 	fp 537 
NBRing	fp 1004 
AMW	

Figure C.1: Importance variables analysis and its molecular structures.

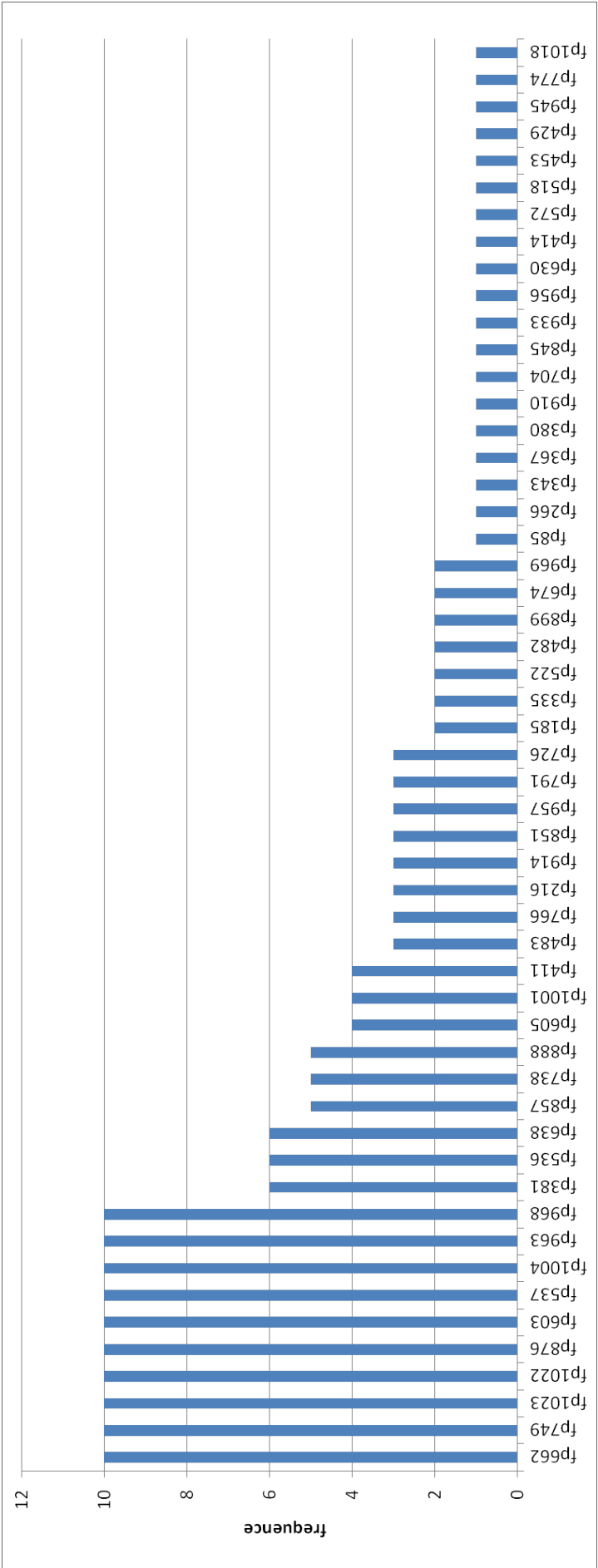


Figure C.2: Frequence of the twenty less important variables.

# Glossary

ADME	Absorbed Distributed Metabolized Excreted
AMW	Average Molecular Weight
BBB	Blood-Brain Barrier
BBB+	passing BBB molecules
BBB-	non-passing BBB molecules
BCSFB	Blood-Cerebrospinal Fluid Barrier
CMDS	Classical Multidimensional Scaling
CNS	Central Nervous System
CSF	Cerebrospinal Fluid
CV	Cross-Validation
DT	Decision Tree
FN	False Negatives
FP	False Positives
ISF	Brain Interstitial Fluid
k-NN	k-Nearest Neighbor
LOOCV	Leave-One-Out Cross-Validation
MCC	Matthews Correlation Coefficient
MW	Molecular Weight
NBRing	Number of Bonds in Rings
NN	Neural Network
NRing	Number of Rings
PCoA	Principal Coordinates Analysis
RF	Random Forest

---

SMILES	Simplified Molecular Input Line Entry Specification
SWIG	Simplified Wrapper and Interface Generator
TN	True Negatives
TP	True Positives

# Bibliography

- N. Joan Abbott, Lars Ronnback, and Elisabeth Hansson. Astrocyte-endothelial interactions at the blood-brain barrier. *Nature Reviews Neuroscience*, 2006.
- Marc Adenot and Roger Lahana. Blood-Brain Barrier Permeation Models: Discriminating between Potential CNS and Non-CNS Drugs Including P-Glycoprotein Substrates. *J. Chem. Inf. Comput. Sci.*, 2004.
- Guy W. Bemis Ajay and Mark A. Murcko. Designing Libraries with CNS Activity. *J. Med. Chem.*, 1999.
- Mohammad S. Alavijeh, Mansoor Chishty, M. Zeeshan Qaiser, and Alan M. Palmer. Drug Metabolism and Pharmacokinetics, the Blood-Brain Barrier, and Central Nervous System Drug Discovery. *NeuroRx: The Journal of the American Society for Experimental NeuroTherapeutics*, 2005.
- Alzheimer’s Drug Discovery Foundation. Drug discovery process, 23-07-2011. [http://www.alzdiscovery.org/pdf/Pipeline\\_Report.pdf](http://www.alzdiscovery.org/pdf/Pipeline_Report.pdf).
- Sylvain Arlot and Alain Celisse. A survey of cross-validation procedures for model selection. *Statistics Surveys*, 2010.
- Peter Armitage and Theodore Colton. *Encyclopedia of Biostatistics*. Wiley Publishing, Indianapolis, Indiana, 2005.
- Open Babel. Fingerprint format (fpt), 21-09-2010. <http://openbabel.org/fingerprint-format>.
- Open Babel. Molecular fingerprints and similarity searching, 4-11-2010. <http://openbabel.org/docs/dev/Features/Fingerprints.html>.
- Pierre Baldi, Soren Brunak, Yves Chauvin, Claus A. F. Andersen, and Henrik Nielsen. Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics*, 2000.
- David M. Beazley. SWIG: An Easy to Use Tool for Integrating Scripting Languages with C and C++. *4th Annual Tcl/Tk Workshop, Monterey, CA*, 1996.

- David J. Begley, Michael W. Bradbury, and Jorg Kreuter. *The Blood-Brain Barrier and Drug Delivery to the CNS*. Marcel Dekker, Inc., 2000.
- Leo Breiman and Adele Cutler. Random forests, 18-5-2011. [http://www.stat.berkeley.edu/~breiman/RandomForests/cc\\_home.htm](http://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm).
- Nathan Brown. Chemoinformatics - An Introduction for Computer Scientists. *ACM Computing Surveys*, 2009.
- Oliviero Carugo. Detailed estimation of bioinformatics prediction reliability through the Fragmented Prediction Performance Plots. *BMC Bioinformatics*, 2007.
- Chao Chen, Andy Liaw, and Leo Breiman. Using Random Forest to Learn Imbalanced Data. *Department of Statistics, University of California, Berkeley*, 2004.
- Marilyn J. Cipolla. *The Cerebral Circulation*. Morgan & Claypool Life Sciences, 2009.
- NCI/CADD CIR. Chemical identifier resolver beta 4, 18-5-2011. <http://cactus.nci.nih.gov/chemical/structure>.
- Project CRAN-R. randomforest: Breiman and cutler's random forests for classification and regression, 10-4-2011. <http://cran.r-project.org/web/packages/randomForest/randomForest.pdf>.
- Project CRAN-R. knnflex: A more flexible knn, 12-10-2010. <http://cran.r-project.org/web/packages/knnflex/knnflex.pdf>.
- Project CRAN-R. Fit neural networks, 18-10-2010. <http://stat.ethz.ch/R-manual/R-devel/library/nnet/html/nnet.html>.
- Project CRAN-R. Classical (metric) multidimensional scaling, 29-10-2010. <http://stat.ethz.ch/R-manual/R-patched/library/stats/html/cmdscale.html>.
- Patrizia Crivori, Gabriele Cruciani, Pierre-Alain Carrupt, and Bernard Testa. Predicting Blood –Brain Barrier Permetation from Three-Dimensional Molecular Structure. *J. Med. Chem.*, 2000.
- Gabriele Cruciani, Manuel Pastor, and Wolfgang Guba. VolSurf: a new tool for the pharmacokinetic optimization of lead compounds. *European Journal of Pharmaceutical Sciences*, 2000.
- dalke scientific. Generating molecular fingerprints with openbabel, 6-7-2011. [http://www.dalkescientific.com/writings/diary/archive/2008/06/27/generating\\_fingerprints\\_with\\_openbabel.html](http://www.dalkescientific.com/writings/diary/archive/2008/06/27/generating_fingerprints_with_openbabel.html).

- Daylight. Clogp reference manual, 2-10-2011. <http://www.daylight.com/dayhtml/doc/clogp/index.html#PCMsc5>.
- Daylight. Fingerprints - screening and similarity, 3-12-2010. <http://www.daylight.com/dayhtml/doc/theory/theory.finger.html>.
- Rolf Dermietzel, David C. Spray, and Maiken Nedergaard. *Blood-Brain Interface: From Ontogeny to Artificial Barriers*. WILEY-VCH, 2006.
- Eric Deconinck Dmitry A. Konovalov, Danny Coomans and Yvan Vander Heyden. Benchmarking of QSAR Models for Blood-Brain Barrier Permeation. *J. Chem. Inf. Model.*, 2007.
- Scott Doniger, Thomas Hofmann, and Joanne Yeh. Predicting CNS Permeability of Drug Molecules: Comparison of Neural Network and Support Vector Machine Algorithms. *Journal of Computational Biology*, 2000.
- Robert Gentleman. *R Programming for Bioinformatics*. Chapman & Hall/CRC computer sciences and data analysis series, 2009.
- Robert Gentleman, Vincent Carey, Wolfgang Huber, Rafael Irizarry, and Sandrine Dudoit. *Bioinformatics and Computational Biology Solutions using R and Bioconduct*. Springer, 2002.
- Jeffrey W. Godden, Ling Xue, and Jürgen Bajorath. Combinatorial Preferences Affect Molecular Similarity/Diversity Calculations Using Binary Fingerprints and Tanimoto Coefficients. *J. Chem. Inf. Comput. Sci.*, 2000.
- Jeff Heaton. *Introduction to Neural Networks for Java*. Heaton Research, 2008.
- George Hripcsak and Adam S. Rothschild. Agreement, the F-Measure, and Reliability in Information Retrieval. *Journal of the American Medical Informatics Association*, 2005.
- I. T. Jolliffe. *Principal Component Analysis*. Springer Series in Statistics, 2002.
- Anthony King. Breaking through the barrier. *Chemistry World*, 2011.
- H. Li, C. Y. Ung, C. W. Yap, Y. Xue, Z. R. Li, Z. W. Cao, and Y. Z. Chen. Prediction of Genotoxicity of Chemical Compounds by Statistical Learning Methods. *Chem. Res. Toxicol.*, 2005a.
- Hu Li, Chun Wei Yap, Choong Yong Ung, Ying Xue, Zhi Wei Cao, and Yu Zong Chen. Effect of Selection of Molecular Descriptors on the Prediction of Blood –Brain Barrier Penetrating and Nonpenetrating Agents by Statistical Learning Methods. *J. Chem. Inf. Model.*, 2005b.

- Ambikanandan Misra, Ganesh S., and Aliasgar Shahiwala. Drug delivery to the central nervous system: a review. *J Pharm Pharmaceut Sci*, 2003.
- Group NCI/CADD. Cadd group chemoinformatics tools and user services, 18-5-2011. <http://cactus.nci.nih.gov/>.
- Noel M O'Boyle, Chris Morley, and Geoffrey R Hutchison. Pybel: a Python wrapper for the OpenBabel cheminformatics toolkit. *Chemistry Central Journal*, 2008.
- OpenSMILES. Opensmiles specification, 12-05-2011. <http://opensmiles.org/spec/open-smiles.html>.
- William M. Pardridge. *Introduction to the Blood-Brain Barrier: Methodology, biology and pathology*. Cambridge University Press, 1998.
- William M. Pardridge. Blood-Brain Barrier Drug Targeting: The Future of Brain Drug Development. *molecular interventions*, 2003.
- William M. Pardridge. The Blood-Brain Barrier: Bottleneck in Brain Drug Development. *The Journal of the American Society for Experimental NeuroTherapeutics*, 2005.
- Python. Python programming language, 23-10-2010. <http://www.python.org/>.
- Payam Refaeilzadeh, Lei Tang, and Huan Liu. Cross-validation. In M. Tamer Ã-zsu and Ling Liu, editors, *Encyclopedia of Database Systems*. Springer, 2009.
- M. Filipe Santos and Carla Azevedo. *Data Mining - Descoberta de conhecimento em bases de dados*. FCA - Editora de Informática, Lda, 2005.
- Carolin Strobl, Anne-Laure Boulesteix, Achim Zeileis, and Torsten Hothorn. Bias in Random Forest Variable Importance Measures: Illustrations, Sources and a Solution. *Report Series / Department of Statistics and Mathematics, WU Vienna University of Economics and Business, Vienna*, 2006.
- Hyontai Sug. A comparison of RBF networks and random forest in forecasting ozone day. *International Journal of Mathematics and Computers in Simulation*, 2010.
- Vladimir Svetnik, Andy Liaw, Christopher Tong, J. Christopher Culberson, Robert P. Sheridan, and Bradley P. Feuston. Random Forest: A Classification and Regression Tool for Compound Classification and QSAR Modeling. *J. Chem. Inf. Comput. Sci.*, 2003.
- M. W. B. Trotter, B. F. Buxton, and S. B. Holden. Support vector machines in combinatorial chemistry. *Meas. Control*, 2001.



- Katya Tsaïoun, Michel Bottlaender, and Aloise Mabondzo. ADDME - Avoiding Drug Development Mistakes Early: central nervous system drug discovery perspective. *BMC Neurology*, 2009.
- David Weininger. SMILES, a Chemical Language and Information System. 1. Introduction to Methodology and Encoding Rules. *J. Chem. Inf. Comput. Sci.*, 1988.
- Scott A. Wildman and Gordon M. Crippen. Prediction of physicochemical parameters by atomic contributions. *J. Chem. Inf. Comput. Sci.*, 1999.
- Shuxiang Xu and Ling Chen. A Novel Approach for Determining the Optimal Number of Hidden Layer Neurons for FNNs and Its Application in Data Mining. *5th International Conference on Information Technology and Applications*, 2008.
- Liyang Zhang, Hao Zhu, Tudor I. Oprea, Alexander Goldbraikh, and Alexander Tropsha. QSAR Modeling of the Blood-Brain Barrier Permeability for Diverse Organic Compounds. *Pharmaceutical Research*, 2008.
- Yuan H. Zhao, Michael H. Abraham, Adam Ibrahim, Paul V. Fish, Susan Cole, Mark L. Lewis, Marcel J. de Groot, and Derek P. Reynolds. QSAR Modeling of the Blood-Brain Barrier Permeability for Diverse Organic Compounds. *J. Chem. Inf. Model.*, 2007.
- Alain F. Zuur, Elena N. Ieno, and Graham M. Smith. *Analysing Ecological Data*. Springer, 2007.