

University of Maribor
Faculty of Electrical Engineering and Computer Science

DOCTOR OF PHILOSOPHY THESIS

**PROTEIN SOLUBILITY CLASSIFICATION IN
BIOMEDICAL CONCEPT SPACE**

DOKTORSKA DISERTACIJA

**KLASIFIKACIJA TOPNOSTI PROTEINOV V
PROSTORU BIOMEDICINSKIH KONCEPTOV
TEKSTOVNE ANALIZE**

November, 2011

Simon Kocbek

University of Maribor
Faculty of Electrical Engineering and Computer Science

DOCTOR OF PHILOSOPHY THESIS

**PROTEIN SOLUBILITY CLASSIFICATION IN
BIOMEDICAL CONCEPT SPACE**

DOKTORSKA DISERTACIJA

**KLASIFIKACIJA TOPNOSTI PROTEINOV V
PROSTORU BIOMEDICINSKIH KONCEPTOV
TEKSTOVNE ANALIZE**

Author: Simon Kocbek

Supervisor: prof. dr. Peter Kokol

Co-Supervisor: assoc. prof. dr. Maria Garcia de la Banda,

Monash University

November, 2011

UDK: 004.8:004.62:577.112.089(043.2)



Univerza v Mariboru

Maribor, 25. 1. 2011
Številka: DR 6/2011/425-MGM

Na osnovi 287., 140., 142. in 144. člena Statuta Univerze v Mariboru (Statut UM-UPB8, Ur. l. RS, št. 1/2010) ter sklepa 38. redne seje Senata Univerze v Mariboru z dne 25. 1. 2011 v zvezi z vlogo doktorskega kandidata Simona Kocbeka za sprejem odločitve o predlagani temi doktorske disertacije in mentorja,

izdajam naslednji

SKLEP

Odobri se tema doktorske disertacije Simona Kocbeka s Fakultete za elektrotehniko, računalništvo in informatiko z naslovom »Klasifikacija topnosti proteinov v prostoru biomedicinskih konceptov tekstovne analize«. Za mentorja se imenuje red. prof. dr. Peter Kokol, za somentorico pa izr. prof. dr. Maria Garcia de la Banda. Kandidat mora članici predložiti izdelano doktorsko disertacijo v zadostnih izvodih najpozneje do 24. 1. 2015.

Obrazložitev:

Kandidat Simon Kocbek je dne 30. 6. 2010 na Fakulteti za elektrotehniko, računalništvo in informatiko vložil vlogo za potrditev teme doktorske disertacije z naslovom »Klasifikacija topnosti proteinov v prostoru biomedicinskih konceptov tekstovne analize«. Za mentorja je bil predlagan red. prof. dr. Peter Kokol, za somentorico pa izr. prof. dr. Maria Garcia de la Banda.

Senat Fakultete za elektrotehniko, računalništvo in informatiko je na osnovi pozitivnega mnenja komisije za oceno teme doktorske disertacije, ki je ugotovila, da kandidat izpolnjuje pogoje za pridobitev doktorata znanosti, in ocenila, da je predlagana tema ustrezna, sprejel pozitivno mnenje in poslal predlog teme doktorske disertacije s predlogom mentorja in somentorice v odobritev Senatu univerze.

Senat Univerze v Mariboru je po proučitvi vloge in na osnovi določil Statuta Univerze v Mariboru sprejel svojo odločitev o predlagani temi doktorske disertacije in imenoval mentorja in somentorice, kot izhaja iz izreka.

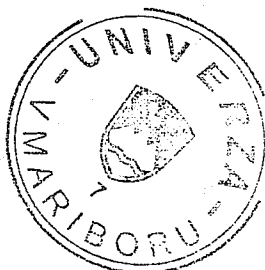
V skladu s 144. členom Statuta Univerze v Mariboru mora kandidat za pridobitev doktorata znanosti najpozneje v štirih letih od dneva izdaje tega sklepa, članici predložiti izdelano doktorsko disertacijo v zadostnih izvodih. Kandidatu je bil določen rok glede na datum sprejetja teme na pristojnem organu.

Pouk o pravnem sredstvu:

Zoper ta sklep je možna pritožba na Senat Univerze v Mariboru v roku 8 dni od prejema tega sklepa.

Obvestiti:

1. Kandidata.
2. Fakulteto.
3. Arhiv.



Rektor:
Prof. dr. Ivan Rozman



HR EXCELLENCE IN RESEARCH

I hereby certify that the work presented in this thesis is my own work and that to the best of my knowledge it is original except where indicated by reference to other authors.

Izjavljam, da sem delo napisal samostojno. Poglavlja in deli, ki se nanašajo na dela drugih avtorjev, so navedeni in ustrezno referencirani.

Simon Kocbek

Protein solubility classification in biomedical concept space

UDK: 004.8:004.62:577.112.089(043.2)

Keywords:

machine learning, protein solubility classification, biomedical concept, feature selection, text mining

Abstract:

Proteins are an essential part of every organism and each protein has its own function, which depends on the protein's structure. The latter is an important research topic and researchers often isolate proteins from complex mixtures to study their structures. The isolation process is in many ways influenced by the protein's solubility since insoluble proteins are usually harder to isolate than soluble ones. In addition, low protein solubility has been linked to different diseases. For these reasons, researchers often wish to identify which proteins are more likely to be soluble. As a result, several protein solubility classification algorithms have been proposed. Roughly speaking, these algorithms take a set of soluble and insoluble proteins as an input, learn their differences and produce a classifier that can be used to predict solubility for new proteins.

In this thesis we propose a new method for protein solubility classification, which uses text mining techniques to define protein attributes. This new method extracts biomedical knowledge from scientific literature and presents this knowledge in the form of so called biomedical concept attributes. These attributes present a novel approach of describing proteins in the classification process, since today's state-of-the-art classification methods use mostly attributes derived from the protein's sequence. To evaluate the new method, this thesis describes the classification scheme for an empirical study which measures the impact of the new attributes on the protein solubility classification. In the study, the twenty most common sequence derived attribute datasets are analysed, to which we gradually add five types of biomedical concept attributes. We measure the performance of the classifiers obtained by these attribute datasets.

As a result, this thesis introduces several original scientific contributions. First of all, an analysis of protein databases that contain information about protein solubility is performed. Secondly, the method for extracting biomedical concept attributes is presented. Next, an

original comparison of methods that use biomedical concept attributes with those that use only sequence-derived attributes is performed. The thesis demonstrates that the new attributes increase the performance of some classifiers. Finally, it identifies types of words and word associations from the medical literature that are associated with protein solubility.

Klasifikacija topnosti proteinov v prostoru biomedicinskih konceptov tekstovne analize

UDK: 004.8:004.62:577.112.089(043.2)

Ključne besede:

strojno učenje, klasifikacija topnosti proteinov, biomedicinski koncept, izbira atributov, tekstovno rudarjenje

Razširjen povzetek:

Proteini so pomemben del vsakega organizma in imajo številne pomembne funkcije, katere so v veliki meri odvisne od strukture proteina. Zadnja je mnogokrat predmet raziskav, kjer strokovnjaki izolirajo posamezen protein in proučijo njegove strukturne lastnosti. Na proces izolacije proteina v veliki meri vpliva njegova topnost, saj je protein z nizko stopnjo topnosti zelo težko izolirati. Prav tako so netopni proteini razlog za nekatere pomembne bolezni. Zaradi teh razlogov je analiza topnih proteinov pomembna naloga različnih biofizičnih raziskav. Postopek identifikacije topnih proteinov je običajno izveden s pomočjo poskusov v bioloških laboratorijih. Problem nastopi, ker so ti poskusi velikokrat neuspešni. Zato želijo strokovnjaki velikokrat vnaprej vedeti, kateri proteini imajo več možnosti za visoko stopnjo topnosti. Posledično so se razvile številne metode, ki uporabljajo tehnike nadzorovanega strojnega učenja za klasifikacijo topnosti proteinov. Te metode klasificirajo proteine v topne in netopne ter se uporabljajo za napovedovanje topnosti za nove primerke.

V preteklosti so raziskovalci uporabili različne statistične metode in metode strojnega učenja na različnih podatkovnih bazah z namenom, da zgradijo uspešen klasifikator topnosti proteinov. Prvi tak poizkus sta opravila Wilkinson in Harrison v letu 1991 [31], metoda pa je bila izboljšana leta 1999 [33]. V obeh primerih je šlo za statistično analizo zgradbe proteinov, ki so topni ali pa agregirajo v neločljive gmote. Uspešnost napovedi topnosti pri teh začetnih metodah je bila 88%. Čeprav je ta rezultat dokaj visok, je glavna pomanjkljivost obeh metod, da sta uporabili majhno množico proteinov za oceno natančnosti klasifikacije in rezultatov ni mogoče posplošiti na večje množice. Kasneje, leta 2000, so raziskovalci uporabili različne metode strojnega učenja kot so odločitvena drevesa, Naivni Bayes in metoda podpornih vektorjev [13]. Rezultati so pokazali, da ni bistvenih razlik v natančnosti klasifikacije med

temi metodami, saj so vse klasificirale pravilno 66% proteinov. O podobni uspešnosti so poročali tudi drugi avtorji [14, 32], ki so uporabljali metode odločitvenih dreves. Leta 2007 so raziskovalci uporabili izboljšane metode podpornih vektorjev in uspeli povišati natančnost klasifikacije na 72% [29]. Kasneje so dodali še algoritem Naivni Bayes kot sekundarni klasifikator in kljub isti natančnosti v primerjavi s predhodno metodo uspeli povišati rezultat v ostalih metrikah za merjenje uspešnosti klasifikatorja (npr. AUC in MCC) [30]. V letu 2009 sta bila predstavljena dva algoritma, ki sta bazirala na metodi podpornih vektorjev, ampak sta bila uporabljena na različnih podatkovnih bazah. Njuna uspešnost je bila 74% [34] in 80% [73].

Pri podrobni analizi predhodnih del smo identificirali naslednje tri bistvene dejavnike, ki vplivajo na uspešnost klasifikacije topnosti proteinov: (a) vrsta metode strojnega učenja za gradnjo klasifikatorja, (b) kvaliteta učne množice in (c) atributi, ki opisujejo proteine. Ugotovili smo, da se je metoda podpornih vektorjev izkazala kot najuspešnejša pri večini del, ter da ima podatkovna baza iz leta 2009 [73] najbolj natančne informacije o topnosti proteinov. Ugotovili smo tudi, da so vse predhodne metode omejene z naborom atributov, ki so izračunani iz sekvence proteinov.

V disertaciji tako predlagamo novo metodo za klasifikacijo topnosti proteinov, ki s pomočjo tehnik tekstovnega rudarjenja izlušči medicinsko znanje iz strokovne literature in ga predstavi v obliki atributov. Te attribute poimenujemo atributi biomedicinskih konceptov in predstavljajo novost na področju klasifikacije topnosti proteinov. Do sedaj uporabljene metode so namreč omejene z uporabo atributov, ki so večinoma izpeljani le iz sekvence proteina.

Hipoteza, ki jo zagovarjamo je naslednja:

Z opisom proteinov v prostoru biomedicinskih konceptov in uvedbo novih atributov, ki ne temeljijo le na primarni strukturi proteina ter uporabo naprednih metod strojnega učenja, lahko zgradimo uspešnejši klasifikator za napovedovanje topnosti proteinov v primerjavi z obstoječimi metodami.

Hipotezo smo razdelili v naslednja dva sklopa hipotez:

Hipoteza 1:

Do sedaj uporabljeni atributi proteinov za napovedovanje topnosti ne nosijo dovolj informacij za optimalno klasifikacijo.

Hipoteza 2:

Z združitvijo že obstoječe množice najpomembnejših atributov za napovedovanje topnosti proteinov in najpomembnejših atributov iz prostora biomedicinskih konceptov, lahko izboljšamo uspešnost klasifikatorja za nekatere metode strojnega učenja.

Podane hipoteze dokazujemo z razvojem predlagane klasifikacijske sheme in njeno primerjavo z obstoječimi metodami. Primerjane so naslednje tri tehnike strojnega učenja: Naivni Bayes (angl. Naïve Bayes), odločitvena drevesa (angl. Decision Trees) in metoda podpornih vektorjev (angl. Support Vector Machines). Rezultate primerjamo s tremi različnimi metrikami, prav tako pa izvedemo Wilcoxonov statistični test predznačenih rangov, s katerim preverimo statistično signifikanco dobljenih rezultatov. Izkaže se, da atributi biomedicinskih konceptov izboljšajo rezultate pri dveh izmed skupno treh metod.

V okviru disertacije podamo tudi številne znanstvene prispevke. Predlagana je metoda za ekstrakcijo atributov biomedicinskih konceptov iz strokovne literature na podlagi imena oziroma identifikacijske številke proteina. Nadalje ponudimo originalno primerjavo metod, ki uporabljajo nove attribute, z metodami, ki ponujajo že uveljavljene attribute izpeljane iz sekvence proteina. Kot se pokaže v disertaciji, novi atributi doprinesejo k uspešnosti klasifikacije topnosti proteinov. Podan je tudi algoritem za implementacijo najuspešnejšega klasifikatorja z atributi biomedicinskih konceptov. Zadnji prispevek vključuje novo medicinsko znanje, ki ponudi indice o tem, katere skupine besed in besednih zvez iz strokovne literature so najbolj povezane s topnostjo proteinov.

Disertacija je sestavljena iz skupno osem poglavij, katera podrobno predstavijo teoretično ozadje področij, kot so nadzorovano strojno učenje, tekstovno rudarjenje ter struktura in topnost proteinov. Uvodno poglavje predstavi problem, predlagano rešitev in hipoteze. Predstavljeni so tudi pričakovani znanstveni prispevki.

Poglavje 2 predstavi potrebno teoretično ozadje metod strojnega učenja, kjer se osredotočimo na algoritme nadzorovanega učenja. Natančneje, predstavljene so štiri klasifikacijske metode, kjer vsaka izmed njih uporablja unikatne tehnike za gradnjo klasifikatorjev. Predstavljene so tudi prednosti in slabosti vsake izmed metod in problemi, kot sta preveliko in premalo prileganje učnim podatkom (angl. over-fitting in under-fitting). Poglavje se zaključi z opisom rešitev za te probleme, kot so metode za nastavljanje parametrov klasifikatorja in metode za izbiro najpomembnejših atributov za podatke v učni množici.

Prvi del tretjega poglavja predstavi potrebno teoretično ozadje o proteinih. Opisane so štiri stopnje strukture proteina, kjer so predstavljene tudi povezave med njimi. Prav tako sta

predstavljeni dve različni vrsti topnosti proteinov. Drugi del tretjega poglavja opiše analizo dveh podatkovnih baz, ki sta v preteklosti bili uporabljeni za gradnjo klasifikatorjev za topnost proteinov. Poglavje se zaključuje s podrobnim opisom podatkovne baze eSol, ki je bila uporabljena za gradnjo klasifikatorja v okviru doktorske disertacije.

Poglavje 4 predstavi tehnike tekstovnega rudarjenja, ki jih lahko uporabimo za iskanje biomedicinskih konceptov. Prvi del tega poglavja se osredotoči na metode za iskanje uporabnih vzorcev v nestrukturiranem tekstu. Opisani so procesi in koraki kot so tokenizacija (angl. tokenization), segmentacija (angl. sentence boundary detection, segmentation) in korenjenje (angl. stemming). Drugi del tega poglavja predstavi definicijo atributov biomedicinskih konceptov v kontekstu doktorske disertacije in opiše orodje FACTA, ki ga uporabimo za iskanje biomedicinskih konceptov v znanstveni literaturi. Predstavljenih je naslednjih pet skupin biomedicinskih konceptov: bolezen, zdravilo, simptom, spojina in encim.

Poglavje 5 opiše eksperimentalno okolje za empirično primerjavo predlagane metode. Podrobno je opisana klasifikacijska shema za gradnjo klasifikatorjev s tremi tehnikami, prav tako pa je predstavljena evalvacija zgrajenih klasifikatorjev. Osredotočimo se predvsem na opis metod za izračun in izbiro atributov. Izvedena je empirična primerjava dvajsetih baz sekvenčnih atributov, ki jim postopoma dodajamo nove attribute in spremljamo doprinose k uspešnosti treh pogosto uporabljenih klasifikacijskih metod. Dodatno je opisano tudi okolje Weka, ki je uporabljeno za implementacijo klasifikacijske sheme.

V Poglavju 6 so predstavljeni rezultati za vsako izmed klasifikacijskih metod, ki so bile uporabljene v eksperimentih. Izvedena je primerjava med klasifikatorji, ki uporabljajo le attribute izpeljane iz sekvenc proteinov in klasifikatorji, ki uporabljajo tudi attribute biomedicinskih konceptov. Na koncu tega poglavja je prav tako predstavljen klasifikator topnosti proteinov z največjo natančnostjo.

Poglavje 7 interpretira rezultate in analizira kako ti vplivajo na postavljene hipoteze. Prikazan je postopek potrditve hipoteze in zaključek, kjer je razvidno, da atributi biomedicinskih konceptov pripomorejo k izboljšanju uspešnosti klasifikatorjev za topnost proteinov. V tem poglavju so prav tako opisani izvorni znanstveni prispevki doktorske disertacije.

Poglavje 8 zaključuje doktorsko disertacijo s kratkim povzetkom, predstavi nerešene probleme in nakaže smernice za prihodnja raziskovanja.

Acknowledgments

I would like to thank my supervisor prof. dr. Peter Kokol for his support during my postgraduate studies.

I would like to thank my co-supervisor assoc. prof. dr. Maria Garcia de la Banda for introducing me to proteins and her guidance in the phase of writing this thesis.

I would also like to thank my colleagues Gregor and Igor at the Research Institute for their support and fun time.

A special thank you goes to my family for their love, patience and support during my postgraduate studies.

Finally, I gratefully acknowledge the support from the Slovenian Research Agency for its grant which helped me to accomplish my research.

Simon Kocbek

CONTENTS

1.	Introduction.....	1
1.1	Motivation.....	1
1.2	Goals and contributions.....	3
1.2.1	Hypothesis and research methodology.....	3
1.2.2	Expected scientific contributions.....	4
1.3	Thesis structure.....	5
2.	Machine learning.....	7
2.1	Introduction.....	7
2.2	Basic terminology.....	8
2.2.1	Input of machine learning algorithms.....	8
2.2.2	Output of machine learning algorithms.....	9
2.2.3	Types of machine learning algorithms.....	9
2.2.4	Training and test set.....	10
2.3	Supervised machine learning.....	11
2.3.1	Attributes in supervised machine learning.....	12
2.4	Methods for building classifiers.....	15
2.4.1	K-nearest neighbours.....	15
2.4.2	Naïve Bayes.....	16
2.4.3	Decision trees.....	20
2.4.4	Support Vector Machines.....	23
2.5	Parameters of machine learning techniques.....	26
2.5.1	Over-fitting and under-fitting.....	26
2.5.2	Optimizing parameters.....	27
2.6	Evaluating machine learning algorithms.....	28
2.6.1	Evaluation of a single classifier.....	28
2.6.2	Comparing the performance of classifiers on multiple datasets.....	31
3.	Proteins.....	35
3.1	Introduction.....	35
3.2	Protein structure.....	35

3.3	Protein solubility	38
3.3.1	Gathering soluble proteins.....	39
3.3.2	The eSol database	40
4.	Text mining and biomedical concepts.....	42
4.1	Introduction.....	42
4.1.1	Text mining.....	42
4.1.2	Extraction of biomedical concepts	44
5.	The experimental environment.....	48
5.1	Introduction.....	48
5.2	Protein attribute/feature datasets.....	49
5.2.1	Sequence derived attribute datasets	49
5.2.2	Biomedical concept attribute datasets	52
5.2.3	Merged attribute datasets	55
5.3	Feature selection, classification and evaluation	56
5.4	Framework for supervised machine learning.....	59
6.	Results	62
6.1	Introduction.....	62
6.2	Naïve Bayes	62
6.2.1	Sequence derived attribute datasets	62
6.2.2	Merged attribute datasets	64
6.3	Decision trees.....	69
6.3.1	Sequence derived attribute datasets	69
6.3.2	Merged attribute datasets	70
6.4	Support Vector Machines.....	73
6.4.1	Sequence derived attribute datasets	73
6.4.2	Merged attribute datasets	74
6.5	Results overview.....	76
7.	Discussion.....	78
7.1	Introduction.....	78
7.2	Discussion of the results.....	79
8.	Conclusion	82
	References.....	84
	Biography	90
	Statements	91

LIST OF FIGURES

Figure 2.1: 10-fold cross validation.	11
Figure 2.2: Types of attributes.	13
Figure 2.3: Filter (a) and Wrapper (b) feature selection methods.	14
Figure 2.4: Classifying a new instance with k-nearest neighbours for k=3 (left) and k=5 (right).	16
Figure 2.5: An example of a decision tree.	21
Figure 2.6: Sub-tree replacement and sub-tree raising (adopted from [51]).	23
Figure 2.7: An example of a simple hyperplane.	24
Figure 2.8: Another example of a hyperplane.	24
Figure 2.9: Transformation of a two-dimensional space.	25
Figure 2.10: Several different solutions for a classification problem where data cannot be completely separated with a simple solution.	27
Figure 2.11: A sample ROC curve.	31
Figure 3.1: Two amino acids connect into a simple polymer.	35
Figure 3.2: Peptide backbone.	36
Figure 3.3: Levels of protein structure [89].	37
Figure 3.4: Solubility index frequency for cytoplasmic proteins.	41
Figure 4.1: Document analysis steps.	43
Figure 4.2: An example of associated biomedical concepts for a search query “p53”.	46
Figure 4.3: An example of a document where tumour is associated with protein p53.	47
Figure 5.1: Classification scheme.	49
Figure 5.2: A part of an XML file returned by the FACTA web service.	53
Figure 5.3: An example of determining biomedical concept (BC) attributes for two proteins.	54
Figure 5.4: 10-fold cross validation with feature selection.	57
Figure 5.5: Visualisation options in Weka.	59
Figure 5.6: A CSV input for Weka.	60
Figure 5.7: An ARRF input for Weka.	60
Figure 6.1: Confusion matrixes for the best three classifiers on sequence derived attribute datasets with Naive Bayes.	63
Figure 6.2: MCC values for five merged datasets and different number of attributes selected.	65

LIST OF TABLES

Table 2.1: An example of machine learning input.	8
Table 2.2: A simple dataset.	18
Table 2.3: The data counts and probabilities.	18
Table 2.4: A confusion matrix for a binary classification problem.	28
Table 2.5: Critical values for the Wilcoxon-signed ranks test at $\alpha = 0.05$, $\alpha = 0.02$ and $\alpha = 0.01$. A classifier is significantly better than another if it performs better on at least w_α data sets.	32
Table 2.6: Comparison of accuracy measurements for two classifiers on 12 databases.	33
Table 3.1: Standard amino acid abbreviations.	36
Table 4.1: An example of document presentation.	44
Table 5.1: Amino acid alphabets used to compute frequencies for monomers, dimmers and trimmers.	50
Table 5.2: The Sequence representation for different alphabets.	51
Table 5.3: Attributes computed from protein sequence.	51
Table 5.4: Sequence-derived attribute datasets used in our experiment.	52
Table 5.5: Extracted biomedical concept attribute datasets for the eSol database.	53
Table 5.6: Number of attributes in merged attribute datasets.	56
Table 6.1: Results for Naïve Bayes built on sequence derived attribute datasets.	63
Table 6.2: Merged attribute datasets used in one performance measurement.	64
Table 6.3: Performance measurements for $NBm_{disease}$ and NBm_{enzyme} with Information Gain.	66
Table 6.4: Performance measurements for NBm_{drug} , $NBm_{compound}$ and $NBm_{symptom}$ with Information Gain.	68
Table 6.5: Results for $J48s$	70
Table 6.6: Performance measurements for $J48m_{drug}$ and $J48m_{enzyme}$ with Information Gain.	71
Table 6.7: Performance measurements for $J48m_{disease}$, $J48m_{compound}$ and $J48m_{symptom}$ with Information Gain.	72
Table 6.8: Results for $SMOs$	73
Table 6.9: Performance measurements for $SMOm_{drug}$ and $SMOm_{symptom}$ with Information Gain.	74
Table 6.10: Performance measurements for $SMOm_{disease}$, $SMOm_{compound}$ and $SMOm_{enzyme}$ with Information gain.	75
Table 6.11: Overview of the statistical significance of the results.	76
Table 6.12: Results for two best classifiers built on SdAd_eSol and MAd_eSol_Symptom.	76

LIST OF ALGORITHMS

Algorithm 5.1: Extraction of biomedical concepts for eSol.....	54
Algorithm 5.2: The general classification scheme for building and evaluating classifiers.	58
Algorithm 6.1: The algorithm for classifying the eSol proteins in biomedical concept space.	77

1

Introduction

1.1 Motivation

Proteins are essential chemical compounds responsible for many functions in organisms. For instance, some proteins have structural roles while others act as enzymes which catalyse important biological reactions. It has been proven that the function of a protein is determined by the protein's structure [1]. Understanding the latter is therefore an important research area. To achieve this, researchers often need to produce and isolate proteins from complex mixtures [2]. An important protein's property that influences this isolation process is its *solubility*. It has been shown that only soluble proteins can be effectively isolated while insoluble proteins form the so called *inclusion bodies*. The latter are harder to isolate and researchers use different techniques to solubilise them [56]. For instance, some common strategies for increasing protein solubility are changing the temperature of the production process [11], adding different additives in the process [12] and the use of protein engineering methods [15, 18]. In addition to difficulties when isolating proteins, low protein solubility can lead to several diseases [3].

For these reasons, identifying soluble proteins is an important challenge in biophysical studies and is often achieved by performing experiments in laboratories. However, this is usually a trial and error process with low success rate [4, 5]. Therefore, researchers have proposed the use of machine learning algorithms [6] as an alternative way to identify soluble proteins. These methods, which are the focus of this thesis, take a set of proteins as their input and output a *classifier*, which classifies proteins into soluble and insoluble [29-34]. Researchers can then use this classifier for predicting the solubility of new proteins. This allows them to identify proteins that are likely to be problematic from the solubility standpoint.

In recent years, researchers have used different protein databases on which they applied several machine learning techniques for building a successful classifier for protein solubility. The first attempt was a simple statistical method that was introduced by Wilkinson and Harrison in 1991 [31] and improved in 1999 [33]. Both methods performed a statistical analysis of the composition of several proteins that do and do not form inclusion bodies, and reported a success rate (i.e., ratio of correctly classified proteins) of 88%. However, this success rate is a result of the analysis of a very low number of proteins and the solution cannot be generalized for larger data sets. Later, in 2000, Christendat et al. [13] reported the use of several machine learning algorithms, such as decision trees, Bayesian classifiers, and support vector machines, to classify soluble proteins. The study concluded that these algorithms perform equally well with a success rate around 66%. The similar success rate was later also reported by other authors such as Bertone et al. [14] and in 2004 and Goh et al. [32], who used decision trees. In 2006 Idicula-Thomas et al. [29] used support vector machines to predict protein solubility and reported a success rate of 72%. The method was improved in 2007, when the Naive Bayes algorithm was added as a secondary classifier [30]. The authors reported a success rate of 72% and introduced several other metrics for measuring performances of classification algorithms, where their method outperformed the previous ones. In 2009, Magnan et al. [34] designed a new algorithm which was based on support vector machines and resulted in a success rate of 74%. Also in 2009, Niwa et al. [73] reported the use of support vector machines for classifying entire ensemble of proteins from a bacterium and reported a success rate of 80%.

A closer look into the reviewed studies has allowed us to identify the following three factors that influence the success rate of a protein solubility classification method: (a) the machine learning algorithm for building the classifier, (b) the proteins that are used to build and evaluate the classifier and (c) the attributes that describe these proteins. Since the authors have used several different machine learning methods on different protein databases with different attributes, not all of the reported results can be directly compared. However, we have concluded that the support vector machines algorithm resulted in the most successful protein solubility classification [29, 30, 34, and 73]. Support Vector Machines have been proven to offer the best results several times before, also in other fields of bioinformatics, such as the gene expression classification [17, 36]. In addition, we have concluded that the dataset used in [34] contains proteins from the previous works, while the dataset used in [73] contains the most accurate information about solubility. And finally, we have concluded that all reviewed works used attributes that are mostly calculated directly from the protein's basic structure [55] and are, therefore, limited.

Improving the success rate of the reviewed classifiers is an important challenge, since the number of proteins used in protein solubility classification can be tens of thousands and a success rate of 70-80% results in many misclassified proteins. It is therefore no surprise that authors, such as [73], indicate that more work has to be done to improve the success rate.

1.2 Goals and contributions

The main goal of this thesis is to improve the success rate of protein solubility classification. To achieve this goal, we expand the attribute space with the introduction of new attributes that are not limited to the protein's structure.

We believe that very significant amount of useful information about proteins is held in the scientific literature. If we can extract and present this information in terms of protein attributes, we will be able to use it for the classification purposes. Therefore, the new attributes introduced in this thesis are defined by a novel method that uses text mining techniques to extract medical knowledge in the form of words that describe diseases, drugs, symptoms, enzymes and chemical compounds, from articles published in the MEDLINE [26] system. We call these attributes *biomedical concept attributes*. The most relevant biomedical concept attributes are selected and ranked according to how frequently they appear in the articles. The classifiers obtained using standard attributes for protein solubility classification are compared to those obtained also using the new attributes. Three different classification algorithms are used to test the new attributes and we compare the algorithms using several different metrics all of which have been in the literature to compare protein solubility classification algorithms. The most common performance measure is the success rate (also called the accuracy) which is calculated as the number of correctly classified instances divided by the total number of instances. Since biological databases usually contain unbalanced data, the accuracy metric does not provide enough information about the performance of a classifier. Therefore, other metrics such as the Matthews Correlation Coefficient (MCC) [35] are also used.

1.2.1 Hypothesis and research methodology

Based on the goals we want to achieve, we define the main hypothesis of our thesis as follows:

A new, more successful classifier for protein solubility classification can be built using biomedical concept attributes, which do not depend only on the primary structure of a protein.

We examine the hypothesis by extending it into two sub-hypotheses:

Hypothesis 1:

By merging the most relevant sequence derived attributes and the most relevant biomedical concept attributes, we improve protein solubility classification for some methods.

Hypothesis 2:

Protein attributes derived from the protein's primary structure do not carry all information needed for optimal protein solubility classification.

In this thesis, we try to confirm the hypothesis by selecting an appropriate protein database, implementing different classification algorithms that build classifiers using biomedical concept attributes and comparing them with the classifiers obtained only using attributes derived from the protein's structure. In particular, we use the following three common classification algorithms: Naïve Bayes, Decision trees and Support Vector Machines. For each method we follow 3 main steps as follows.

In the first step, we build 20 classifiers using 20 different sequence derived attribute datasets that have been shown to influence the protein solubility classification process [34]. We measure the classifiers' performance with several different metrics and we identify the attribute dataset A_{best} that resulted in the classifier with the best performance. In the second step, we extend the attribute space by defining and selecting the most relevant biomedical concept attributes. We add these new attributes to each of the 20 sequence derived attribute datasets. Again, we build 20 classifiers, this time using 20 different merged attribute datasets. In the third step, we first compare the performance scores measurements for the classifier obtained by A_{best} with those for the classifier obtained by A_{best} with added biomedical concept attributes. When the latter results in higher scores, it can be concluded that a better classifier was obtained with added biomedical concept attributes. In addition, to test statistical significance of the results, we also compare the performances for other pairs of classifiers (C_{s_i} , C_{m_i}), where C_{s_i} is a classifier obtained by the i -th sequence derived attribute dataset and C_{m_i} is a classifier obtained by the i -th sequence derived attribute dataset with added biomedical concept attributes. The Wilcoxon signed-ranks test [59] is performed to measure the statistical significance of these results. We repeat the second and the third steps for the following five different biomedical concept groups: diseases, drugs, symptoms, enzymes and chemical compounds.

1.2.2 Expected scientific contributions

The expected main scientific contributions of this thesis include:

- an analysis of protein databases for protein solubility (Section 3),

- a description of a novel approach for using biomedical concepts, retrieved by text mining techniques, in the classification process. An analysis of text mining techniques that can be used to assist this process is performed (Section 4),
- a method for converting biomedical concepts retrieved by text mining techniques, into biomedical concepts attributes for proteins, which offer new knowledge to the classifier and improve its performance (Section 4 and Section 5),
- an original comparison between existing algorithms for predicting protein solubility with the new developed algorithm, which will classify the proteins with the help of different groups of biomedical concepts. The comparison is done for the following three different methods: Naïve Bayes, Decision Trees and Support Vector Machines (Section 6, Section 7),
- a protein solubility classification algorithm for building a classifier with the best performance (Section 6),
- new medical knowledge extracted from the classification results (Section 6).

1.3 Thesis structure

This thesis is divided into eight sections, each containing several sub-sections. Let us describe each section in more detail.

Section 2 introduces the required theoretical background on machine learning algorithms, focusing on supervised classification techniques. In particular, four classification techniques are described, each of them representing a distinctive way of building classifiers. Advantages and disadvantages of each technique are presented. In addition, several methods for reducing the number of attributes in the classification process are illustrated.

The first part of Section 3 introduces the required background on proteins. Four levels of protein structure are described and the connections between them are given. In addition, the section provides different definitions of protein solubility. The second part of Section 3 provides our analysis of two datasets that have been previously used for classifying soluble proteins. The section concludes with a detailed description of the database used in the experiments in this thesis.

Section 4 discusses text mining techniques and their connection to the biomedical concept attributes used in this thesis. In the first part of the section, the basics steps of extracting useful patterns from unstructured text are introduced, while the second part of the section provides our analysis of a tool which mines biomedical concepts from literature.

Section 5 depicts the environment for our empirical study. A detailed description of the classification scheme for building several different classifiers and their evaluation is

presented. In addition, the section also discusses the Weka framework, which is used for implementing the classification scheme.

Section 6 presents the results of each classification algorithm used. It compares the classifiers built with only sequence derived attributes with classifiers built also with biomedical concept attributes. At the end of this section, the algorithm with the best performance for protein solubility classification is constructed.

Section 7 interprets the results and discusses their connection to the hypothesis of this thesis. In addition, it also identifies our original scientific contributions.

Section 8 concludes this thesis with a short summary and discusses several open problems and future research directions.

2

Machine learning

2.1 Introduction

This section presents the basics of machine learning, a specialized field of artificial intelligence which develops computer algorithms that make it possible for computers to learn new knowledge from the study of different datasets.

Machine learning has applications in many different fields such as natural language processing (NLP), search engines, medical diagnosis, bioinformatics, financial analyses and predictions, voice and speech recognition and intelligence in computer games. In the last two decades several definitions of machine learning have appeared. The following are some of the most influential definitions which also provide valuable insights into this specialized field:

- *Machine learning is a science of the artificial. The field's main objects of study are artefacts, specifically algorithms that improve their performance with experience [37].*
- *The field of machine learning is concerned with the question of how to construct computer programs that automatically improve with experience [38].*
- *Machine learning is programming computers to optimize a performance criterion using example data or past experience [6].*
- *Machine learning is an area of artificial intelligence concerned with the study of computer algorithms that improve automatically through experience. In practice, this involves creating programs that optimize a performance criterion through the analysis of data [40].*

When reading the above definitions, one can identify the common concepts of machine learning: computer algorithms and automatic improvement through data and past experience.

The following sections describe the basic terminology in machine learning and take a closer look at the meaning of these concepts.

2.2 Basic terminology

2.2.1 Input of machine learning algorithms

Every machine learning algorithm takes a set of *instances* (also called examples, samples or objects) as the input. In addition, each instance is characterized by a set of predetermined *attributes* (also called features) and their values. Therefore, each input dataset is represented as a matrix of instances versus attributes. Table 2.1 shows an example of such an input, where rows represent instances and columns represent attributes. Note that the example is used only for illustrating purposes, as the data in real machine learning problems typically consists of hundreds or even thousands of instances and attributes. The table has twelve instances, each characterized by six attributes, Weather temperature, Weather outlook, Weather humidity, Weather wind, Health and Go Out. As illustrated by the table, attributes can be of different types, such as numbers or strings.

Table 2.1: An example of machine learning input.

Weather temperature	Weather outlook	Weather humidity	Weather wind	Health	Go Out
15	sunny	87	no	cold	no
12	sunny	92	strong	healthy	no
25	overcast	88	no	healthy	yes
20	rainy	98	no	healthy	yes
19	rainy	72	strong	fever	no
21	overcast	67	mild	healthy	yes
22	sunny	97	strong	healthy	no
24	sunny	72	no	healthy	yes
12	rainy	82	no	healthy	yes
22	overcast	92	mild	healthy	yes
28	overcast	77	no	healthy	yes
13	rainy	93	strong	cold	no

Input datasets have to be well formed, i.e. have to be in the format required by the machine learning algorithm that is used to classify the datasets. For instance, some algorithms are able to handle only continue attribute values, while others can also process discrete ones. Preparing the data is one of the most important steps in the machine learning process and it includes several sub-steps, such as gathering the data, defining the correct type of data (e.g. integer or string), correctly handling missing values and identifying inaccurate values. Note

that while this step (preparing the data) is not the focus of this thesis, we have to follow certain rules, as described in Section 5.

2.2.2 Output of machine learning algorithms

Most machine learning algorithms look for structural patterns in the input data and output descriptions of these patterns [51]. How the output is presented, i.e., which *knowledge representation* is used, depends on the type of machine learning algorithm used. For instance, a simple form of the knowledge representation is a *decision list* which consists of logical *if-then rules*. Let us illustrate this form of representation by means of an example: imagine that the instances in Table 2.1 represent a decision about sending a child out to play in different weather and health conditions. The weather and health attributes represent the conditions, while the last attribute represents the decision made under these conditions. The following rules can be extracted:

IF Weather wind = no AND Health = healthy then Go out = yes

IF Weather outlook = sunny and Weather temperature > 22 Go out = yes

These rules represent structural patterns in the input data and can be used in the future to take new decisions. This can be seen as *classifying* new instances according to some missing *class attribute*. Consider, for example, new instances of Table 2.1 where the class attribute **Go out** is missing. We can use the above decision list to classify these new instances into two classes: yes and no, corresponding to the cases in which the child will be allowed to play outside or not, respectively. The attributes that are not class attributes are called the *descriptive* attributes.

Decision lists are only one possible form of knowledge representation. In this thesis, we will use different a different form called *decision tree* introduced in Section 2.4.3. Note, however, that often the knowledge representation resulting from a machine learning algorithm provides no insight into the decision making of the algorithm and only a mapping function that classifies instances is given. An example of this kind of machine learning algorithm is a *Support Vector Machine* (SVM), introduced in Section 2.4.4.

2.2.3 Types of machine learning algorithms

Machine learning techniques can be divided into five groups [51, 69]:

- *Supervised learning* algorithms operate, in a sense, under supervision since they are provided with the correct outcome and the class attribute of each input instance is known (we also say that the data is *labeled*). This type of algorithm is often referred to as a *classification or regression algorithm* and its output is a function called

classifier (for discrete output values) or a *regression function* (for continuous output values).

- In *unsupervised learning* algorithms, the input instances have no class associated to them. This type of algorithm is referred to as a *clustering algorithm*.
- *Semi-supervised learning* is a mixture of the above two techniques. While the goal is to classify new instances, the input data contains both unlabeled and labeled data.
- In *reinforcement learning* the algorithm learns how to react to actions in the environment and tries to maximize some notion of cumulative reward. The input data is never labeled, nor sub-optimal actions explicitly corrected.
- *Transduction* is similar to supervised learning but it uses both labeled and unlabeled instances from the test set (which are unlabeled) during the learning phase. As a result, transduction algorithms offer better results compared to supervised algorithms. However, they do not build any predictive model which could be used for new, previously unknown instances and the whole learning process has to be repeated if a new unlabeled instance is added.

This thesis focuses on supervised machine learning techniques since their characteristics make them the most suitable for protein solubility classification of the database described in Section 3.3.

2.2.4 Training and test set.

It is essential to measure the performance of a supervised machine learning algorithm when it extracts the patterns from the data. For this purpose it is common to partition the input dataset into two groups: the training set and the test set. The former group is used for the extraction of patterns (the algorithm is “trained” to do this) while the latter group serves for testing the effectiveness of the extracted patterns. While the partition depends on the specific problem, it is usually made randomly with most of the data being used for training, and only a smaller portion of the data being used for testing. A common technique for partitioning the data, also used in this thesis, is the *N-fold cross validation*.

Cross validation is a common technique to evaluate built classifiers when training and test data are limited. For N-fold cross validation the technique randomly splits the data into N subsets where one subset is used for testing and N-1 subsets are used for training. When partitioning the data, the class in the original dataset is represented in approximately the same proportion in each subset as in the full dataset. The whole process is repeated N times to avoid the so called *selection bias* [22]. It has been shown that N=10 results in the best estimate of the classifier’s performance and, therefore, 10-fold cross-validation has become the standard method used in practice (Figure 2.1) [51].

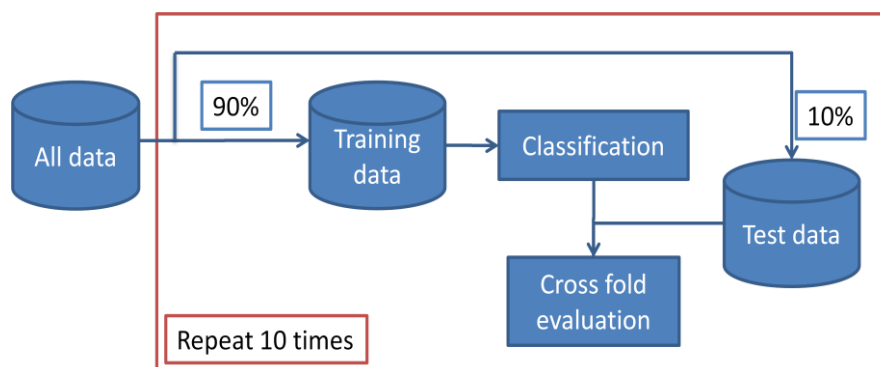


Figure 2.1: 10-fold cross validation.

2.3 Supervised machine learning

As stated before, supervised machine learning algorithms learn from training instances for which the correct class attribute values are known. Therefore, they learn on the basis of previous knowledge and cannot learn by themselves. While different definitions of supervised learning emerged [41, 42, 52], the following is one of the clearest:

Supervised learning is a machine learning technique whereby the algorithm is first presented with training data which consists of examples which include both the inputs and the desired outputs; thus enabling it to learn a function. The learner should then be able to generalize from the presented data to unseen examples [40].

The training data in supervised machine learning consists of pairs (D, C) of training instances where, for each instance, D provides the names and values for the descriptive attributes, i.e., attribute dataset, and C provides the output values and the name for the class attributes. The output of the algorithm is a function that maps new instances of D to a linear value (*regression algorithm*) or to a discrete value (*classification algorithm*) for C . The goal of the learning algorithm is to define the mapping function in such a way that its output values obey the rules defined by the learning instances. The rest of this sub-section mainly focuses on classification algorithms since they are used in our experiments.

To illustrate problem solving with supervised machine learning, let's take a look at an example of character recognition, where the algorithm's task is to recognise scanned images of handwritten, typewritten or printed letters. This is a classification problem (outputs are discrete values since there is a discrete, finite number of characters) and it is solved in the following five steps:

- First, we decide what the learning instances are. In the example of character recognition the single instance is a character.
- Second, the learning data has to be gathered through measurements and it has to describe the reality of the problem well. Although there is not general definition of what enough data is, today's datasets usually contain at least a few hundred instances. In our example, we would collect as many handwritten, typewritten or printed different characters as possible.
- Third, we have to describe each object from the collected data in terms of a set of attributes. This step is very important, particularly when it is not clear in advance what attributes should be used [63]. Furthermore, the number of attributes selected is also important since it might affect the performance of the classifier. In our example, the attributes that describe each character could include the number of straight/curved lines, the angle between them and the size of the character.
- Fourth, we have to decide which classification method we want to use for building the classifier. This step can be particularly time consuming and difficult. The quality of the results strongly depends on the method selected and some problems are better suited by some of the methods. Unfortunately, choosing an optimal method is a complex task even for experts who are familiar with the field, since there are many factors that influence the effectiveness of the classifier. For our example, there are many different possibilities including, but not limited to, neural networks, decision trees and support vector machines. These will be described in Section 2.4.
- Fifth, the selected method is used to take the learning data as the input and use these data to output the classifier.
- Finally, the classifier's performance is measured with the test data. Note that the test data should not depend on the training data in any way. Several metrics are used for the performance measurements, as described in Section 2.6.

2.3.1 Attributes in supervised machine learning

As mentioned before, supervised machine learning uses two groups of attributes: descriptive and class attributes. Descriptive attributes are used for describing each instance as accurately as possible, while class attributes define the classes in which we want to divide the instances. This section describes these attributes in more detail.

2.3.1.1 Types of attributes

The value of an attribute can be *categorical* or *numeric* (Figure 2.2). The latter must be a number while the former can be any predefined label. Numeric attributes can be further divided into *continuous* (e.g. temperature, length) or *discrete* (e.g. number of children) while categorical attributes can be *nominal* (e.g. type of a colour) or *ordinal* (e.g. level of education), depending on whether they have an order relationship or not.

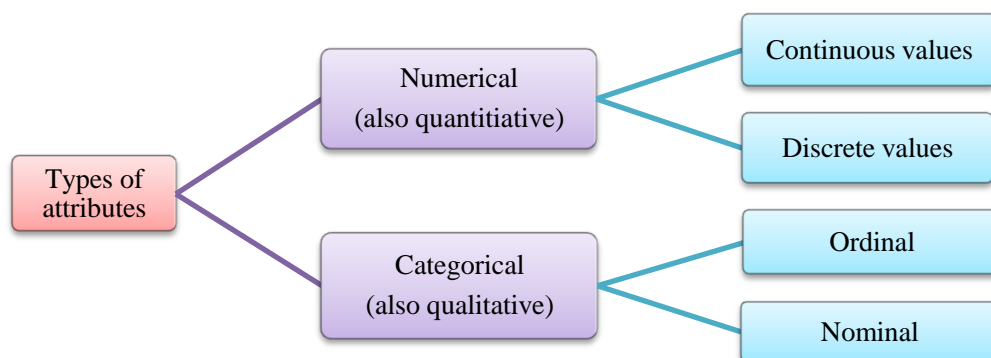


Figure 2.2: Types of attributes.

Note that not all classification algorithms can work with every type of attributes. Therefore, the type of the attributes also influences the classification algorithm that can be used during the learning process.

2.3.1.2 Attribute/feature selection

As we have seen in previous sections, a classification algorithm is typically presented with a set of training instances, their attribute values and class labels. The task of the algorithm is to train a classifier that will be useful in classifying new instances. Often, input training datasets contain irrelevant attributes which can degrade the performance of the algorithm (both in speed and predictive accuracy) for predicting the desired output [77]. It is common for performance to be improved when the number of attributes is reduced. Importantly, this has been also the case for the classification algorithms applied to protein solubility classification problems [29-31]. Unfortunately, it is not possible to rank attributes simply according to their individual properties and select only the best. This is because the attribute properties may depend strongly on each other and a subset of individually irrelevant attributes may prove to be rather significant due to their positive interaction effects. Therefore, the optimal solution would be to test all attribute subset combinations. However, this is impractical since it requires enormous amounts of computational time. Therefore, different non-optimal methods have been proposed to find a subset of the original attributes that generate a classifier with better performance. In general, two types of such methods are used: *feature selection* and *feature extraction*.

Feature selection methods select a subset of relevant attributes from the original set of attributes. This is usually done in two steps: ranking and removing of the attributes. A feature

selection method ranks the attributes in such a way that the more relevant attributes are ranked higher while the less relevant attributes are ranked lower, and usually removes attributes that are not ranked among the top n attributes where n is a user defined parameter.

The ranking of features can be performed in two main ways. *Univariate* methods evaluate attributes one by one without considering any dependencies between attributes. *Multivariate* methods consider information from multiple attributes simultaneously. Attribute selection in multivariate methods can furthermore be divided in two main ways [77]. *Filter* methods select relevant attributes before the classification step with no knowledge about the classifier. *Wrapper* methods include (wrap) the learning algorithm in the selection process. While filter methods make an independent assessment based on general characteristics of the data, wrapper methods evaluate the subset of attributes using the machine learning algorithm that will ultimately be used for learning. Figure 2.3 illustrates both concepts. Since wrapper methods already use the machine learning algorithm in the selection step, they tend to be more computationally intensive than filter methods.

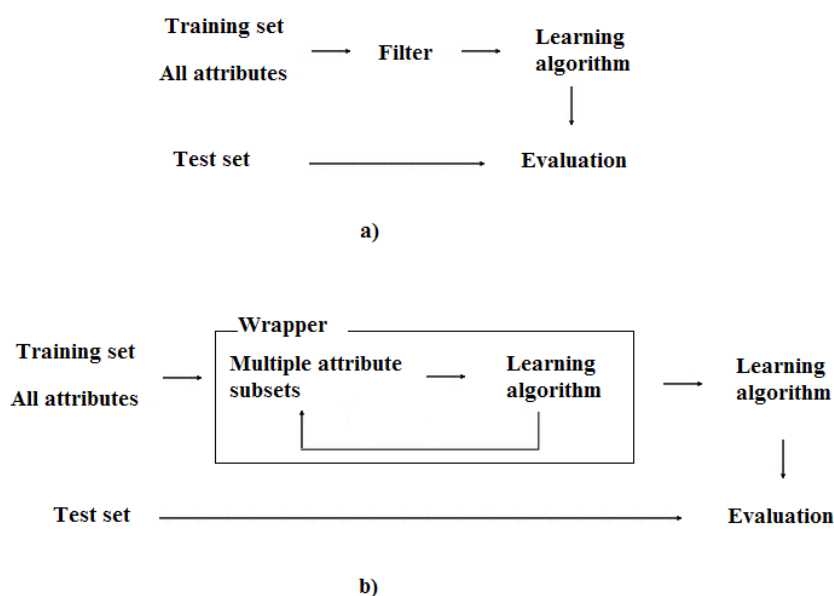


Figure 2.3: Filter (a) and Wrapper (b) feature selection methods.

In this thesis we used the fastest univariate and fastest multivariate methods currently available: *Information Gain* [38] and *ReliefF* [51]. *Information Gain* ranks each attribute based on the decrease in entropy (uncertainty associated with a random attribute) achieved when the attribute is present in the dataset [43]. Attributes with maximum *Information Gain* (and minimum entropy) are then selected. *ReliefF* is a filter multivariate method that ranks

each attribute based on its ability to distinguish between instances that are near to each other. The basic idea is to draw instances at random, compute their nearest neighbours, and adjust a feature weighting vector to give more weight to features that discriminate the instance from neighbours of different classes.

As mentioned before, the other possibility to reduce the number of attributes is to use feature extraction methods, which transform (rather than remove) existing attributes into new ones. Although these methods are not used in this thesis (since we assume that some of our attributes contain noise and have to be removed), we briefly describe the most common representative of these methods the Principal Component Analysis (PCA) [78], which has also been used in protein solubility classification [29]. PCA performs a linear transformation of the initial feature space to a lower dimensional space aiming to achieve the maximal variance of the data in the low-dimensional representation. To transform the feature space, an attribute correlation matrix (i.e., a matrix where mutual relationships of attributes are presented) of the data is constructed and the eigenvectors of this matrix are computed. The eigenvectors with the largest eigenvalues (the principal components) can now be used to reconstruct a large fraction of the variance of the original data. In this way the most important variance information is retained in a lower dimensional feature space.

2.4 Methods for building classifiers

This section describes four classification techniques all of which are unique in their own way and which together represent the four main different groups of supervised machine learning techniques. Furthermore, they are the basis for understanding the classification scheme introduced in later sections.

2.4.1 K-nearest neighbours

The K-nearest neighbours method is a typical representative of instance based methods where each new instance is compared with the existing ones using a distance metric, and the closest existing k instances are used to assign the class label to the new one. This is achieved via a majority vote where the class label of each k neighbour is used as vote.

Several metrics can be used to compute the distance between the object and its neighbours. The most common of these is the *Euclidean distance* metrics [79] which works as follows. Let's say that an instance x is described with m attributes and that we define a function $f_i(x)$ which outputs the value of attribute i where $i = 1, 2, \dots, m$. Then, the Euclidean distance d between two objects x_1 and x_2 is defined as:

$$d(x_1, x_2) = \sqrt{\sum_{i=1}^m (f_i(x_1) - f_i(x_2))^2} \quad (2.1)$$

The k-nearest neighbours method has been empirically shown to work best on datasets with continuous and normalized attributes. Common k values for binary classification problems are one, three and five (even numbers are not recommended since they can lead to a tie, i.e., the same number of votes [38]). The selection of the k parameter can significantly affect the result as it can be seen on Figure 2.4, where the new instance (denoted by symbol ‘?’) is assigned to the negative (positive) class when the votes of three (five) neighbours are considered. Generally, higher k values result in more accurate, but also more computationally intensive algorithms.



Figure 2.4: Classifying a new instance with k-nearest neighbours for k=3 (left) and k=5 (right).

The k-nearest neighbour method has several disadvantages:

- The results regarding the classification process are not easily interpreted since it provides no information regarding how decisions are made.
- The classification process usually needs significant amounts of processing power since all calculations are made during the classification.
- All instances have to be saved in main memory and stay there until the classification process ends. This is a big problem for large datasets, such as those usually used in bioinformatics.
- The accuracy of the algorithm can be severely degraded if the data contains noisy or irrelevant features.

2.4.2 Naïve Bayes

Naïve Bayes is a probabilistic classifier whose aim is to calculate conditional probabilities of the instances for each class with the minimal possible error. It is based on Bayes' theorem, which computes the probability of the occurrence of a hypothesis H being true given some experimental data instance D . This conditional probability is called the *posterior probability*

and it is equal to the probability that D would be produced given H is true (i.e. *likelihood*), multiplied by the probability of H being true before D are seen, divided by the *prior* probability of seeing that particular D . We write Bayes' theorem with the following equation:

$$P(H/D) = \frac{P(D/H)P(H)}{P(D)} \quad (2.2)$$

As it can be seen from the equation, $P(H/D)$ will be large if:

1. $P(H)$ is large, i.e. the hypothesis was likely regardless of the particular instance.
2. $P(D)$ is small, i.e. the instance is very rare.
3. $P(D/H)$ is large, i.e. the hypothesis predicts the instance with high probability.

We can apply Bayes' theorem to the classification process by making the posterior probability to be the probability of assigning a class label to an instance given its attributes. Therefore, H represents a particular class label C , while D is set of values A_1, \dots, A_n of the descriptive attributes of a given instance. Considering this, we can rewrite Equation 2.2 in the following way:

$$P(C/A_1, \dots, A_n) = \frac{P(A_1, \dots, A_n/C)P(C)}{P(A_1, \dots, A_n)} \quad (2.3)$$

In practice we are only concerned with computing the values for the numerator of this equation, since the denominator does not depend on the class C and the values of the attributes A_i are given, making the denominator effectively constant. In addition, Naïve Bayes assumes that the attribute values are conditionally independent, given the classification of the instance. As a result of this assumption, the likelihood can be calculated as:

$$P(A_1, \dots, A_n/C) = P(A_1/C)P(A_2/C) \dots P(A_n/C) = \prod_{i=1}^n P(A_i/C) \quad (2.4)$$

Considering Equation 2.2 and Equation 2.3, we can now write:

$$P(C/A_1, \dots, A_n) = \frac{1}{Z} P(C) \sum_{i=1}^n P(A_i/C) \quad (2.5)$$

where Z is the denominator scaling constant and it is used to normalize the results.

To illustrate Naïve Bayes in practice, let us consider the following simple example, where we have a binary classification problem that classifies instances into the *yes* or *no* classes. Table 2.2 contains the learning data (rows 2-6) and a new instance to be classified (the last row). The data attributes can have values 0 or 1. Table 2.3 contains a summary of the data obtained by counting how many times each attribute–value pair occurs within each class value (*yes* and *no*). For instance, it can be seen from Table 2.2 that *Att1* has value 0 for three examples, one of which have *Class* = *yes* and two of which have *Class* = *no*. The cells in the first part of Table 2.3 simply count these occurrences for all possible values of each attribute. In the second part of Table 2.3 this information is rewritten in the form of observed probabilities. For example, of the three class attributes with value *yes*, *Att1* has value 0 in one example, yielding a probability of 1/3. The values in bold are the observed probabilities for the attribute values of the new instance and are used for the calculations appearing in the rest of this section.

Table 2.2: A simple dataset.

Att1	Att2	Att3	Class
1	1	0	yes
0	0	0	yes
1	1	1	no
0	0	1	no
0	1	1	yes
1	0	1	?

Table 2.3: The data counts and probabilities.

	Att1		Att2		Att3	
	yes	no	yes	no	yes	no
Count 0	2	1	1	1	1	0
Count 1	1	1	2	1	2	2
Probability 0	2/3	1/2	1/3	1/2	1/3	0/2
Probability 1	1/3	1/2	2/3	1/2	2/3	1/2

Let us now classify the new instance according to the Bayes rule. The instance has the attribute set $AttSet = \{(Att1, 1), (Att2, 0), (Att3, 1)\}$ and can be assigned either *yes* or *no* to class label C . To classify it correctly, we have to calculate the following two conditional

probabilities: $P(\text{yes}/\text{AttSet})$ and $P(\text{no}/\text{AttSet})$, i.e., the probability that the new instance will be classified as *yes* and *no*, respectively, given its attribute set.

Let us start by calculating $P(\text{yes}/\text{AttSet})$. According to Equation 2.3, this conditional probability is a product of $P(\text{AttSet}/\text{yes})$, i.e., the probability of seeing the attribute set given the class label *yes*, and $P(\text{yes})$, i.e. the probability of an instance being classified as *yes* without knowing anything about its attribute set. According to Equation 2.4 and the bolded probabilities in Table 2.3 we can calculate:

$$P(\text{AttSet}/\text{yes}) = \sum_{i=1}^3 P(\text{Att}_i/\text{yes}) = P(1/\text{yes}) * P(0/\text{yes}) * P(1/\text{yes}) = \frac{1}{3} * \frac{1}{3} * \frac{2}{3} = \frac{2}{27} \quad (2.6)$$

Since three out of five learning instances belong to class *yes*, we can also calculate:

$$P(\text{yes}) = \frac{3}{5} \quad (2.7)$$

Considering Equation 2.5, we can then write:

$$P(\text{yes}/\text{AttSet}) = \frac{1}{Z} * \frac{3}{5} * \frac{2}{27} = \frac{1}{Z} * 0.044 \quad (2.9)$$

In a similar way, we can calculate the conditional probability of the new instance being assigned the *no* class label given its attribute set:

$$P(\text{no}/\text{AttSet}) = \frac{1}{Z} * \frac{2}{5} * \frac{1}{2} * \frac{1}{2} * \frac{1}{2} = \frac{1}{Z} * 0.050 \quad (2.10)$$

Since the sums of both probabilities should be 1, we use the Z constant to normalize the values so that Z is the sum of $P(\text{yes}/\text{AttSet})$ and $P(\text{no}/\text{AttSet})$ which gives us the final results:

$$P(\text{yes}/\text{AttSet}) = \frac{1}{0.050 + 0.044} * 0.050 = 0.53 \quad (2.11)$$

and

$$P(\text{no}/\text{AttSet}) = \frac{1}{0.050 + 0.044} * 0.044 = 0.47 \quad (2.12)$$

According to these results, the new instance is successfully classified as *yes*. Note that, in the described form, the Naïve Bayes classification can go badly awry. Specifically, if a particular attribute value does not occur in the training set in conjunction with every class value, the calculated posterior probability will always be zero. Consider, for example a new instance whose *Att3* has value zero. Then, its $P(Att3/no)$ is also zero and, since the other probabilities are multiplied by this, the final probability of *no* would be zero no matter how large the other probabilities were. This can be fixed by using the *Laplace estimator* [51] in the following way:

$$P(0/no) = \frac{0 + 1}{2 + 2} \quad (2.13)$$

As it can be noted, the estimator increased the numerator by one. Since there are two instances with outcome *no* in the learning data, this change has to be compensated with increasing the denominator by two. This technique ensures that no attribute value that occurs zero times receives a zero probability. Note that the numbers used to increase the numerator and the denominator have to be added also to other conditional probabilities for the same attribute with the same hypothesis (i.e., $P(1/no)$ in our example).

The main advantage of Naïve Bayes is that it offers simple, fast and highly scalable model building and scoring. It can also be used for both binary and multiclass classification problems. Its main disadvantage is that it assumes that attributes are independent which is usually not true with real data. However, in practice it many times performs better than more complicated models, especially on small amounts of data, and it is therefore, recommended to build Naïve Bayes classifiers before more sophisticated methods are applied [51].

2.4.3 Decision trees

Decision trees are very popular due to their ability to represent the results in a decision tree format which is easy to interpret for experts, as they can see the decision making in the classifying process. The basic idea is to construct a tree whose leaves are labeled with a particular value for the class attribute and whose inner nodes represent descriptive attributes. Given an inner node *N*, the children of *N* correspond to different possible values of the associated descriptive attribute. Once a decision tree is built, determining the class value for a new instance is achieved by following a path from the root to a leaf according to the values of the descriptive attributes of the instance. The class value assigned will be that labelling the leaf.

In general, decision trees seek the optimal solution of the problem by means of a recursive separation of the problem space. This can be described as a *divide-and-conquer* approach consisting of the following steps:

1. Select the root attribute, i.e. the attribute that divides the problem space in the most effective way.
2. Partition the problem space based on the values of the root attribute.
3. Repeat the first and the second steps for each subset of attributes until the criterion for stopping the tree building process is achieved, i.e. until:
 - All the instances in a branch node belong to the same class. If so, the method simply creates a leaf node for the decision tree that selects that class.
 - The algorithm runs out of attributes to choose from. If so, a leaf is created with the class attribute of the majority of the instances in the leaf.
 - The number of instances in a branch node is lower than the minimum allowed number of instances for a leaf. Again, a leaf is created with the class attribute of the majority of the instances in the leaf.

The described method was first introduced by Quinlan in 1986 and it is still in use today [43]. Figure 2.5 shows a simple decision tree for classifying two types of tissue (Tumour and Normal) depending on the occurrence in the tissue of different genes. When interpreting the decision tree for a given instance, we start at the root node (i.e. 32598_at) and continue our way down the tree according to values of the attributes of our instance. For example, if our instance has the 32598_at attribute with a value less or equal to 29 and the value of the 38028_at attribute is greater than 14, this instance will be classified as Normal, since the path followed by the instance is the one marked with the blue arrows in Figure 2.5.

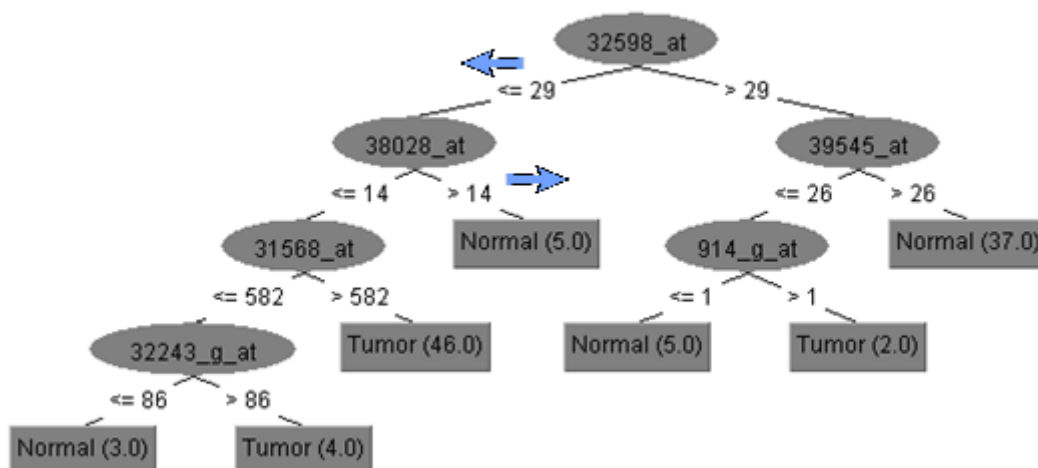


Figure 2.5: An example of a decision tree.

Quinlan also proposed the C4.5 algorithm [44] which has become the most common method used for building decision trees and is the one used in this thesis. This algorithm uses the concept of information entropy introduced in Section 2.3.1.2. In addition, it also uses the *gain ratio* concept which avoids choosing a node that would split the training examples into many small subsets and it makes sure that attributes that have only unique values (e.g. id number) or only a few different values (e.g. date) are not at the top of the tree. Both concepts are used to determine which node to split next in the algorithm by selecting the attribute with maximum ratio between its gain ratio and its entropy.

Once the tree is finished, the C4.5 algorithm goes back through the tree and attempts to remove branches that do not help. This phase is called *pruning* of the tree [44] and it is used to avoid *over-fitting* (see Section 2.5.1 for a detailed description of this concept) the tree to the instances from the learning data. In this way, the classification process is improved and the tree is usually smaller and easier to understand. Two methods are used to prune an already built tree: *sub-tree raising* (which is also used in this thesis) and *sub-tree replacement*. In the sub-tree replacement method, some sub-trees are replaced with single leaves using an algorithm that proceeds from the leaves and works up towards the root. Sub-tree raising is a more complex operation where an entire sub-tree is lifted closer to the root to replace the sub-tree's parent node. If the node has any other branches, the instances from those branches have to be reclassified into the sub-tree.

Both pruning methods are illustrated on Figure 2.6 (adopted from [51]). On the left-hand side of the figure, node B is replaced with one of its leaves while on the right-hand side node B is replaced with one of its sub-trees. Note that in the latter, instances represented with 4 and 5 are reclassified into the sub-tree (the node C with its branches). Therefore, we mark the sub-tree's leaves with inverted commas.

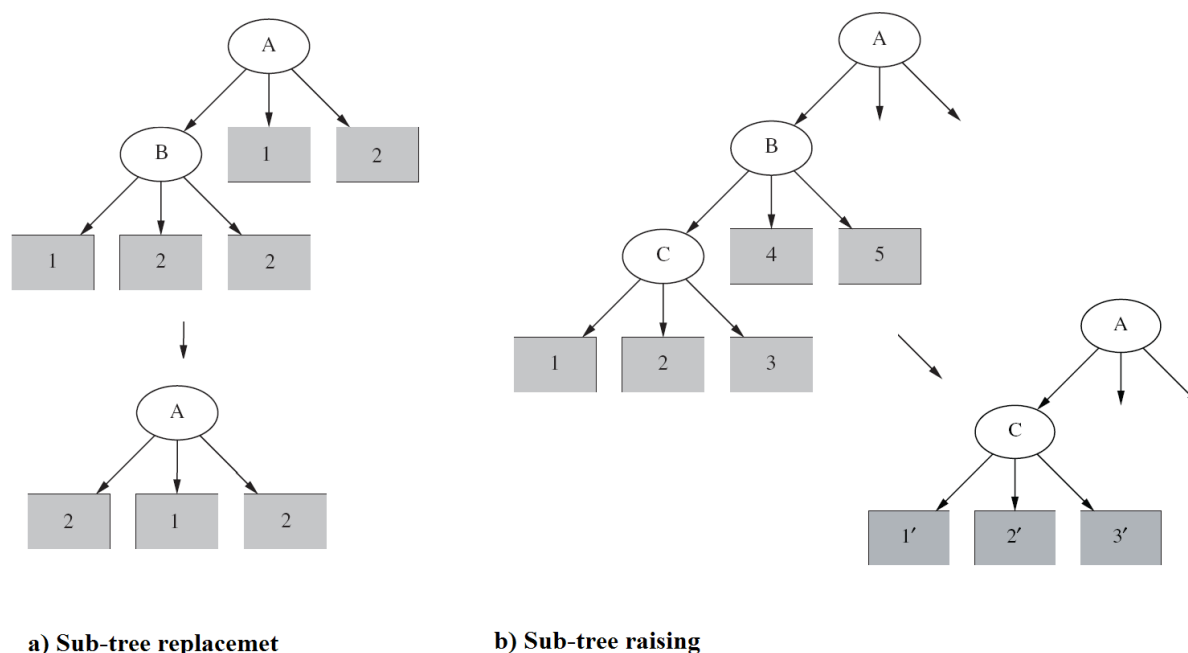


Figure 2.6: Sub-tree replacement and sub-tree raising (adopted from [51]).

C4.5 has replaced older versions of decision trees methods like ID3 [43] and CART [45] due to the following advantages:

- It is able to prune finished trees.
- It is able to handle continuous and discrete attributes. For continuous attributes, C4.5 creates a threshold and then splits instances into those whose attribute value is above the threshold and those whose attribute value is below than or equal to it [87].
- It is able to handle attributes with missing values which are not used in gain and entropy calculations.

In addition, an improved version of C4.5 algorithm has been introduced by Quinlan [51] and it addresses some of C4.5's disadvantages, such as slow tree building and high memory usage during the classification process. The new version (C5.0) is mostly used in commercial products, while researchers in academia still prefer the older version.

2.4.4 Support Vector Machines

The Support Vector Machine (SVM) is a supervised machine learning method introduced by Vapnik in 1995 [46] and which originates from statistical learning theories. SVM takes a set of input data and predicts, for each given input, membership to one of two possible classes. To do so, it defines a small set of points that lie on the border line of the problem space and builds a discriminative function $f(x)$ that finds the longest distance between instances with

opposite class values. It then constructs an N-dimensional hyperplane that optimally separates the data into the two classes of interest.

Figure 2.7 shows a simple classification example with two class values (+ and -) where the optimal separating hyper-plane is a line. The dashed lines mark the distance between the dividing line and the closest objects to the line. The distance between the dashed lines is called the *margin* and the instances that constrain the width of the margin are the *support vectors*.

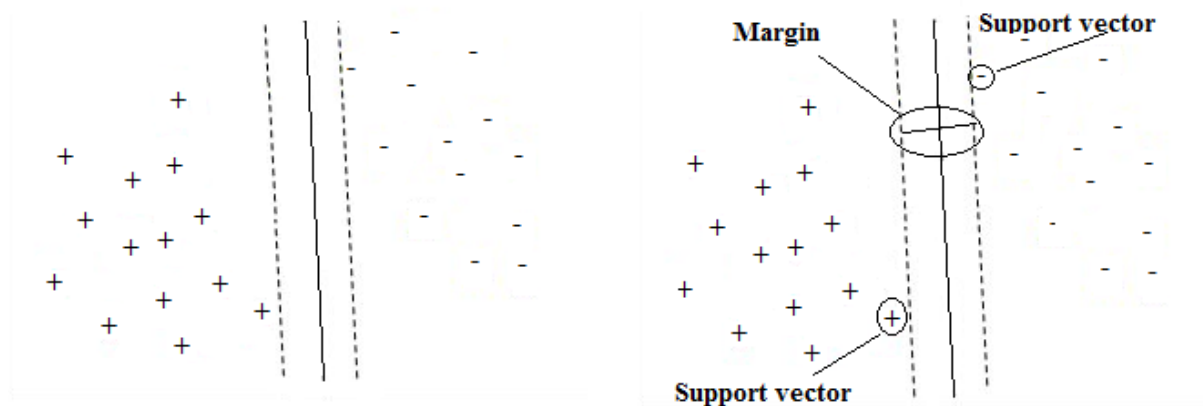


Figure 2.7: An example of a simple hyperplane.

It is easy to see that the solution shown in Figure 2.7 is not the only possible solution for the problem. In fact, there are an infinite number of possible lines that can divide the problem space. For instance, Figure 2.8 shows another possible solution. The SVM algorithm tries to find an optimal solution by choosing the separating line that maximises the margins. In this context, the line on Figure 2.8 is considered to be superior to that of the first example.

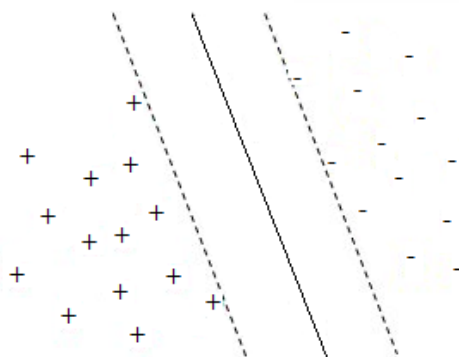


Figure 2.8: Another example of a hyperplane.

While the hyperplane used in the examples is a simple linear line, in practice, however, often the data cannot be separated linearly. In these cases SVM transforms the original space, to a new space with a higher dimension. The transformation is done using a *kernel function* which maps the data into the new space where the hyperplane can separate the objects. Figure 2.9 shows a two-dimensional problem (left) where objects cannot be separated with a linear line. To solve the problem, SVM transforms the two-dimensional space into a three dimensional one where it is possible to separate the objects with a simple plane (right).

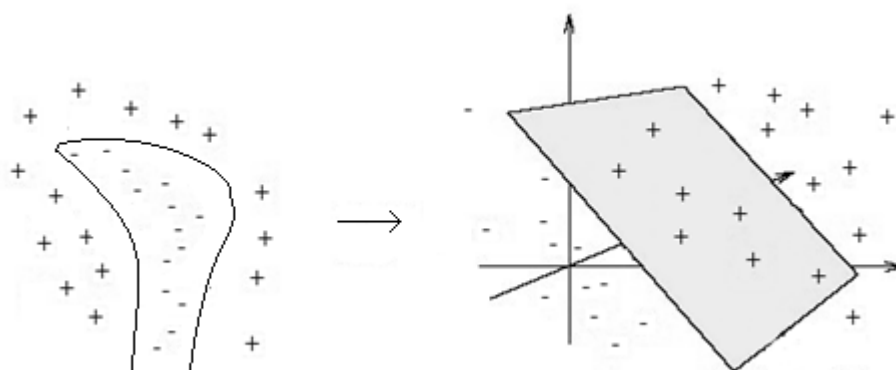


Figure 2.9: Transformation of a two-dimensional space.

Several kernel functions can be used for the transformation of the data presentation, with two of the most common being the polynomial kernel and the Gaussian radial basis function (RBF) kernel. If x presents an instance and y presents its class label, these kernel functions are defined as follows:

- the polynomial kernel:

$$k(x, y) = (\alpha x \cdot y + c)^d \quad (2.14)$$

- the Gaussian radial basis function (RBF) kernel:

$$k(x, y) = \exp(-\delta |x - y|^2) \quad (2.15)$$

Parameters α , c and d in the polynomial kernel and parameter δ in the RBF kernel are used to adjust the transformation and should be tuned to form an optimal transformation of the original space.

SVMs have been proven to be the most accurate classifiers in many areas, including protein solubility [29,30,34]. However, they do have several disadvantages [88]:

- Deciding which kernel to use for a specific problem and tuning its parameters can be problematic since several kernel functions can be used to make the transformation.
- They tend to be computationally intensive both in terms of processing speed and terms of memory usage.
- Only binary SVM classifiers can be built.
- The results are not easy to interpret and there is no knowledge about the decision making of the classifier.

2.5 Parameters of machine learning techniques

In previous sections we have described different techniques for building machine learning classifiers. One of the major challenges when using these methods is the tuning of the parameters (e.g. the k parameter for k -nearest neighbours classifier, and the kernel parameters for SVM) since their values significantly affect the quality of the classifier. There are two extreme situations that can occur when incorrect parameters are chosen: *over-fitting* and *under-fitting* of the learning model.

2.5.1 Over-fitting and under-fitting

Over-fitting occurs when the parameters are selected in such a way that they completely fit the classifier. While this increases the accuracy of the classifier, this kind of classifier might not generalize well to new data. Under-fitting occurs when the selection of parameters leads to a classifier that is too simple and might be too general.

Figure 2.10 illustrates these two concepts by showing two different classes of data which cannot be completely separated in a simple way. Again, the classes are represented by the positive '+' and negative '-' symbols, while the '?' symbol represents a new instance that has to be classified. Let us, assume that the correct class label of the new instance is '+' and that the isolated negative instance in the top right corner represents a noise point. The figure shows three different solutions. The first solution (b) illustrates the under-fitting concept: the chosen classifier (a simple line) is unable to separate both classes accurately and most of the positive instances are miss-classified. The second solution (c) illustrates the over-fitting concept: a too complex solution where the classifier tries to completely fit to the data. Although all training instances are classified correctly, due to its complexity, the classifier assigns the wrong class label (-) to the new instance. To avoid the problems illustrated by classifiers (b) and (c) a trade-off has to be made and the last part of the figure (d) illustrates a better solution for the data.

It is important to note that incorrect selection of attributes in learning data may also contribute to the described problems. Too many chosen attributes may force the classifier to over-fit while too less attributes may cause under-fitting.

To avoid both problems it is therefore essential to define attributes carefully and use the feature selection/extraction methods described in previous sections. In addition, the parameters for building classifiers have to be optimized and several techniques for finding the best parameter values have been proposed [49]. The following section describes two techniques for finding optimal parameters for machine learning algorithms [51].

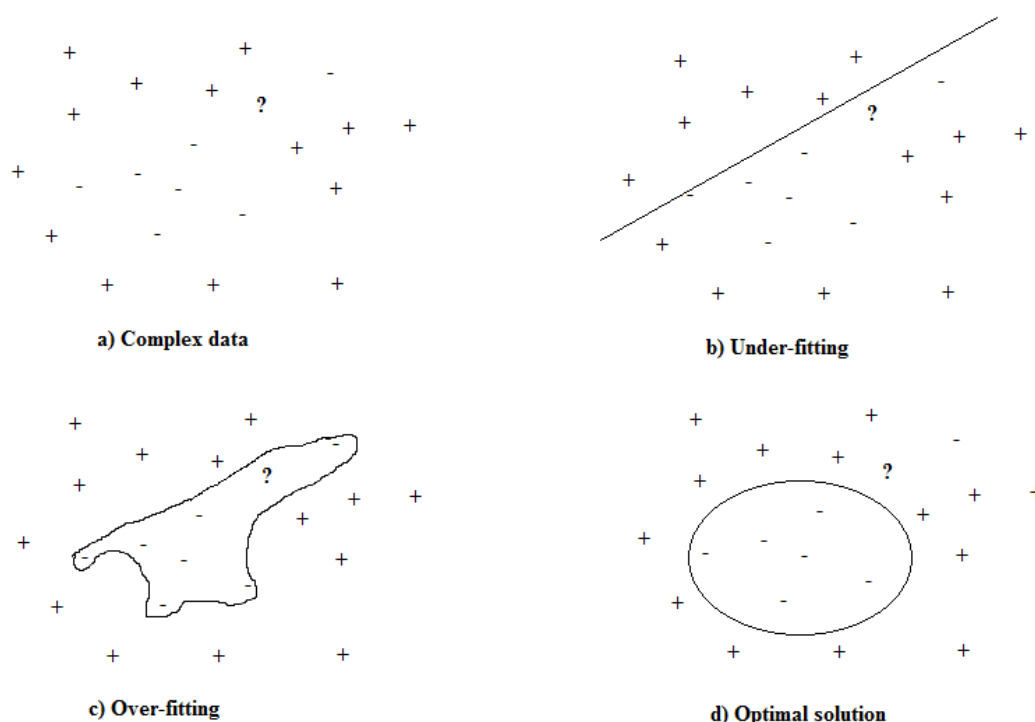


Figure 2.10: Several different solutions for a classification problem where data cannot be completely separated with a simple solution.

2.5.2 Optimizing parameters

This section describes the two methods for optimizing algorithm parameters that have been considered for this thesis. They were chosen due to their ability to optimise more than one parameter at the same time. The first technique is called *grid search* and it tries all possible values of each parameter across the specified search range using geometric steps. The second method is called *pattern search* and it tries to lower the number of tries by starting at the centre of the specified search range and making trial steps in each direction for each parameter. When the learning model improves, the search centre moves to the new point and

the process starts again. When no improvement is found, the step size is reduced (initial size can be set by the user) and the search is tried again. The method stops when the search step size is broken down to a pre-specified tolerance parameter.

An important difference between the two methods is their computational time performance. Since grid search technique has to be evaluated at many points within the grid for each parameter, it uses a lot of computer resources and, therefore, takes longer to find the optimal parameter values. For instance, if we want to find optimal values for three parameters in steps of 10, we need to perform 1000 evaluations. On the other hand, pattern search could require fewer evaluations since it follows a path and usually does not search through the whole grid. For this reason, pattern search is used for optimizing parameters for learning algorithms in this thesis. However, the main disadvantage of pattern search is that it could miss global optimal parameter values.

2.6 Evaluating machine learning algorithms

2.6.1 Evaluation of a single classifier

To evaluate a built classifier one must determine its effectiveness when used with real data. It would be ideal if the algorithm could be tested and evaluated on all possible input values. Since this is clearly not possible, there are several metrics available to measure the approximate effectiveness of a machine learning algorithm. Suppose we wish to estimate a classifier's performance on a test dataset consisting of n instances. It is common to present the predicted and actual classifications using a *confusion matrix*, i.e. a matrix of size $L \times L$ where L represents the number of different class attribute values and the value in cell c_{ij} of row i and column j provides the number of instances known to have class value i that have been classified as class j . Table 2.4 shows a confusion matrix for a binary classification problem ($L=2$) where only two classes are defined: Negative and Positive. In this case the values of each of the four cells correspond to four very well known concepts in machine learning: true negative/positive (TN/TP) and false negative/positives (FN/FP). TN/TP corresponds to instances that have been assigned the correct class label (either Negative or Positive) while false FN/FP represents the number of instances that have been assigned a wrong class label (either Negative or Positive).

Table 2.4: A confusion matrix for a binary classification problem.

Actual/Classified	Negative	Positive
Negative	TN	FP
Positive	FN	TP

Several metrics have been defined with the help of the confusion matrix. For the particular case of protein solubility classification the following are the most common [30, 34].

- *Accuracy* is calculated as the number of correctly classified instances divided by the total number of instances.

$$Accuracy = \frac{TP + TN}{n} \quad (2.15)$$

It gives the probability of correct classification. While this is one of the most commonly used metrics, when used alone it often does not provide enough information about the success rate of a classifier. For instance, imagine an input dataset where the proportion of each class in the learning data is 70% for Positive and 30% for Negative. A constant classifier with classification result of Positive would result in 70% accuracy, even though the classifier would be highly non-informative and useless.

- *Sensitivity* of positive/negative class is computed as the number of correctly classified instances from the positive/negative class divided by the number of all classified instances from the positive/negative class.

$$Sensitivity\ of\ positive\ class = \frac{TP}{TP + FN} \quad (2.17)$$

$$Sensitivity\ of\ negative\ class = \frac{TN}{TN + FP} \quad (2.18)$$

It gives the probability that an instance of the positive/negative class will be classified as correctly.

- The *precision* measurement of positive/negative class (also called positive/negative predictive value) is computed as the ratio of correctly classified positive/negative values to the number of all instances classified as positive/negative.

$$Precision\ of\ positive\ class = \frac{TP}{TP + FP} \quad (2.19)$$

$$Precision\ of\ negative\ class = \frac{TN}{TN + FN} \quad (2.20)$$

Precision gives the probability for a positive/negative prediction to be correct.

- *Matthews Correlation Coefficient* (MCC) is an effective evaluation metrics for problems with unbalanced datasets (i.e., datasets where there are many more members of one class than another) and is defined as [34]:

$$MCC = \frac{TP * TN - FP * FN}{\sqrt{(TP + FN)(TP + FP)(TN + FP)(TN + FN)}} \quad (2.21)$$

It can be noted, that MCC uses all for numbers from the confusion matrix to calculate a single value. This value can range between -1 and +1, where the former indicates an inverse classification and the latter indicates a perfect classification. The correlation coefficient for completely random predictions has value zero. MCC has often been mentioned to represent the most balanced evaluation of the classifier's performance [24]. However, as other metrics, when used alone it is also unable to provide enough information for an optimal assessment. For instance, MCC can be very high for a confusion matrix with few or no FP that also has only few TP.

- The *Area Under the ROC (Receiver Operating Characteristics) curve* or shortly *AUC* is another metrics that may can be useful for unbalanced datasets. AUC is represented as a graphical plot (ROC graph) of the sensitivity of the positive class versus the sensitivity of the negative class. ROC curves depict relative tradeoffs between benefits (true positives) and costs (false positives). Figure 2.11 illustrates a sample ROC curve represented with the curved line. Each point on the ROC curve represents a “*sensitivity of positive class versus sensitivity of negative class*” pair corresponding to a particular decision threshold. The diagonal line, where $y = x$, represents a classifier with performance that is equal to random guessing and the points in the upper left-hand triangle/lower right-hand triangle represent a classifier with a better/worse performance than a random classifier. The shaded area represents AUC and, roughly speaking, the larger this area the better a classifier's performance.

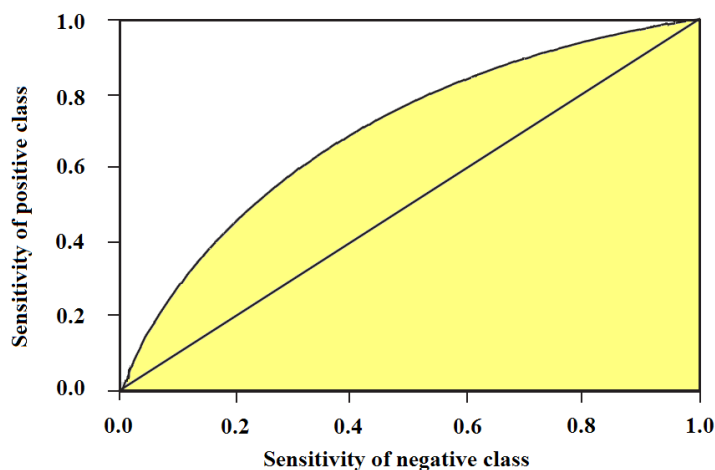


Figure 2.11: A sample ROC curve.

The detailed process of plotting ROC and calculating AUC is described in [25] and its value is equal to the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one. AUC can have values between 0.0 and 1.0 where the latter represents a perfect classifier and the former represents an inverse classifier.

2.6.2 Comparing the performance of classifiers on multiple datasets

When different classifiers are compared on multiple datasets, statistical validation of the results is needed to see if the observed differences in the results occurred by chance alone (*statistically insignificant* result) or not (*statistically significant* result). In this thesis, we compare pairs of classifiers (CI_i , $C2_i$) on several different datasets, where CI_i and $C2_i$ represent two classifiers built by machine learning algorithm i based on two different datasets. Statistical comparison of these classifiers is performed with the *Wilcoxon signed-ranks test* [59], which has been recommended over other statistical tests when comparing two classifiers on multiple datasets [53].

The Wilcoxon signed-ranks test works as follows. Let d_i be the difference between the performance scores of the two classifiers on i -th out of N data sets. The performance score can be Accuracy, MCC or any other evaluation metric value. The differences are ranked according to their absolute values and average ranks are assigned when the algorithms rank equally. Let R_+ be the sum of the ranks obtained for the datasets on which the second algorithm outperformed the first, and R_- the sum of ranks for the opposite. When d_i is 0, the ranks are split evenly among the sums and when there are an odd number of these differences, one is ignored. We write this as the following:

$$R_+ = \sum_{d_i > 0} \text{rank}(d_i) + \frac{1}{2} \sum_{d_i = 0} \text{rank}(d_i) \quad (2.22)$$

$$R_- = \sum_{d_i < 0} \text{rank}(d_i) + \frac{1}{2} \sum_{d_i = 0} \text{rank}(d_i) \quad (2.23)$$

Now, let us write the following definition:

$$T = \min(R_-, R_+) \quad (2.24)$$

where T is used to reject or confirm the so called *null hypothesis*, i.e. the hypothesis that there is no significant difference between the classifiers. The type of the sum of the ranks (positive or negative) that is higher determines which classifier is statistically significantly better. When the sum of the ranks for the positive differences is higher than the sum of the ranks for the negative differences, the new classifier is better and vice versa. For $5 < N < 26$, the difference between the classifiers' performances is known to be significant when T is equal or less to the corresponding values in the table of exact critical values for the Wilcoxon's test, part of which is presented in Table 2.5. The table contains critical values for three different confidence levels (i.e. the degree of certainty that a statistical prediction is accurate) $\alpha = 0.05$, $\alpha = 0.02$ and $\alpha = 0.01$. In general, lower the confidence level, more confident we can be that the statistical prediction is correct. In practice, most statistical books recommend the use of $\alpha = 0.05$.

Table 2.5: Critical values for the Wilcoxon-signed ranks test at $\alpha = 0.05$, $\alpha = 0.02$ and $\alpha = 0.01$. A classifier is significantly better than another if it performs better on at least w_α data sets.

N	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	25	25
w_{0.05}	0	2	4	6	8	11	14	17	21	25	30	35	40	46	52	69	66	73	81	89
w_{0.02}	/	0	2	3	5	7	10	13	16	20	24	28	33	38	43	49	56	62	69	77
w_{0.01}	/	/	0	2	3	5	7	10	13	16	20	23	28	32	38	43	49	55	61	68

For $N > 25$, the null hypothesis is tested with the following equation:

$$z = \frac{T - \frac{N(N+1)}{4}}{\sqrt{\frac{N(N+1)(2N+1)}{24}}} \quad (2.25)$$

where z is distributed approximately normally and the null hypothesis can be rejected when z is smaller than -1.96 (for $\alpha = 0.05$) [54].

Let us illustrate the described procedure on the following example. We measure the performance for two classifiers $C1$ and $C2$ on 12 different datasets ($Db1 - Db11$) with accuracy.

Table 2.6 contains the accuracies measured for both classifiers on all datasets. The Difference and Rank columns contain differences between the accuracies measured on each of the datasets and ranks of the classifiers, respectively. The null hypothesis is that the both classifiers perform equally well in terms of the classification accuracy.

Table 2.6: Comparison of accuracy measurements for two classifiers on 12 databases.

Database	Accuracy (C1)	Accuracy (C2)	Difference	Rank
Db1	0.763	0.768	0.005	3.5
Db2	0.599	0.591	-0.008	7
Db3	0.954	0.971	0.017	9
Db4	0.628	0.661	0.033	12
Db5	0.882	0.888	0.006	5
Db6	0.936	0.931	-0.005	3.5
Db7	0.661	0.668	0.007	6
Db8	0.583	0.583	0	1.5
Db9	1	1	0	1.5
Db10	0.94	0.962	0.022	11
Db11	0.972	0.981	0.009	8

The sum of the ranks for the positive differences is $R_+ = 3.5 + 9 + 12 + 5 + 6 + 11 + 8 + 1.5 = 56$ while the sum of ranks for the negative differences is $R_- = 7 + 3.5 + 1.5 = 12$. The minimum of the both sums is $T = 12$. According to Table 2.5 for a confidence level of $\alpha = 0.05$ and $N = 11$ data sets, the difference between the classifiers is significant when the smaller of the sums is equal or less than 11. Since $12 > 11$, we confirm the null hypothesis.

In practice the above steps are made by specialized software packages, such as IBM SPSS¹, which is also used in this thesis. When comparing two classifiers with SPSS an output value

¹ <http://www-01.ibm.com/software/analytics/spss/>

α is given, which represents the confidence level at which the classifiers are significantly different. To reject the null hypothesis this value should be less than 0.05 which is the recommended critical value.

3

Proteins

3.1 Introduction

This section presents the necessary background on proteins. In particular, it focuses on the information required to adequately understand the dataset used in the experiments performed in this thesis.

3.2 Protein structure

A protein is a large biological molecule made of amino acids linked into linear chains. Amino acids are organic compounds made of carbon, hydrogen, oxygen, nitrogen and (in some cases) sulphur, bonded in characteristic formations. They have a common part and a unique attached side chain which defines each amino acid. They react with each other to form a bond between the *carboxyl group* of one amino acid (-COOH) and the *amino group* (-NH₃) of a second amino acid. This linkage is called the *peptide bond* [89] and it enables amino acids to make large, chainlike molecules called polymers, which may contain as few as two or as many as several thousand amino acids. Figure 3.1 illustrates two amino acids forming the peptide bond and connecting into a simple polymer. The character R represents the side chain which is unique to each amino acid.

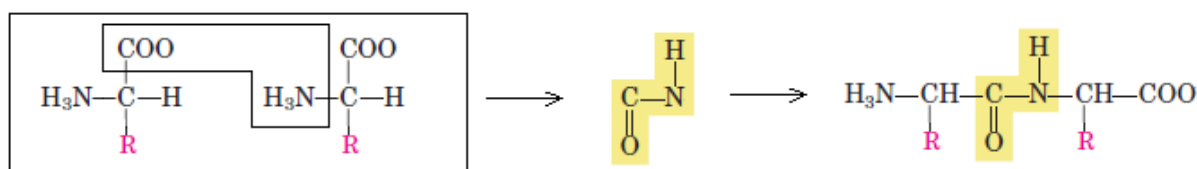


Figure 3.1: Two amino acids connect into a simple polymer.

An amino acid chain with more than ten amino acids is called a *polypeptide*, while a chain with 50 or more amino acids is known as a *protein*. An amino acid unit in a peptide is often referred to as a *residue* and it is the part left over after losing a water molecule when making the peptide bond. The main peptide bond chain in a protein's sequence is referred to as the (*poly*) *peptide backbone* and it is illustrated on Figure 3.2. Every protein starts with an *amino-terminal* (or *N terminal*) which is the residue with a free amino group (Amino acid 1 on the figure) and ends with a *carboxyl-terminal* (or *C terminal*) which is the residue with a free carboxyl group (Amino acid 3 on the figure).

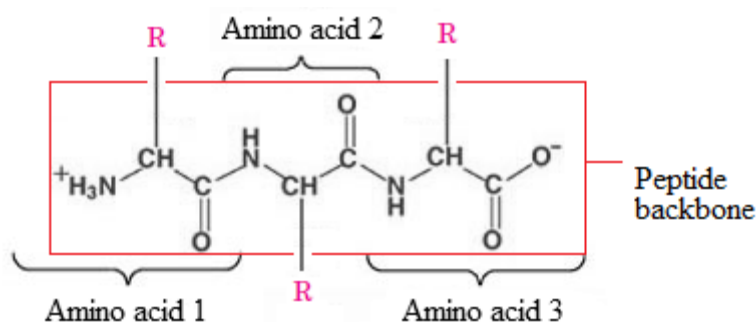


Figure 3.2: Peptide backbone.

Amino acids are commonly represented in the literature using either a three-letter or a one-letter abbreviation. Table 3.1 provides the abbreviations used for each of 20 standard amino acids that commonly occur in nature.

Table 3.1: Standard amino acid abbreviations.

Name	Three letter	One letter
Alanine	Ala	A
Arginine	Arg	R
Asparagine	Asn	N
Aspartic acid	Asp	D
Cysteine	Cys	C
Glutamic acid	Glu	E
Glutamine	Gln	Q
Glycine	Gly	G
Histidine	His	H
Isoleucine	Ile	I
Leucine	Leu	L
Lysine	Lys	K
Methionine	Met	M
Phenylalanine	Phe	F
Proline	Pro	P
Serine	Ser	S
Threonine	Thr	T
Tryptophan	Trp	W
Tyrosine	Tyr	Y
Valine	Val	V

The number of proteins of length L that can be made from 20 amino acids is enormous (20^L) and the specific properties of a protein are largely dependent on the kind and sequence of the amino acids in it [89]. The structure of a protein can be defined according to four different levels, as illustrated in Figure 3.3 (adapted from [89]).

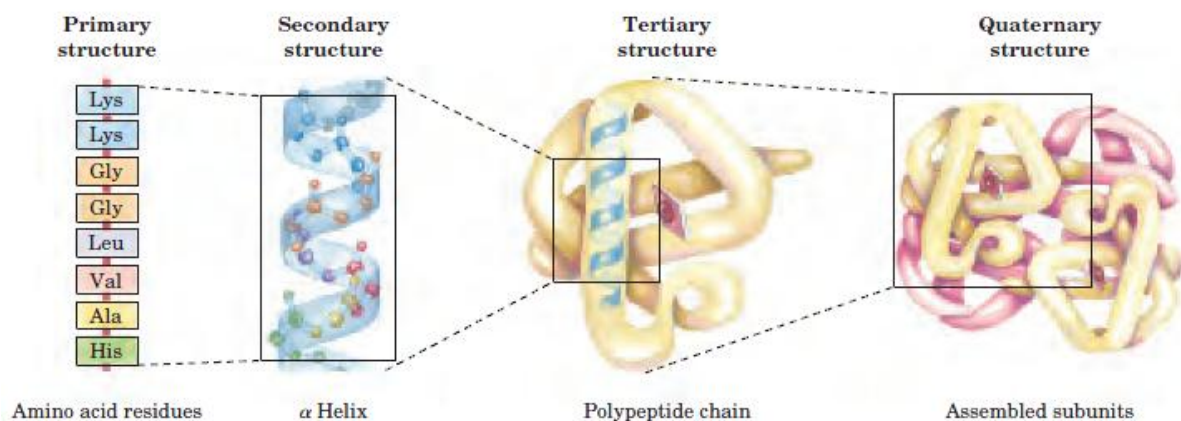


Figure 3.3: Levels of protein structure [89].

The *primary structure* of a protein (also referred to as the protein sequence) is given by its sequence of amino acids starting with the N terminal and ending with the C terminal.

The *secondary structure* of a protein results from the local conformation of some parts of its polypeptide chain. There are only a very few types of secondary structure that are stable and, therefore, occur widely in proteins. The most common are the α helix and β conformations. In the α helix structure the peptide backbone is tightly wound around an imaginary axis drawn longitudinally through the middle of the helix (like the one shown in Figure 3.3), while the β conformations organize polypeptide chains into sheets. Both conformations occur due to the hydrogen bonds that form between amino acids close to each other.

The *tertiary structure* is given by the 3D coordinates of the atoms in the protein and completely defines the structural organization of the protein molecule in 3D. Unlike the secondary structure, where only the interaction between adjacent residues are considered, in the tertiary structure we also consider the interaction between amino acids that are far apart in the sequence and that reside in different types of secondary structure. The physical process where a polypeptide chain folds into its 3D structure is called *protein folding*.

The *quaternary structure* applies to protein complexes, which are formed by the interaction between several protein molecules or subunits. Importantly, it is tertiary and quaternary structures that determine functional and structural roles of proteins, i.e., proteins with the

same tertiary/quaternary have the same function, even if their primary structure is different. This is important since proteins are involved in most live functions and processes. For example, some proteins are the base for structures such as muscle, tendons, hair, skin, and cartilage, while others are enzymes and act as catalysts in different biochemical transformations.

3.3 Protein solubility

Protein solubility is an important protein property since low protein solubility can lead to several diseases [3]. In addition, protein solubility also affects an important process called *protein purification* [2], whose goal is to isolate (or produce) a single protein from a complex mixture. Protein purification is important for the characterization of the function, structure and interactions of proteins and, unfortunately, only soluble proteins can be effectively isolated. Therefore, low protein solubility is a concern in many biomedical experiments [86, 58]. However, the concept of protein solubility is not particularly well defined, with different research groups using different definitions. Below two of the most common definitions of low protein solubility are described.

Low *in vivo* solubility refers to the solubility of proteins that are, before being purified, produced in live cells, usually in a bacterium named *Escherichia coli* (*E.coli*) [29]. *E.coli* is commonly found in the lower intestine of warm-blooded organisms and it is one of the most widely used hosts for the production of proteins of different organisms [3]. This production process is referred to as *heterologous production (over-expression) of recombinant proteins*. Sometimes these proteins aggregate in *E.coli* cells and form aggregates called *inclusion bodies* which are insoluble and include miss-folded proteins that can be a reason for several diseases and are harder to purify. Therefore, we say that a protein is insoluble *in vivo* when it is not soluble upon over-expression in *E.coli*. A significant amount of research has been done to determine why only some proteins are soluble when over-expressed in *E.coli* while others form insoluble inclusion bodies. While the answer is not clear yet, it has been shown that protein sequence plays an important role in this process [29-31] and there are several techniques that sidestep the low solubility problem, such as using a different host or strain of *E.coli*, changing over-expression conditions (e.g. temperature of the process) [83] and using additives which help proteins fold correctly (e.g. adding *chaperones* - proteins that help other proteins to fold) [84, 85].

The second most common definition of low protein solubility is called low *in vitro* solubility and, in contrast with *in vivo* solubility, refers to the solubility of proteins that are produced in protein-synthesizing systems that do not use living cells. The *in vitro* production of

proteins can have some advantages compared to the over-expression of proteins in host organisms (e.g. E.coli), since some proteins can be toxic to the host cell or form insoluble inclusion bodies in it. However, some proteins are also insoluble when produced in vitro and they aggregate to inclusion bodies.

3.3.1 Gathering soluble proteins

Determining protein solubility is often a trial and error process with a low success rate. Therefore, there have been several attempts at building an efficient protein solubility classifier [29-31, 34]. In doing so, researchers have used different techniques to collect soluble proteins and several public available databases including TargetDB [47], Protein Data Bank (PDB) [71] and SwissProt [72] have been used as inputs for solubility classifiers. However, the two definitions of low protein solubility described in the previous section are often a reason for confusion when using these databases. For instance, some proteins are found to be insoluble in E.coli, but they are proven to be soluble in vitro with different techniques (e.g. with refolding [4, 64-67]). Furthermore, the primary objective of databases such as TargetDB or PDB is to store the structure (rather than the solubility properties) of the protein and solubility information is therefore not systematically documented. For example, while proteins in the TargetDB database are equipped with a *soluble* tag, a missing annotation does not necessarily mean an insoluble protein. In addition, the value of the soluble tag may change through time, since a protein that was insoluble in vivo could be solubilised in vitro after further experimentation. Also, in these databases it is hard to identify which proteins were over-expressed in which host.

For these reasons, selecting the most appropriate database for the experiments performed in this thesis was an important step. To do so, we reviewed the most recent in vivo solubility protein prediction techniques [29, 30, 34] and analysed the datasets that were used in those experiments. It was concluded that the following two databases were the most appropriate candidates for the experiments:

- The SOLpro database [34]: contains the most up to date data for protein solubility classification which is freely available online and also includes proteins from previous works such as [29]. The database is perfectly balanced as it contains 8,704 soluble and 8,704 insoluble proteins. The main disadvantage of SOLpro is that contains a large proportion of TargetDB proteins, many of which are missing important information regarding their solubility. In addition, data in SOLpro are collected from several different research groups, each of which used different experiment conditions when measuring solubility.
- The eSol database [73]: includes information about protein solubility of the entire ensemble of E.coli proteins (i.e. 4,132 proteins) which were produced in vitro. This

database was prepared by a single group and the solubility of each protein was measured in the same laboratory under the same conditions.

After careful consideration, we decided to use eSol for two main reasons. Firstly, having data with as little noise as possible is crucial for building classifiers with good performance and SOLpro's reliance on TargetDB is too important to be ignored. Since eSol is gathered by one research group it is more likely to contain less noisy data as SOLpro. Secondly, in this thesis we perform experiments with three different machine learning algorithms, two feature selection methods and around 4,300 new attributes, which is computationally intensive and we cannot afford to perform all these experiments on very large datasets. SOLpro contains almost 18,000 proteins while eSol contains only around 4,000 proteins. Therefore, although, in general, it is better to use as many instances for training machine learning algorithms as possible, in this specific case we decided to use eSol. In the next section we describe this database in more details.

3.3.2 The eSol database

Proteins in the eSol database were produced with the help of PURE (Protein Synthesis Using Recombinant Elements) [73], an in vitro protein-synthesizing system. PURE can produce different types of proteins and it does not use live cells in this process. Instead, it uses only essential E.coli factors [68] responsible for protein synthesis. In the case of eSOL, PURE is used to produce the entire ensemble of E.coli proteins. Therefore, we can say (very superficially) that "an artificial" E.coli host is used to produce E.coli proteins. In addition to its capability of producing proteins, PURE also allows the evaluation of the solubility of individual proteins.

To form the eSol database all E.coli proteins were produced under the same conditions using PURE and no chaperons or other additives were used to increase the solubility. After its production, each protein was given a solubility index (SI) with the help of a special centrifugation assay technique [69]. With this method 2,277 out of 4,132 all E.coli proteins were given their SI on a scale from 0 to 120 percentage points. The rest could not be quantified for specific reasons (see [73] for details) and were therefore ignored. The histogram on Figure 3.4 represents frequencies of proteins with different SI. The x axis represents the SI values while the y axis represents the number of proteins for each of these values. The histogram shows that the distribution is clearly bimodal, rather than normal Gaussian. Based on this distribution the proteins were divided in two groups: highly insoluble ($SI < 30$) and highly soluble proteins ($SI > 70$). Proteins with SI indexes between these two values were not included in the experiment. As a result, the final dataset contained 1,625 proteins, out of which 782 are insoluble and 843 are soluble proteins.

The authors of the eSol database classified the proteins with SVM and reported the accuracy of 80%. Unfortunately, they did not report results for other metrics, therefore, we repeated the experiment following their steps so we were able to compare their methods with ours (see Section 6).

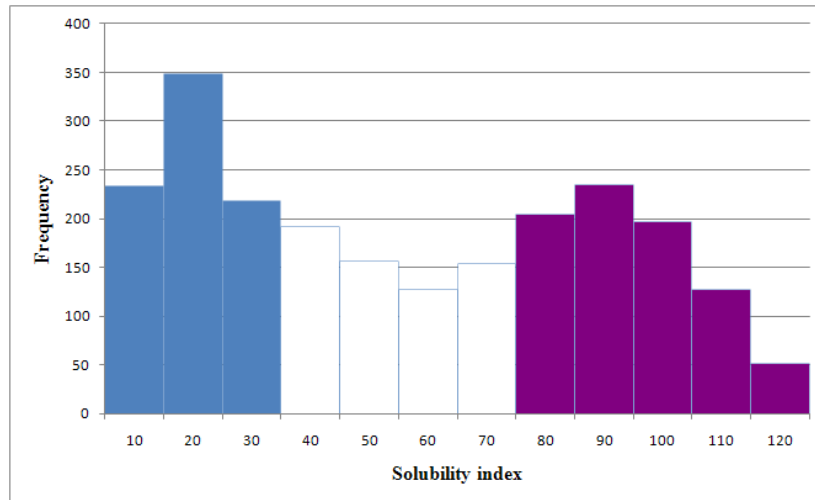


Figure 3.4: Solubility index frequency for cytoplasmic proteins.

4

Text mining and biomedical concepts

4.1 Introduction

Today, more and more information is saved electronically than ever before and a large portion of that information is in the form of text such as books, scientific journals, newspapers, online forums and blogs. It is impossible for humans to process all these information and identify useful knowledge. Also, the rapid increase in scientific publications makes it almost impossible for the individual scientist to keep up to date with the current literature knowledge. Therefore, natural language processing fields such as text mining and text retrieval [70] have been developed to cope with these problems. In the scope of this thesis, several techniques from these fields are used to (a) extract knowledge about proteins from MEDLINE abstracts and (b) present this knowledge in the form of so called *biomedical concepts* which are later used as descriptive attributes in protein solubility classification. Therefore, Section 4.1.1 provides the resumed background on text mining in general and Section 4.1.2 introduces search engines for biomedical concepts.

4.1.1 Text mining

The purpose of text mining is to process texts and extract meaningful patterns from them with the help of machine learning algorithms. An example of a text mining application is measuring sentiment in text [80], where the algorithm tries to determine the general tone (e.g. happy or sad) of a document. For instance, researchers have measured sentiment in blog entries to predict movie sales [57]. Some other applications of text mining include document classification [81] and information security [82].

The raw text cannot be used directly by text mining algorithms, it has to be analysed and put into the correct form. In particular, after the text is gathered and unformatted (e.g. font styles are removed) it goes through five phases: *tokenization*, *segmentation*, *stemming*, *lemmatization* and *part-of-speech tagging*. Let us briefly discuss each of these phases.

During tokenization a text from a document is broken up into words, phrases, symbols, or other meaningful elements called *tokens*. During this step it is important to know what kind of the text is being analysed. For instance, tokens in scientific literature may differ from those in blogs or forums which use different language style (e.g. slang). Segmentation identifies meaningful units including sentences, topics and paragraphs. Stemming is the process of reducing declined, conjugated or derived words to their stem (also referred to as base or root) form. For example, the word “cat” is a stem for words like “cats”, “catlike” and “catmint”. Lemmatization is a process that defines the lemma (i.e. canonical, dictionary, or citation form) for every word (or group of words) in the text. For instance, words “run”, “runs” and “ran” have “run” as their lemma. Lemmatization and stemming are closely related and the difference is that the latter considers only one word at the time and does not possess any knowledge of the context, while the former considers relations in the text and it is able to discriminate between words that have different meanings depending on part of speech. Although, the stem and the lemma of a word are often identical (in English language), that is not always the case. For example, the word “promised” has “promis” as its stem and “promise” as its lemma. The last phase of document analysis, called part-of-speech tagging, marks every word in the text according to its category such as noun, verb or adjective.

After the text is processed in the described way it can be used and manipulated by text mining algorithms. The whole process of collecting the text and preparing it for text mining is illustrated on Figure 4.1.

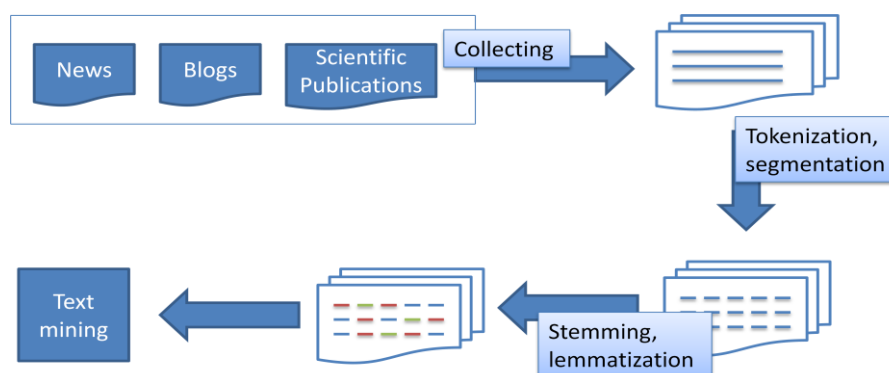


Figure 4.1: Document analysis steps.

To use text as the input for text mining algorithms, different representations of text have been proposed. For instance, every document can be defined as a vector d :

$$d = \{a_1, a_2, a_3, \dots, a_i, \dots, a_m\}$$

where a_i represents a single attribute and m represents the maximum number of attributes. An attribute can be a word, part of a word (e.g. its lemma) or even a number of words. Then, N documents can be represented as a set of documents (vectors) with vector D :

$$D = \{d_1, d_2, d_3, \dots, d_i, \dots, d_N\}$$

where d_i represents a single document vector. A set of documents can be then defined as a two dimensional array as is shown on Table 4.1. The values 1 and 0 in the table represent whether the attribute appears in the corresponding document or not, respectively. This document presentation can now be used in text mining algorithms for classification and extraction of different patterns, with each row representing an instance and each column representing an attribute.

Table 4.1: An example of document presentation.

	a_1	a_2	a_3	...	a_i	...	a_m
d_1	0	1	1	...	0	...	1
d_2	1	0	0	...	1	...	1
d_3	1	0	0	...	1	...	0
...
d_i	0	1	0	...	0	...	1
...	0
d_N	1	0	0	...	0	...	0

4.1.2 Extraction of biomedical concepts

This section describes how text mining can be used to extract useful knowledge about proteins. A lot of information about proteins is published in scientific publications which are available through literature search portals (e.g. NCBI's PubMed/MEDLINE [61]). However, these portals support the scientists only in finding their respective problem-related publications and they do not offer direct extraction of patterns such as which proteins are associated with certain type of disease. Scientists have to search for these knowledge manually which can be unpractical. Therefore, several text mining techniques and tools have been developed to deal with the problem. Applications such as LitMiner [62], XplorMed [60], MedlineR [26] and FACTA [50] search through medical publications and return a ranked list of potentially related key terms (called biomedical concepts) for a given query

key term. With these existing systems, biomedical concepts are limited to genes/proteins, diseases, symptoms, drugs, enzymes and chemical compounds.

In this thesis, FACTA is used to extract biomedical concepts from scientific literature due to its ability to pre-index words and concepts, which results in fast, real-time responses of the system. This is important since the classification scheme used for our experiments (see Section 5) is already very time consuming. In addition, FACTA also accepts the most customized query amongst all the systems and it can be used as a SOAP² web service, which makes it easy to use inside computer programs. The main disadvantage of the system is the fact that it searches only through MEDLINE abstracts rather than through full papers or through papers in any other database. However, since MEDLINE currently contains almost 19 million records³, this will produce enough biomedical concepts for the purposes of this thesis.

We define the term “biomedical concept” as follows:

A biomedical concept is a word or a word association extracted from MEDLINE abstracts using FACTA and that represents genes/proteins, diseases, symptoms, drugs, enzymes or chemical compounds.

Concepts associated with a protein can be retrieved by a search query containing either the name of the protein (e.g. “p53”) or its UNIPROT [72] accession number (e.g. “UNIPROT:P04637”). The query can also be used to indicate which concepts should be returned (e.g. only disease names or disease names with symptoms). After inputting the query, FACTA retrieves the documents that match the query from MEDLINE. The concepts mentioned in the retrieved documents are counted and ranked according to their relevance to the query. Different ranking schemes can be used and the following three ranking criteria are supported by FACTA:

- *Frequency*: counts the number of times a concept appears in the MEDLINE abstracts.
- *Point-wise mutual information*: computed as

$$\frac{\log p(x, y)}{p(x)p(y)} \quad (4.1)$$

² SOAP is a simple XML-based protocol to let applications exchange information over HTTP
http://www.w3schools.com/SOAP/soap_intro.asp

³http://www.nlm.nih.gov/bsd/revup/revup_pub.html#med_update (visited on 25/6/2011)

where $p(x)$ is the proportion of the documents that match query x , $p(y)$ is the proportion of the documents that contain concept y , and $p(x, y)$ is the proportion of the documents that match the query and contain the concept. It gives an indication of how much more the query and concept co-occur than would be expected by chance.

- *Symmetric conditional probability*: computed as the product of two conditional probabilities

$$P(x | y) * P(y | x). \quad (4.2)$$

where $P(x | y)$ is the conditional probability that query x appears in the document given concept y and $P(y | x)$ is the conditional probability that concept y appears in the document given query x . Symmetric conditional probability shows how likely are the concept and the query to be collocated.

In this thesis we used the frequency metric since it was the only metric supported through the SOAP web service at the time of writing. In addition, we believe that counting the number of times that a concept appears in the literature sufficiently represents the concepts relevance to the query. Figure 4.2 shows the biomedical concepts and their frequencies returned by FACTA for the query “p53” (an example of a human protein). Figure 4.3 provides a closer look of a document where the concept “tumour” is found to be associated with the query and, therefore, used to increase the frequency of the concept.

Query: UNIPROT:P04637 (p53)

55,628 document(s) hit in 20,033,079 MEDLINE articles (0.10 seconds)

Concepts found in the documents ranked by [Frequency | Pointwise Mutual Information | Symmetric Conditional Probability] .

Human Gene/Protein		Disease		Symptom		Drug		Enzyme		Compound	
p53	55628	tumor	46386	starvation	161	Progesterone	1336	ERK	1545	DNA	24652
p21	6962	cancer	13999	Anoxia	108	Gel	712	caspase-3	1492	Serine	2887
Polymerase	6306	carcinoma	6524	collapse	64	Glutathione	644	mitogen-activated protein kinase	1341	Serine	2487
Cyclin	6148	Breast Neoplasms	4150	pain	55	Paraffin	496	protein kinase	1279	Threonine	2402
bcl-2	5737	carcinogenesis	3749	seizures	54	Nitric Oxide	486	cdc2	776	oxygen	1537
Bcl-2	3368	Adenocarcinoma	3604	erythema	52	Etoposide	446	proteasome	691	Estrogen	1192
ras	3274	Carcinoma	3494	ataxia	49	Fragment	407	APC	614	Estrogen	1174
Ki-67	3081	Squamous Cell	3482	hyperoxia	43	lysine	277	MAP	596	Cisplatin	1113
		breast cancer		abdominal	35	Tretinoin	234			tyrosine	1008

Figure 4.2: An example of associated biomedical concepts for a search query “p53”.

Down-regulation of Orai3 arrests cell-cycle progression and induces apoptosis in breast cancer cells but not in normal breast epithelial cells.

... This phenomenon is associated with a reduction in CDKs 4/2 (cyclin-dependent kinases) and cyclins E and D1 expression and an accumulation of p21(Waf1/Cip1) (a cyclin-dependent kinase inhibitor) and **p53** (a **tumor**-suppressing protein). ...
PMID:20683915 J. Cell. Physiol. 2011 Feb

Figure 4.3: An example of a document where tumour is associated with protein p53.

5

The experimental environment

5.1 Introduction

This section describes the experimental environment for protein solubility classification. In particular, it focuses on the selected designed classification scheme, the methods for its evaluation and the software tool used for its implementation.

Figure 5.1 presents a high-level view of the classification scheme where the main steps of our experiments are illustrated. It can be noted that two types of attribute datasets (i.e., pairs of attribute names and values) for the eSol proteins are extracted: *Sequence derived attribute datasets (SdAd)* and *Biomedical concept attribute datasets*. Both datasets are then merged to form the *Merged attribute datasets (MAd)*, on which two feature selection methods (i.e., Information Gain and ReliefF) are used to remove any irrelevant attributes. Then, three different classifiers (i.e., Naïve Bayes, SVM and Decision trees) are built and evaluated using the SdAd and MAd attribute datasets. The performances of these classifiers are then compared to confirm or to reject the hypothesis given in Section 1.2.1.

We have already described the eSol database in Section 3.3 which contains information about 1,625 proteins and which is used in the classification process. Section 5.2 presents the protein attributes that we have extracted from both protein sequences and MEDLINE articles with the help of FACTA [50]. Section 5.3 provides a detailed description of the classification and evaluation steps, and provides a brief introduction to the Weka tool, which is used in the implementation process.

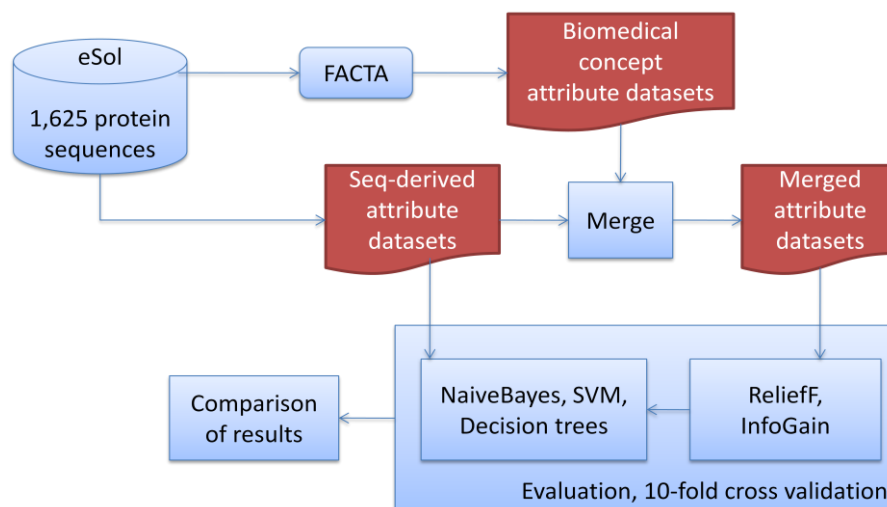


Figure 5.1: Classification scheme.

5.2 Protein attribute/feature datasets

The protein attribute (feature) datasets used in this thesis can be grouped into three groups: the sequence-derived attribute datasets, the biomedical concept attribute datasets and the merged datasets. Each of these three groups is described in more detail in the following subsections.

5.2.1 Sequence derived attribute datasets

As mentioned in Section 2.3.1, determining the right attributes to use during the machine learning process is an important step. For this reason, researchers have put significant amounts of effort towards finding the most relevant attributes that describe proteins well enough to be used in the classification process. Several web services such as PROFET [48] and ProtParam [75] can be used to automatically compute a range of protein attributes. These systems take as input a protein and return a series of values, one per pre-defined attribute (e.g. weight, length). Most of the attributes that researchers have used are sequence-derived, i.e. they are calculated directly from the protein sequence, with the most common attributes being the frequencies of mono-, di- and tri-peptides (i.e., all possible combinations of amino acid groups of one, two and three amino acids, respectively) using different alphabets.

The main reason to use different alphabets is as follows. The standard (also called natural) alphabet consists from 20 letters where each letter represents one of the 20 amino acids found in proteins. From a combinatorial standpoint there is an almost endless variety of sequences that can be made from this alphabet, e.g. for a protein of length 100 there are 20^{100}

possible combinations. It has been proven that only a fraction of these chains fold in stable 3D structures and several groups investigating protein folding suggest that protein folding can be achieved with fewer components than the 20 naturally occurring amino acids [19]. Therefore, several reduced alphabets have been proposed and Table 5.1 describes the seven most commonly used in protein solubility classification [29,30,34]. The values in brackets are used to identify groups that consist of more than one amino acid.

- The Natural alphabet consists of twenty letters, where every letter represents the corresponding amino acid.
- The Hydropho alphabet groups amino acids according to their hydrophobicity [39], i.e. the degree by which the amino acid is repelled from a mass of water. The alphabet was defined by [74].
- The ConfSimi alphabet groups amino acids according to their conformational similarity as defined in [26].
- The BlosumSM alphabet groups amino acids according to the so called blocks substitution matrix (BLOSUM) [19], which is built based on the substitutions of observed alignments of much conserved regions of protein families, as contained in the Blocks database [16].
- The ClustEM14 and ClustEM17 alphabets use the Expectation-Maximization algorithm [20] to group amino acids according to eight numeric scales obtained from the Amino Acid Index Database [21]. Both alphabets are described in detail in [30].
- PhsChem groups amino acids according to the following physico-chemical properties: aliphatic, aromatic, positively charged, sulphurated, negatively charged, amide and alcohol.

Table 5.1: Amino acid alphabets used to compute frequencies for monomers, dimmers and trimmers.

Name	Size	Amino Acid Groups	Reference
Natural	20	A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y	Natural alphabet
Hydropho	5	CFILMVW (1), NQSTY (2), DEKR (3), AG (4), HP (5)	[29]
ConfSimi	7	ACEKLMQR (1), FHXY (2), ITV (3), DN (4), G, P, S	[29]
BlosumSM	8	CILMV (1), DENQ (2), FWY (3), AG (4), KR (5), ST (6), H, P	[29]
ClustEM14	14	DEQ (1), AH (2), FW (3), IV (4), ST (5), C, G, K, L, M, N, P, R, Y	[30]
ClustEM17	17	DE (1), IL (2), NQ (3), A, C, F, G, H, K, M, P, R, S, T, V, W, Y	[30]
PhsChem	7	AGILPV (1), FWY (2), HKR (3), CM (4), DE (5), NQ (6), ST (7)	[34]

Let us illustrate the grouping of amino acids into mono-, di- and tri-peptides for different alphabets by means of an example. Consider the following amino acid sequence in the Natural alphabet: “CDQVAGW”. Table 5.2 shows in its second column the conversion of this sequence into each alphabet and, in the last three columns, it shows some examples of

mono-, di- and tri-peptides found in the sequence. The frequencies of each of these possible combinations are calculated and used as attributes.

Table 5.2: The Sequence representation for different alphabets.

Name	Converted Sequence	Mono-peptides	Di-peptides	Tri-peptides
Natural	CDQVAGW	C, D, Q, ...	CD, DQ, QV, ...	CDQ, DQV, QVA, ...
Hydropho	1321441	1, 3, 2, ...	13, 32, 21, ...	132, 321, 214, ...
ConfSimi	14131G3	1, 4, 1, ...	14, 41, 13, ...	141, 413, 131, ...
BlosumSM	1221443	1, 2, 4, ...	12, 22, 21, ...	122, 221, 214, ...
ClustEM14	31142G3	3, 1, 4, ...	31, 11, 14, ...	311, 114, 142, ...
ClustEM17	C13VAGW	C, 1, 3, ...	C1, 13, 3V, ...	C13, 13V, 3VA, ...
PhsChem	4561112	4, 5, 6, ...	45, 56, 61, ...	456, 561, 611, ...

Besides the frequencies of mono-, di- and tri-peptides, other sequence-derived attributes have been used for protein solubility classification. Table 5.3 describes the most popular ones.

Table 5.3: Attributes computed from protein sequence.

Name	Description
Sequence length (SeqLength)	Number of amino acids in the sequence.
Molecular weight/mass (Mw)	The ratio of the average mass of one molecule of an element or compound to one twelfth of the mass of an atom of carbon-12
Grand average of hydrophobicity index (GRAVY) [73]	Sum of the hydrophobicity values of all the amino acids, divided by the number of amino acids in the sequence.
Isoelectric point (Ip)	The pH at which the net charge of the protein is neutral (the protein carries no net charge since it has an equal number of positive and negative charges).

In our experiments we have concentrated on the attributes that have been previously proved to be relevant for solubility classification from each alphabet. These attributes are grouped into 20 sequence-derived attribute datasets (SdAd) which are shown in Table 5.4 where the name of each attribute dataset is defined as:

“SdAd_”[name-of-the-alphabet]”-“[M(ono)- D(i)- or T(ri)-peptide].

Table 5.4: Sequence-derived attribute datasets used in our experiment.

#	Attributes Dataset	# of attributes	Origin
1	SdAd_Natural-M	20	Attributes used in solubility classification in [30].
2	SdAd_Natural-D	13	Attributes used in solubility classification in [30].
3	SdAd_Natural-T	24	Attributes used in solubility classification in [30].
4	SdAd_Hydro-M	5	Attributes used in solubility classification in [30].
5	SdAd_Hydro-T	12	Attributes used in solubility classification in [30].
6	SdAd_ConfSimi-M	7	Attributes used in solubility classification in [30].
7	SdAd_ConfSimi-D	20	Attributes used in solubility classification in [30].
8	SdAd_ConfSimi-T	15	Attributes used in solubility classification in [30].
9	SdAd_BlosumSm-M	8	Attributes used in solubility classification in [30].
10	SdAd_BlosumSm-D	25	Attributes used in solubility classification in [30].
11	SdAd_ClustEm14-M	14	Attributes used in solubility classification in [30].
12	SdAd_ClustEm14-D	16	Attributes used in solubility classification in [30].
13	SdAd_ClustEm14-T	22	Attributes used in solubility classification in [30].
14	SdAd_ClustEm17-M	17	Attributes used in solubility classification in [30].
15	SdAd_ClustEm17-D	27	Attributes used in solubility classification in [30].
16	SdAd_ClustEm17-T	42	Attributes used in solubility classification in [30].
17	SdAd_PhysChem-M	7	Attributes used in solubility classification in [30].
18	SdAd_PhysChem-D	21	Attributes used in solubility classification in [30].
19	SdAd_Computed	4	The four most often used computed attributes from Table 2.1: Mw, Ip, SeqLength, GRAVY.
20	SdAd_eSol	22	Attributes used in solubility classification in [73]: freq. of Natural monomers with Mw and Ip
Total # of attributes:		342	

The table contains 18 attribute datasets that represent the frequencies of different alphabets (i.e., datasets from 1 to 18), one attribute dataset with the sequence-computed attributes shown in Table 5.3 (i.e., dataset 19), and one attribute dataset that combines the frequencies of monomers of the Natural alphabet with Mw and Ip (i.e., dataset 20). The attribute datasets from 1 to 19 were used in the solubility classification of [34] while the last attribute dataset was used in the solubility classification of [73]. Note that some frequencies of di- and tri-peptides are missing (e.g. SdAd_Hydro-D), and that the numbers of attributes in the di- and tri-peptide datasets are reduced from all the initial attributes in these datasets to the most relevant ones (e.g. attributes in SdAd_Natural-D are reduced from 400 possible attributes to the 13 the most relevant attributes). More information about the reasons and steps used for these reductions can be found in [34].

5.2.2 Biomedical concept attribute datasets

We have used FACTA (see Section 4.1.2) to extract biomedical concepts associated with the proteins. Although the FACTA web service supports UNIPROT identifiers as its input, we cannot use this feature since FACTA's dictionary is applicable only to human proteins and eSol contains E.coli proteins. Therefore, the names of the eSol proteins serve as the FACTA input. In addition, only five groups of biomedical concepts are used since the gene/protein

group only returns human proteins and it is not included. The FACTA web service takes a SOAP request as an input and returns concepts in the form of an XML file, which has to be parsed to extract the needed information about the concepts. Figure 5.2 illustrates a part of the XML file, where three biomedical concepts from the Enzyme group are shown.

```

- <column category="Enzyme">
- <concept>
  <id>EC:2.7.10.1</id>
  <name>ERK</name>
  <value>1545</value>
</concept>
- <concept>
  <id>EC:3.4.22.56</id>
  <name>caspase-3</name>
  <value>1492</value>
</concept>
- <concept>
  <id>EC:2.7.11.24</id>
  <name>mitogen-activated protein kinase</name>
  <value>1341</value>
</concept>

```

Figure 5.2: A part of an XML file returned by the FACTA web service.

Table 5.5 contains descriptions of the extracted concepts and their quantities for the eSol proteins. The name of each dataset is defined as:

“BcAd_”[name-of-the-biomedical-concept-group]

Table 5.5: Extracted biomedical concept attribute datasets for the eSol database.

Attribute dataset	Examples of concepts	Num. of retrieved concepts
BcAd_Symptom	Starvation, pain, collapse, nausea, headache.	237
BcAd_Disease	Tumor, cancer, liver neoplasms.	1494
BcAd_Drug	Progesterone, taxol.	527
BcAd_Enzyme	Protein kinase, catalase, proteasome.	1100
BcAd_Compund	Serine, estrogen, calcium.	983
		Total: 4341

Each concept is transformed into an attribute where the concept’s name determines the attribute’s name and the concept’s frequency determines the attribute’s value. When a

protein is not associated with a concept, the value of the corresponding attribute is zero. Figure 2.2 illustrates the extraction of biomedical concept (BC) attributes for two proteins.

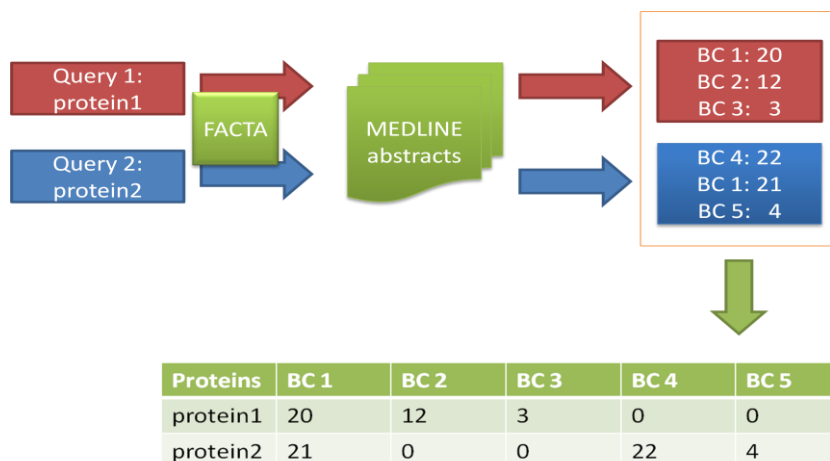


Figure 5.3: An example of determining biomedical concept (BC) attributes for two proteins.

The extraction of biomedical concepts for eSol proteins and the determination of biomedical concept attribute datasets can be described with the following algorithm:

Algorithm 5.1: Extraction of biomedical concepts for eSol.

Extract biomedical concepts for eSol:

P : set of protein names

N : number of proteins

P_i : name of a single protein

Q : FACTA query

C : a set of biomedical concepts with their frequencies

C_j : a single biomedical concept

A_k : a single attribute dataset where k is one of the following (symptom, disease, drug, enzyme, compound)

a : an attribute

a_n : name of the attribute

a_v : value of the attribute

For $i = 1$ to N do:

$Q = P_i$

Query FACTA with Q for frequencies of biomedical concepts
Save returned biomedical concepts and their frequencies into C
For each C_j from C (where $j = 1$ to $|C|$) do:
 Define new attribute a
 $a_n =$ name of the concept C_j
 $a_v =$ frequency of the concept C_j
 Determine type k of C_i
 Save a to the corresponding A_k

5.2.3 Merged attribute datasets

To test the effect of the biomedical concept attributes on solubility prediction, we merged the sequence-derived attribute datasets with the ones FACTA extracted from MEDLINE. Each sequence-derived attribute dataset is combined with each biomedical concept attribute dataset, resulting in 100 (20 sequence-derived attribute datasets multiplied by 5 FACTA-extracted attribute datasets) new merged attribute datasets (MAd) with X number of attributes. For each merged attribute dataset X is calculated as

$$X = X_1 + X_2 \tag{5.1}$$

where X_1 and X_2 represent the number of attributes from the corresponding sequence derived attribute dataset (Table 5.4) and the corresponding biomedical concept attribute dataset (Table 5.5), respectively. Table 5.6 shows the number X of attributes in each of the merged attribute datasets. The numbers in brackets show the X_1 and X_2 values. The name of a single merged datasets is defined as:

“MdAd_”[name-of-the-alphabet]”-“[mono-, di- or tri-peptide]”-“[name-of-the-concept-group],

example: MAd_PhysChem-D_Compound is a merged attribute dataset obtained from merging the datasets SdAd_PhysChem-D and BcAd_Compound.

Table 5.6: Number of attributes in merged attribute datasets.

		BcAd_ Symptom (237)	BcAd_ Disease (1494)	BcAd_ Drug (527)	BcAd_ Enzyme (1100)	BcAd_ Compound (983)
SdAd_Natural-M	(20)	257	1514	547	1120	1003
SdAd_Natural-D	(13)	250	1507	540	1113	996
SdAd_Natural-T	(24)	261	1518	551	1124	1007
SdAd_Hydro-M	(5)	242	1499	532	1105	988
SdAd_Hydro-T	(12)	249	1506	539	1112	995
SdAd_ConfSimi-M	(7)	244	1501	534	1107	990
SdAd_ConfSimi-D	(20)	257	1514	547	1120	1003
SdAd_ConfSimi-T	(15)	252	1509	542	1115	998
SdAd_BlosumSm-M	(8)	245	1502	535	1108	991
SdAd_BlosumSm-D	(25)	262	1519	552	1125	1008
SdAd_ClustEm14-M	(14)	251	1508	541	1114	997
SdAd_ClustEm14-D	(16)	253	1510	543	1116	999
SdAd_ClustEm14-T	(22)	259	1516	549	1122	1005
SdAd_ClustEm17-M	(17)	254	1511	544	1117	1000
SdAd_ClustEm17-D	(27)	264	1521	554	1127	1010
SdAd_ClustEm17-T	(42)	279	1536	569	1142	1025
SdAd_PhysChem-M	(7)	244	1501	534	1107	990
SdAd_PhysChem-D	(21)	258	1515	548	1121	1004
SdAd_Computed	(5)	242	1499	532	1105	988
SdAd_eSol	(22)	259	1516	549	1122	1005

5.3 Feature selection, classification and evaluation

As seen in Figure 5.1, the feature selection and classification steps are illustrated inside the *10-fold cross validation* box. It is important to note that the most common mistake of machine learning applications in the life sciences is to use both training and test sets for feature selection followed by classification with N-fold cross validation as pointed out in [23]. While this procedure ensures that a unique test dataset is used for the classification, the information from this test set has already been used for the inference of the classifier by choosing an optimal subset of attributes for the learning algorithm. Therefore, the feature selection step has to be part of the evaluation process and, thus, included inside the cross validation loop, since the test data must not be used for feature selection methods. Otherwise, estimates of the classifier's performance may be over optimistic. The details of the 10-fold cross validation process with feature selection are illustrated in Figure 5.4.

To evaluate the actual gain in the classifier's performance due to the new attributes, datasets with merged attributes are included in the feature selection step (since the sequence derived attribute datasets already contain only the most relevant attributes). The number of selected attributes in the merged attribute datasets depends on the specific dataset and the specific

classification algorithm used, and it ranges from the total number of protein sequence derived attributes in the dataset to some pre-defined maximum number. The latter is flexible and it is set during the empirical experiments that are described in Section 6.

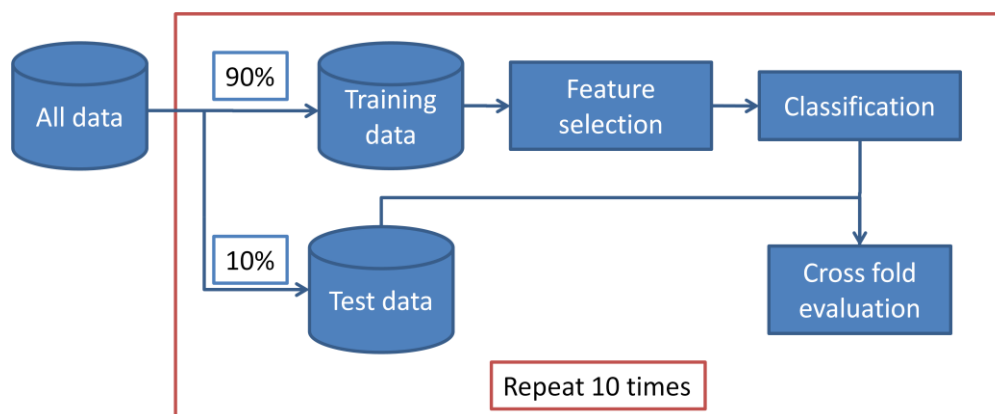


Figure 5.4: 10-fold cross validation with feature selection.

During the classification step, three different classification algorithms are used for creating classifiers. SVM is chosen since it was used in the original work of classifying eSol proteins [73] where authors reported an accuracy of 80%. In addition, as mentioned before, SVM has been proven to be the most accurate classification method for protein solubility classification [29-31,34] and, therefore, it is also the most promising for our experiments. Since 100 new attribute datasets were introduced, we decided to also use the Naïve Bayes and the Decision trees classification methods, to analyse how the new attributes affect these classifiers' behaviours. The former has been used due its simplicity and fast performance, while the latter has been used due to the capability of interpreting its results.

The evaluation step is needed to assess the quality of the new attributes and confirm our hypothesis described in Section 1.2.1. In each fold the classifier is evaluated with the following metrics: Accuracy, Sensitivity, Precision, MCC and AUC. At the end of the cross validation process the ten values for each metric are averaged to yield an overall evaluation result and the results of each classifier are compared with each other. Classifiers are directly compared with the average Accuracy, MCC and AUC values. In addition, Sensitivity and Precision for both classes are analysed for the classifiers with the best performances with the previous three metrics. We use the Wilcoxon signed-ranks test to compare pairs of classifiers, as described in Section 2.6.2.

The general classification scheme can be described by the following algorithm:

Algorithm 5.2: The general classification scheme for building and evaluating classifiers.

Build and evaluate classifiers for solubility prediction:

P : set of eSol proteins

N : number of eSol proteins

P_{train} : subset of P that is the training set

P_{test} : subset of P that is the test set

C : a set of machine learning techniques (Naïve Bayes, SMO, J48)

C_j : a single machine learning technique

A_{sd} : a set of sequence derived attribute datasets

A_{sdk} : a single sequence derived attribute dataset

A_m : a set of merged(combined) attribute datasets

A_{mk} : a single merged(combined) attribute dataset

S : a set of feature selection methods (InfoGain, ReliefF)

S_l : a single feature selection method

a_{min} : a minimum number of selected attributes

a_{max} : a maximum number of selected attributes

R : a set of built classifiers (results)

For $i = 1$ to 10 do:

$P_{train} =$ randomly select $\frac{9}{10}N$ proteins from P

$P_{test} = P - P_{train}$

For each C_j from C (where $j = 1$ to $|C|$) do:

For each A_{sdk} from A_{sd} (where $k = 1$ to $|A_{sd}|$) do:

Build classifier c with C_j on P_{train} with A_{sdk}

Save c into R

For each A_{mk} from A_m (where $k = 1$ to size of $|A_m|$) do:

For every S_l from S (where $l = 1$ to $|S|$) do:

For every t (where $t = a_{min}$ to a_{max}) do:

Select t attributes from A_{mk} with S_l

Build classifier c with C_j on P_{train} with t attributes

Save c into R

For each c_i from R where $i = 1$ to $|R|$ do:

Evaluate c_i on P_{test}

5.4 Framework for supervised machine learning

In this section we briefly introduce the Weka framework [49], which is the basis for the classification scheme developed in this thesis. The first version of Weka was developed in 1993 at the Waikato University in New Zealand and it has been used widely in academia for research purposes, since it can easily be adapted to new problems and machine learning techniques, and offers effective data visualisation. In addition, it is based on open source principles.

In the scope of this thesis two Weka components are used: the Explorer tool (Figure 5.5) and the Weka source code written in the Java. The former is used to manipulate the data and to create the learning model prototypes before performing the actual in depth implementation of the classification scheme in Java. This is because although Explorer offers a convenient way of working with machine learning algorithms, using the Weka source code directly yields better results. In addition, when working with large datasets (such as ours) the experiments usually take a long time and, if something goes wrong, it is possible to get partial results when working with the source code. Therefore, we have upgraded Weka's source code with additional components for implementing our classification scheme and for managing our protein inputs to build the classifiers.

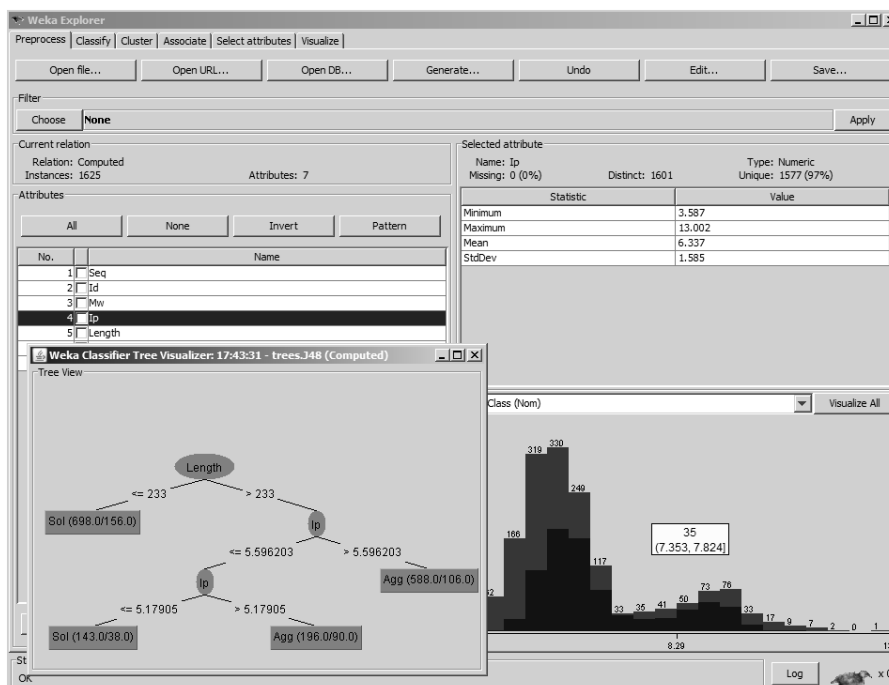


Figure 5.5: Visualisation options in Weka.

The input file for Weka can be a comma separated values (CSV) file or an attribute relation file format (ARFF) file. Both files are simple ASCII text files that contain pairs of instances and their class labels. In a CSV file each instance is presented with a new row where a single instance's attribute values are separated by commas. The first row represents the names of the attributes where the last attribute usually represents the class attribute. Figure 5.6 illustrates the CSV file format for m instances and n attributes where $inst_j_v_i$ represents the i -th value of the j -th instance.

```
att_name_1, att_name_2, att_name_3, ..., att_name_i, ..., att_name_n
inst_1_v_1, inst_1_v_2, inst_1_v_3, ..., inst_1_v_i, ..., inst_1_v_n
inst_2_v_1, inst_2_v_2, inst_2_v_3, ..., inst_2_v_i, ..., inst_2_v_n
.....
inst_j_v_1, inst_j_v_2, inst_j_v_3, ..., inst_j_v_i, ..., inst_j_v_n
.....
inst_m_v_1, inst_m_v_2, inst_m_v_3, ..., inst_m_v_i, ..., inst_m_v_n
```

Figure 5.6: A CSV input for Weka.

ARFF files have two distinct sections. The first section is the header information, which is followed by the data information. The former contains the name of the relation and a list of the attributes with their types (e.g. numeric), while the latter contains the actual data. Figure 5.7 illustrates an ARFF file with m instances, each with n attributes.

```
% Comments are possible in ARRF
%
@RELATION name_of_relation

@ATTRIBUTE att_name_1 att_type_1
@ATTRIBUTE att_name_2 att_type_2
@ATTRIBUTE att_name_3 att_type_3
...
@ATTRIBUTE att_name_i att_type_i
...
@ATTRIBUTE att_name_n att_type_n

@DATA
inst_1_v_1, inst_1_v_2, inst_1_v_3, ..., inst_1_v_i, ..., inst_1_v_n
inst_2_v_1, inst_2_v_2, inst_2_v_3, ..., inst_2_v_i, ..., inst_2_v_n
.....
inst_j_v_1, inst_j_v_2, inst_j_v_3, ..., inst_j_v_i, ..., inst_j_v_n
.....
inst_m_v_1, inst_m_v_2, inst_m_v_3, ..., inst_m_v_i, ..., inst_m_v_n
```

Figure 5.7: An ARRF input for Weka.

Weka offers support for all the three machine learning techniques used in this thesis. Note however, that (a) the algorithm used for building C4.5 decision trees is referred to as J48 [51], and (b) the Sequential Minimal Optimization (SMO) algorithm [76] is used for building SVM classifiers. SMO breaks the quadratic problem of building SVM into a series

of smallest possible sub-problems which are then solved analytically. As a result, SMO builds classifiers much faster and is less memory intensive than other popular implementations of SVM (such as LibSVM [28]). Since the goal of this thesis is to evaluate several different classifiers on different attribute datasets, fast performance is one of the main criteria in selecting machine learning techniques.

We use parameter search techniques (see Section 2.5.2) on the sequence derived attribute datasets for finding optimal parameter values for all three machine learning algorithms and use also these parameter values on the merged attribute datasets. This way the classifiers built with different attribute datasets can be compared.

6

Results

6.1 Introduction

This section provides the results collected during the experiments. The results are divided into three different sections according to the machine learning technique being used. Each section has two sub-sections, one for each type of attribute datasets: the sequence derived attribute datasets and the merged attribute datasets.

6.2 Naïve Bayes

Naïve Bayes is the least time consuming classification method amongst all three used in this thesis and, as such, it is used as the preliminary assessment method for the quality of biomedical concept attribute datasets. We denote the Naïve Bayes classifier built on sequence derived attributes by *NBs* and the Naïve Bayes classifier built on merged datasets by *NBm*.

6.2.1 Sequence derived attribute datasets

First, Naïve Bayes is used on all of the 20 sequence derived attribute datasets. Table 6.1 shows the accuracy (Acc), MCC, AUC, Sensitivity of Soluble/Insoluble (Si Sol/Si Insol) and Precision of Soluble/Insoluble (Pr Sol/Pr Insol) results obtained for each dataset. As described in Section 2.6, the closer the results are to value 1, the better they are. Note that none of the attribute datasets results in the highest values in all the metrics and it is hard to compare the classifiers directly using all the metrics at once. Since our datasets are unbalanced (they contain around 10% more soluble proteins), the most important metrics for our experiments are MCC and AUC. As mentioned in Section 2.6, they give a good overall indication of the classifiers performance.

The table shows that Naïve Bayes performed the best on **SdAd_eSol** with the highest Acc, MCC, AUC, Si Ins, Pr Sol and Pr Ins while **SdAd_ClustEm14-T** (and **SdAd_ClustEm14-T**) resulted in higher Si Sol scores. A closer look at the confusion matrixes for these databases (Figure 5.7) reveals that the latter two attribute datasets offer better classification of soluble proteins at the cost of more misclassified insoluble proteins compared to SdAd_eSol. However, SdAd_eSol offers the most balanced classification which also results in highest Acc, MCC and AUC scores. Note that this attribute dataset offers the lowest number of misclassified insoluble proteins. This is important, since misclassified insoluble proteins waste researcher's time and resources on studying wrong proteins.

Table 6.1: Results for Naïve Bayes built on sequence derived attribute datasets.

Attributes Dataset	Acc	MCC	AUC	Si Sol	Si Ins	Pr Sol	Pr Ins
SdAd_Natural-M	0.739	0.486	0.821	0.681	0.802	0.787	0.700
SdAd_Natural-D	0.682	0.364	0.738	0.675	0.689	0.701	0.663
SdAd_Natural-T	0.487	-0.008	0.463	0.255	0.738	0.512	0.479
SdAd_Hydro-M	0.652	0.308	0.705	0.606	0.701	0.686	0.623
SdAd_Hydro-T	0.660	0.323	0.717	0.638	0.684	0.685	0.637
SdAd_ConfSimi-M	0.658	0.323	0.714	0.599	0.721	0.698	0.625
SdAd_ConfSimi-D	0.731	0.470	0.790	0.676	0.790	0.777	0.694
SdAd_ConfSimi-T	0.676	0.368	0.739	0.574	0.785	0.742	0.631
SdAd_BlosumSm-M	0.714	0.432	0.769	0.687	0.744	0.743	0.688
SdAd_BlosumSm-D	0.671	0.345	0.723	0.647	0.698	0.698	0.647
SdAd_ClustEm14-M	0.710	0.426	0.782	0.662	0.762	0.750	0.677
SdAd_ClustEm14-D	0.723	0.445	0.784	0.734	0.711	0.733	0.713
SdAd_ClustEm14-T	0.621	0.241	0.663	0.758	0.473	0.608	0.645
SdAd_ClustEm17-M	0.748	0.508	0.830	0.677	0.825	0.806	0.703
SdAd_ClustEm17-D	0.702	0.402	0.756	0.731	0.670	0.705	0.698
SdAd_ClustEm17-T	0.632	0.262	0.642	0.720	0.537	0.626	0.640
SdAd_PhysChem-M	0.729	0.471	0.798	0.651	0.813	0.790	0.684
SdAd_PhysChem-D	0.739	0.482	0.809	0.702	0.779	0.774	0.708
SdAd_Computed	0.736	0.475	0.806	0.705	0.770	0.767	0.707
SdAd_eSol	0.782	0.570	0.855	0.733	0.834	0.826	0.743
Best	0.782	0.570	0.855	0.758	0.834	0.826	0.743

Actual/Classified	Soluble	Insoluble
Soluble	619	224
Insoluble	226	556

a) SdAd_ClustEm14-D

Actual/Classified	Soluble	Insoluble
Soluble	639	204
Insoluble	412	370

b) SdAd_ClustEm14-T

Actual/Classified	Soluble	Insoluble
Soluble	618	225
Insoluble	130	652

c) SdAd_eSol

Figure 6.1: Confusion matrixes for the best three classifiers on sequence derived attribute datasets with Naive Bayes.

We can also notice a very low (random) performance on the classifier built on **SdAd_Natural-T**. This attribute dataset is one of the attribute datasets used in solubility classification on the SOLpro database (see Section 3.3.1). Since it achieves such a low performance on eSol, it is (with its merged datasets) removed from our future experiments involving the Naïve Bayes method.

6.2.2 Merged attribute datasets

The goal of these experiments is to measure the performance of the Naïve Bayes classifier on each of the sequence derived attribute datasets when merged with disease, drug, symptom, enzyme and compound attributes (we denote these classifiers by $NBm_{disease}$, NBm_{drug} , $NBm_{symptom}$, NBm_{enzyme} and $NBm_{compound}$, respectively). To assess the classifiers' performances, we use five different measurements (one for each biomedical concept group) to compare each of these classifiers with the corresponding NBs. The datasets used in each measurement are represented in Table 6.2, where $[X]$ is replaced with one of the biomedical concept groups. For instance, MAd_Natural-M_Disease contains biomedical concept attributes from the disease group. Note that, although MAd_Natural-T $_{[X]}$ is not used with Naïve Bayes due to the poor performance obtained in previous experiments, we include this database in the table since it is used later with other classification algorithms.

As described in Section 2, two feature selection methods are used for identifying the most relevant attributes from the merged datasets before the classification step. Therefore, each of the five measurements is performed twice, once for each feature selection method.

Table 6.2: Merged attribute datasets used in one performance measurement.

Db #	Name	Db #	Name
1	MAd_Natural-M $_{[X]}$	11	MAd_ClustEm14-M $_{[X]}$
2	MAd_Natural-D $_{[X]}$	12	MAd_ClustEm14-D $_{[X]}$
3	MAd_Natural-T $_{[X]}$	13	MAd_ClustEm14-T $_{[X]}$
4	MAd_Hydro-M $_{[X]}$	14	MAd_ClustEm17-M $_{[X]}$
5	MAd_Hydro-T $_{[X]}$	15	MAd_ClustEm17-D $_{[X]}$
6	MAd_ConfSimi-M $_{[X]}$	16	MAd_ClustEm17-T $_{[X]}$
7	MAd_ConfSimi-D $_{[X]}$	17	MAd_PhysChem-M $_{[X]}$
8	MAd_ConfSimi-T $_{[X]}$	18	MAd_PhysChem-D $_{[X]}$
9	MAd_BlosumSm-M $_{[X]}$	19	MAd_Computed $_{[X]}$
10	MAd_BlosumSm-D $_{[X]}$	20	MAd_eSol $_{[X]}$

We first use Information Gain to select attributes. Attributes are selected by starting with the number of attributes in the corresponding sequence derived attribute dataset, and increasing the number of attributes by 1, up to a maximum of 200. For instance, since SdAd_Natural-M contains 20 attributes and MAd_Natural-M_Symptom contains 257 attributes (see Table 5.4 and Table 5.6), Information Gain is used to select attributes ranging from 21 to 200 for MAd_Natural-M_Symptom. Ideally, the maximum number of selected attributes would be the maximum number of attributes in the merged dataset (257 for our example), so that every possible number of selected attributes would be covered. However, this would be too computationally intensive (some merged attribute datasets contain more than 1500 attributes) for this thesis. In addition, it has been proven that low numbers of attributes usually contribute to better results in protein solubility classification compared to classifiers with high number of attributes [34]. The latter also shows to be true for the database used in this thesis after the analysis of results for the classifiers built on merged datasets. To illustrate this behaviour, Figure 6.2 shows the MCC values for five merged datasets: MAd_Computed_Compound (mcc_cmp), MAd_Computed_Disease (mcc_dis), MAd_Computed_Drug (mcc_drg), MAd_Computed_Enzyme (mcc_enz) and MAd_Computed_Symptom (mcc_sym) measured for different number of selected attributes with Information Gain. The x axis shows the number of selected attributes in each classification step while the y axis shows MCC values. As it can be noticed, the MCC values slowly degrade as the number of attributes selected increases over a certain value. Since our experiments showed that this graph represents the typical behaviour also for the AUC and Acc performance measurements on all other built classifiers (not shown), we decided not to conduct experiments for more than 200 selected attributes.

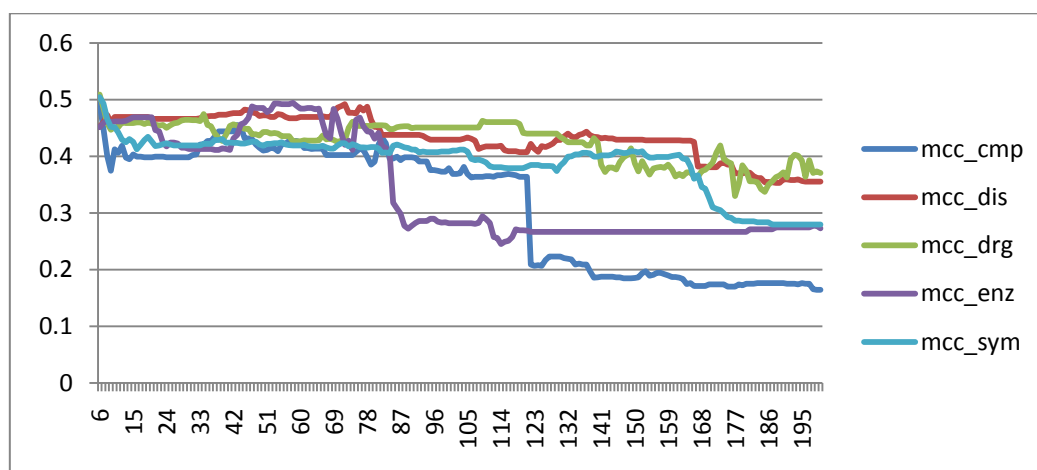


Figure 6.2: MCC values for five merged datasets and different number of attributes selected.

The results of applying Naïve Bayes as the classification algorithm and Information Gain as the feature selection method are grouped into two tables. Table 6.3 shows the Acc, MCC and

AUC performance measurements for the best $NBm_{disease}$ and NBm_{enzyme} classifiers, while Table 6.4 shows the results for the best NBm_{drug} , $NBm_{compound}$ and $NBm_{symptom}$ classifiers. For both tables the numbers in the *Db* column correspond to the attribute datasets in Table 6.2. The values in brackets represent the increase or decrease in the classifier's performance, where positive/negative values show that NBm performed better/worse than the corresponding NBs , and value 0.000 indicates there has been no change in the performance. The values in bold mark the results for the dataset which contains the sequence derived attributes that resulted in the best classifier in the previous experiment. The last two rows in each section show the number for positive/negative/neutral differences (P/N/Z), and the confidence values α for the Wilcoxon test (see Section 2.6.2). For the latter, $\alpha < 0.05$ indicates statistically significant differences in the classifiers' performances, while other α values show that any changes in the classifiers' performances are not statistically significant.

Table 6.3: Performance measurements for $NBm_{disease}$ and NBm_{enzyme} with Information Gain.

$NBm_{disease}$									
Db	Acc	MCC	AUC	Db	Acc	MCC	AUC		
1	0.740(+0.001)	0.488(+0.002)	0.821(+0.001)	12	0.722(-0.001)	0.444(-0.001)	0.784(0.000)		
2	0.682(0.000)	0.364(0.000)	0.738(0.000)	13	0.621(0.000)	0.241(0.000)	0.663(0.000)		
4	0.652(0.000)	0.308(0.000)	0.706(+0.001)	14	0.749(+0.001)	0.509(+0.001)	0.830(0.000)		
5	0.661(+0.001)	0.324(+0.001)	0.718(+0.001)	15	0.702(0.000)	0.402(0.000)	0.756(0.000)		
6	0.658(0.000)	0.323(0.000)	0.714(0.000)	16	0.633(+0.001)	0.263(+0.001)	0.642(0.000)		
7	0.731(0.000)	0.470(0.000)	0.790(0.000)	17	0.729(0.000)	0.472(+0.001)	0.797(-0.001)		
8	0.676(0.000)	0.367(-0.001)	0.738(-0.001)	18	0.739(0.000)	0.482(0.000)	0.809(0.000)		
9	0.715(+0.001)	0.433(+0.001)	0.769(0.000)	19	0.736(0.000)	0.475(0.000)	0.806(0.000)		
10	0.671(0.000)	0.346(+0.001)	0.723(0.000)	20	0.782(0.000)	0.571(+0.001)	0.855(0.000)		
11	0.711(+0.001)	0.429(+0.003)	0.782(0.000)						
P		Acc: 6	MCC: 9	AUC: 3	N		Acc: 1 MCC: 2 AUC: 2	Z	Acc: 12 MCC: 8 AUC: 14
α Acc: 0.059 MCC: 0.029 AUC: 0.655									
NBm_{enzyme}									
Db	Acc	MCC	AUC	Db	Acc	MCC	AUC		
1	0.739(0.000)	0.486(0.000)	0.820(0.000)	12	0.723(0.000)	0.445(0.000)	0.784(0.000)		
2	0.682(0.000)	0.364(0.000)	0.738(0.000)	13	0.621(0.000)	0.241(0.000)	0.663(0.000)		
4	0.652(0.000)	0.308(0.000)	0.705(0.000)	14	0.748(0.000)	0.508(0.000)	0.830(0.000)		
5	0.660(0.000)	0.323(0.000)	0.717(0.000)	15	0.702(0.000)	0.402(0.000)	0.756(0.000)		
6	0.658(0.000)	0.323(0.000)	0.714(0.000)	16	0.632(0.000)	0.262(0.000)	0.642(0.000)		
7	0.731(0.000)	0.470(0.000)	0.790(0.000)	17	0.729(0.000)	0.471(0.000)	0.798(0.000)		
8	0.676(0.000)	0.368(0.000)	0.739(0.000)	18	0.739(0.000)	0.482(0.000)	0.809(0.000)		
9	0.714(0.000)	0.432(0.000)	0.769(0.000)	19	0.736(0.000)	0.475(0.000)	0.806(0.000)		
10	0.671(0.000)	0.345(0.000)	0.723(0.000)	20	0.781(-0.001)	0.569(-0.001)	0.856(+0.001)		
11	0.710(0.000)	0.426(0.000)	0.782(0.000)						
P		Acc: 0	MCC: 0	AUC: 1	N		Acc: 1 MCC: 1 AUC: 0	Z	Acc: 18 MCC: 18 AUC: 18
α Acc: 0.317 MCC: 0.317 AUC: 0.317									

According to the α values in the upper part of Table 6.3, $NBm_{disease}$ performed significantly better in MCC compared to the corresponding NBs , while, there were no significant differences measured with the other two metrics. The lower part of the table shows results for NBm_{enzyme} , for which no significant differences are found since all three α values are above 0.05.

Table 6.4 shows the results for the NBm_{drug} , $NBm_{compound}$ and $NBm_{symptom}$ classifiers. Common to these classifiers is that they all outperform NBs when measuring their performance with the AUC metric. In addition, NBm_{drug} and $NBm_{compound}$ perform better on the majority of datasets also when measured with Acc and MCC. However, the Wilcoxon signed-ranks test shows that these differences are not statistically significant.

Table 6.4: Performance measurements for NBm_{drug} , $NBm_{compound}$ and $NBm_{symptom}$ with Information Gain.

NBm_{drug}							
Db	Acc	MCC	AUC	Db	Acc	MCC	AUC
1	0.733(-0.006)	0.471(-0.015)	0.823(+0.003)	12	0.729(+0.006)	0.456(+0.011)	0.788(+0.004)
2	0.684(+0.002)	0.369(+0.005)	0.743(+0.005)	13	0.613(-0.008)	0.223(-0.018)	0.671(+0.008)
4	0.642(-0.010)	0.293(-0.015)	0.712(+0.007)	14	0.751(+0.003)	0.511(+0.003)	0.832(+0.002)
5	0.664(+0.004)	0.330(+0.007)	0.723(+0.006)	15	0.708(+0.006)	0.415(+0.013)	0.760(+0.004)
6	0.655(-0.003)	0.314(-0.009)	0.718(+0.004)	16	0.635(+0.003)	0.269(+0.007)	0.653(+0.011)
7	0.735(+0.004)	0.477(+0.007)	0.793(+0.003)	17	0.735(+0.006)	0.478(+0.007)	0.801(+0.003)
8	0.684(+0.008)	0.383(+0.015)	0.744(+0.005)	18	0.737(-0.002)	0.479(-0.003)	0.811(+0.002)
9	0.719(+0.005)	0.441(+0.009)	0.773(+0.004)	19	0.736(0.000)	0.475(0.000)	0.818(+0.012)
10	0.679(+0.008)	0.360(+0.015)	0.732(+0.009)	20	0.780(-0.002)	0.566(-0.004)	0.857(+0.002)
11	0.706(-0.004)	0.418(-0.008)	0.785(+0.003)				
P Acc: 11		MCC: 11 AUC: 19		N Acc: 7		MCC: 7 AUC: 0	
				Z Acc: 1		MCC: 1 AUC: 0	
α Acc: 0.337 MCC: 0.585 AUC: 0.000							
$NBm_{compound}$							
Db	Acc	MCC	AUC	Db	Acc	MCC	AUC
1	0.735(-0.004)	0.477(-0.009)	0.821(+0.001)	12	0.726(+0.003)	0.450(+0.005)	0.786(+0.002)
2	0.680(-0.002)	0.360(-0.004)	0.742(+0.004)	13	0.615(-0.006)	0.227(-0.014)	0.668(+0.005)
4	0.654(+0.002)	0.311(+0.003)	0.706(+0.001)	14	0.750(+0.002)	0.508(0.000)	0.831(+0.001)
5	0.663(+0.003)	0.329(+0.006)	0.720(+0.003)	15	0.706(+0.004)	0.410(+0.008)	0.759(+0.003)
6	0.654(-0.004)	0.315(-0.008)	0.715(+0.001)	16	0.634(+0.002)	0.268(+0.006)	0.650(+0.008)
7	0.735(+0.004)	0.477(+0.007)	0.792(+0.002)	17	0.737(+0.008)	0.483(+0.012)	0.798(0.000)
8	0.674(-0.002)	0.363(-0.005)	0.742(+0.003)	18	0.735(-0.004)	0.475(-0.007)	0.811(+0.002)
9	0.718(+0.004)	0.438(+0.006)	0.771(+0.002)	19	0.742(+0.006)	0.487(+0.012)	0.811(+0.005)
10	0.679(+0.008)	0.360(+0.015)	0.728(+0.005)	20	0.782(0.000)	0.571(+0.001)	0.856(+0.001)
11	0.708(-0.002)	0.420(-0.006)	0.783(+0.001)				
P Acc: 11		MCC: 11 AUC: 18		N Acc: 7		MCC: 7 AUC: 0	
				Z Acc: 1		MCC: 1 AUC: 1	
α Acc: 0.272 MCC: 0.472 AUC: 0.000							
$NBm_{symptom}$							
Db	Acc	MCC	AUC	Db	Acc	MCC	AUC
1	0.736(-0.003)	0.480(-0.006)	0.821(+0.001)	12	0.723(0.000)	0.445(0.000)	0.784(0.000)
2	0.681(-0.001)	0.363(-0.001)	0.739(+0.001)	13	0.621(0.000)	0.240(-0.001)	0.663(0.000)
4	0.650(-0.002)	0.306(-0.002)	0.706(+0.001)	14	0.750(+0.002)	0.511(+0.003)	0.831(+0.001)
5	0.658(-0.002)	0.319(-0.004)	0.719(+0.002)	15	0.705(+0.003)	0.409(+0.007)	0.757(+0.001)
6	0.651(-0.007)	0.309(-0.014)	0.715(+0.001)	16	0.636(+0.004)	0.269(+0.007)	0.645(+0.003)
7	0.732(+0.001)	0.472(+0.002)	0.791(+0.001)	17	0.725(-0.004)	0.459(-0.012)	0.798(0.000)
8	0.672(-0.004)	0.359(-0.009)	0.738(-0.001)	18	0.739(0.000)	0.482(0.000)	0.809(0.000)
9	0.714(0.000)	0.431(-0.001)	0.770(+0.001)	19	0.736(0.000)	0.475(0.000)	0.808(+0.002)
10	0.662(-0.009)	0.326(-0.019)	0.724(+0.001)	20	0.782(0.000)	0.570(0.000)	0.855(0.000)
11	0.714(+0.004)	0.433(+0.007)	0.783(+0.001)				
P Acc: 5		MCC: 5 AUC: 13		N Acc: 8		MCC: 10 AUC: 1	
				Z Acc: 6		MCC: 4 AUC: 5	
α Acc: 0.309 MCC: 0.267 AUC: 0.02							

The second part of our experiments used Naïve Bayes with ReliefF as the feature selection method. However, the results (not shown) yield no significant improvements when comparing the performances of the classifiers obtained with Information Gain. As described in Section 2.3.1, ReliefF considers more than one attribute at the time when selecting the relevant attributes. As a result, this makes it computationally very intensive. In addition, both ReliefF and Information Gain are filter methods, which means that the same attributes are selected no matter what method is used for the classification. For these reasons, we decided not to use ReliefF in the rest of the experiments.

6.3 Decision trees

As mentioned in Section 2.4.3, decision trees are popular due to the capability of interpreting its results. As mentioned in Section 5.4, we used *J48* in Weka, and denote the classifier built on sequence derived attributes by *J48s* and the classifier built on merged datasets by *J48m*. The latter is furthermore marked with *J48m_{disease}*, *J48m_{drug}*, *J48m_{symptom}*, *J48m_{enzyme}* and *J48m_{compound}*, representing the classifiers with added Disease, Drug, Symptom, Enzyme and Compound biomedical concept attributes, respectively.

6.3.1 Sequence derived attribute datasets

Table 6.5 shows the results of *J48s* built on 20 different datasets, where (as before) the values in bold show the best performance score for each metric. Note that **SdAd_Computed** results in the highest Acc, MCC, AUC and Pr Ins scores, **SdAd_PhysChem-M** results in the highest Si Ins and Pr Sol scores, and **SdAd_Natural-T** results in the highest Si Sol score. While the first two datasets offer competitive scores in all metrics, the latter dataset performs very badly when measured with other metrics.

Table 6.5: Results for *J48s*.

Attributes Dataset	Acc	MCC	AUC	Si Sol	Si Ins	Pr Sol	Pr Ins
SdAd_Natural-M	0.701	0.406	0.720	0.661	0.744	0.736	0.671
SdAd_Natural-D	0.624	0.249	0.628	0.610	0.639	0.646	0.603
SdAd_Natural-T	0.546	0.101	0.528	0.921	0.143	0.537	0.626
SdAd_Hydro-M	0.654	0.313	0.675	0.604	0.707	0.690	0.623
SdAd_Hydro-T	0.641	0.285	0.656	0.600	0.684	0.672	0.614
SdAd_ConfSimi-M	0.652	0.306	0.680	0.628	0.678	0.677	0.628
SdAd_ConfSimi-D	0.657	0.315	0.649	0.645	0.670	0.678	0.637
SdAd_ConfSimi-T	0.671	0.352	0.712	0.603	0.746	0.719	0.635
SdAd_BlosumSm-M	0.660	0.323	0.668	0.624	0.698	0.690	0.633
SdAd_BlosumSm-D	0.624	0.247	0.662	0.637	0.610	0.638	0.609
SdAd_ClustEm14-M	0.706	0.414	0.710	0.681	0.733	0.733	0.681
SdAd_ClustEm14-D	0.660	0.319	0.671	0.662	0.657	0.676	0.643
SdAd_ClustEm14-T	0.593	0.187	0.627	0.590	0.597	0.612	0.574
SdAd_ClustEm17-M	0.726	0.454	0.734	0.713	0.740	0.748	0.705
SdAd_ClustEm17-D	0.633	0.265	0.633	0.630	0.636	0.651	0.614
SdAd_ClustEm17-T	0.616	0.229	0.624	0.683	0.543	0.617	0.614
SdAd_PhysChem-M	0.717	0.442	0.760	0.656	0.783	0.765	0.678
SdAd_PhysChem-D	0.694	0.387	0.693	0.698	0.689	0.708	0.679
SdAd_Computed	0.748	0.497	0.798	0.828	0.662	0.726	0.781
SdAd_eSol	0.734	0.467	0.718	0.753	0.714	0.739	0.728
Best	0.748	0.497	0.798	0.921	0.783	0.765	0.781

6.3.2 Merged attribute datasets

This section contains results of each *J48m* classifier compared with *J48s*. Compared to the experiments performed with use of Naïve Bayes, here we use all 20 merged datasets in each measurement since the *J48s* classifier built on SdAd_Natural-T performed better than the one built with Naïve Bayes.

Although the *J48* algorithm uses its own mechanisms for selecting the most relevant attributes when building the tree, we can sometimes get slightly better results if we perform attribute selection as a pre-processing step. This can be the case particularly on smaller datasets, where the divide-and-conquer approach of decision trees results in rapidly decreasing amounts of data from which to determine attribute relevancy. Therefore, Information Gain is applied to select relevant attributes. Again, the results are grouped into two tables. Table 6.6 shows the results for *J48m_{drug}* and *J48m_{enzyme}*. Both classifiers performed worse than *J48s* on the majority of datasets in all three metrics. However, according to the α values, these differences are not statistically significant.

Table 6.6: Performance measurements for $J48m_{drug}$ and $J48m_{enzyme}$ with Information Gain.

$J48m_{drug}$											
Db	Acc	MCC	AUC	Db	Acc	MCC	AUC				
1	0.668(-0.033)	0.337(-0.069)	0.662(-0.058)	11	0.679(-0.027)	0.364(-0.050)	0.687(-0.023)				
2	0.644(+0.020)	0.289(+0.040)	0.652(+0.024)	12	0.649(-0.011)	0.297(-0.022)	0.668(-0.003)				
3	0.540(-0.006)	0.108(+0.007)	0.551(+0.023)	13	0.622(+0.029)	0.243(+0.056)	0.638(+0.011)				
4	0.639(-0.015)	0.289(-0.024)	0.668(-0.007)	14	0.673(-0.053)	0.352(-0.102)	0.675(-0.059)				
5	0.619(-0.022)	0.237(-0.048)	0.648(-0.008)	15	0.644(+0.011)	0.286(+0.021)	0.649(+0.016)				
6	0.614(-0.038)	0.225(-0.081)	0.642(-0.038)	16	0.638(+0.022)	0.273(+0.044)	0.638(+0.014)				
7	0.678(+0.021)	0.358(+0.043)	0.664(+0.015)	17	0.682(-0.035)	0.370(-0.072)	0.713(-0.047)				
8	0.638(-0.033)	0.278(-0.074)	0.670(-0.042)	18	0.682(-0.012)	0.364(-0.023)	0.679(-0.014)				
9	0.657(-0.003)	0.312(-0.011)	0.649(-0.019)	19	0.754(+0.006)	0.509(+0.012)	0.789(-0.009)				
10	0.646(+0.022)	0.289(+0.042)	0.670(+0.008)	20	0.730(-0.004)	0.465(-0.002)	0.744(+0.026)				
P Acc: 7		MCC: 8	AUC: 8	N Acc: 13		MCC: 12	AUC: 12	Z Acc: 0		MCC: 0	AUC: 0
α Acc: 0.156 MCC: 0.167 AUC: 0.304											
$J48m_{enzyme}$											
Db	Acc	MCC	AUC	Db	Acc	MCC	AUC				
1	0.689(-0.012)	0.378(-0.028)	0.696(-0.024)	11	0.665(-0.041)	0.334(-0.080)	0.684(-0.026)				
2	0.628(+0.004)	0.256(+0.007)	0.649(+0.021)	12	0.652(-0.008)	0.306(-0.013)	0.669(-0.002)				
3	0.550(+0.004)	0.093(-0.008)	0.578(+0.050)	13	0.618(+0.025)	0.234(+0.047)	0.631(+0.004)				
4	0.638(-0.016)	0.293(-0.020)	0.661(-0.014)	14	0.670(-0.056)	0.348(-0.106)	0.680(-0.054)				
5	0.626(-0.015)	0.262(-0.023)	0.643(-0.013)	15	0.635(+0.002)	0.270(+0.005)	0.651(+0.018)				
6	0.628(-0.024)	0.255(-0.051)	0.649(-0.031)	16	0.639(+0.023)	0.275(+0.046)	0.638(+0.014)				
7	0.673(+0.016)	0.349(+0.034)	0.672(+0.023)	17	0.689(-0.028)	0.381(-0.061)	0.721(-0.039)				
8	0.617(-0.054)	0.239(-0.113)	0.648(-0.064)	18	0.676(-0.018)	0.352(-0.035)	0.687(-0.006)				
9	0.659(-0.001)	0.319(-0.004)	0.651(-0.017)	19	0.747(-0.001)	0.494(-0.003)	0.797(-0.001)				
10	0.644(+0.020)	0.285(+0.038)	0.666(+0.004)	20	0.713(-0.021)	0.430(-0.037)	0.708(-0.010)				
P Acc: 7		MCC: 6	AUC: 7	N Acc: 13		MCC: 14	AUC: 13	Z Acc: 0		MCC: 0	AUC: 0
α Acc: 0.121 MCC: 0.086 AUC: 0.185											

Table 6.7 shows the results for $J48m_{disease}$, $J48m_{compound}$ and $J48m_{symptom}$. In general, these classifiers also performed worse when compared to the corresponding $J48s$ but the differences in Table 6.7 are statistically significant. Specifically, the α values for $J48m_{disease}$ and $J48m_{compound}$ indicate that these two classifiers result in significantly lower Acc and MCC values while $J48m_{symptom}$ performs significantly worse when measuring performance with the Acc metric. Note that, although adding biomedical concept attributes results in lower performance scores on the majority of datasets, $J48m_{compound}$ and $J48m_{symptom}$ perform better on the most successful sequence derived attribute dataset. $J48m_{symptom}$ is also the best decision tree classifier overall.

Table 6.7: Performance measurements for $J48m_{disease}$, $J48m_{compound}$ and $J48m_{symptom}$ with Information Gain.

J48m_{disease}								
Db	Acc	MCC	AUC	Db	Acc	MCC	AUC	
1	0.676(-0.025)	0.356(-0.050)	0.688(-0.032)	11	0.668(-0.038)	0.339(-0.075)	0.686(-0.024)	
2	0.630(+0.006)	0.264(+0.015)	0.650(+0.022)	12	0.646(-0.014)	0.291(-0.028)	0.673(+0.002)	
4	0.521(-0.025)	0.019(-0.082)	0.507(-0.021)	13	0.601(+0.008)	0.200(+0.013)	0.615(-0.012)	
5	0.615(-0.039)	0.240(-0.073)	0.640(-0.035)	14	0.666(-0.060)	0.345(-0.109)	0.672(-0.062)	
6	0.623(-0.018)	0.258(-0.027)	0.646(-0.010)	15	0.625(-0.008)	0.247(-0.018)	0.626(-0.007)	
7	0.630(-0.022)	0.256(-0.050)	0.655(-0.025)	16	0.633(+0.017)	0.264(+0.035)	0.623(-0.001)	
8	0.677(+0.020)	0.358(+0.043)	0.679(+0.030)	17	0.688(-0.029)	0.378(-0.064)	0.729(-0.031)	
9	0.625(-0.046)	0.259(-0.093)	0.655(-0.057)	18	0.683(-0.011)	0.373(-0.014)	0.704(+0.011)	
10	0.659(-0.001)	0.321(-0.002)	0.650(-0.018)	19	0.746(-0.002)	0.492(-0.005)	0.799(+0.001)	
11	0.646(+0.022)	0.292(+0.045)	0.678(+0.016)	20	0.715(-0.019)	0.435(-0.032)	0.718(0.000)	
P Acc: 5		MCC: 5	AUC: 6	N Acc: 15		MCC: 15	AUC: 13	Z Acc: 0
						MCC: 0	AUC: 1	
α Acc: 0.014 MCC: 0.017 AUC: 0.083								
J48m_{compound}								
Db	Acc	MCC	AUC	Db	Acc	MCC	AUC	
1	0.671(-0.030)	0.347(-0.059)	0.678(-0.042)	11	0.658(-0.048)	0.323(-0.091)	0.669(-0.041)	
2	0.620(-0.004)	0.242(-0.007)	0.631(+0.003)	12	0.642(-0.018)	0.284(-0.035)	0.659(-0.012)	
4	0.537(-0.009)	0.108(+0.007)	0.544(+0.016)	13	0.613(+0.020)	0.223(+0.036)	0.629(+0.002)	
5	0.650(-0.004)	0.322(+0.009)	0.672(-0.003)	14	0.662(-0.064)	0.332(-0.122)	0.675(-0.059)	
6	0.629(-0.012)	0.269(-0.016)	0.646(-0.010)	15	0.634(+0.001)	0.269(+0.004)	0.648(+0.015)	
7	0.615(-0.037)	0.229(-0.077)	0.632(-0.048)	16	0.619(+0.003)	0.235(+0.006)	0.622(-0.002)	
8	0.657(0.000)	0.315(0.000)	0.653(+0.004)	17	0.691(-0.026)	0.386(-0.056)	0.715(-0.045)	
9	0.645(-0.026)	0.290(-0.062)	0.667(-0.045)	18	0.675(-0.019)	0.352(-0.035)	0.670(-0.023)	
10	0.657(-0.003)	0.315(-0.008)	0.647(-0.021)	19	0.761(+0.013)	0.522(+0.025)	0.800(+0.002)	
11	0.642(+0.018)	0.285(+0.038)	0.669(+0.007)	20	0.713(-0.021)	0.428(-0.039)	0.731(+0.013)	
P Acc: 5		MCC: 7	AUC: 8	N Acc: 14		MCC: 13	AUC: 12	Z Acc: 1
						MCC: 0	AUC: 0	
α Acc: 0.013 MCC: 0.038 AUC: 0.059								
J48m_{symptom}								
Db	Acc	MCC	AUC	Db	Acc	MCC	AUC	
1	0.657(-0.044)	0.324(-0.082)	0.682(-0.038)	11	0.654(-0.052)	0.312(-0.102)	0.671(-0.039)	
2	0.626(+0.002)	0.255(+0.006)	0.644(+0.016)	12	0.652(-0.008)	0.303(-0.017)	0.679(+0.008)	
4	0.528(-0.018)	0.071(-0.030)	0.543(+0.015)	13	0.609(+0.016)	0.215(+0.028)	0.629(+0.002)	
5	0.630(-0.024)	0.297(-0.016)	0.658(-0.017)	14	0.660(-0.066)	0.333(-0.120)	0.679(-0.055)	
6	0.633(-0.008)	0.269(-0.016)	0.641(-0.015)	15	0.631(-0.001)	0.261(-0.004)	0.625(-0.008)	
7	0.624(-0.028)	0.246(-0.060)	0.644(-0.036)	16	0.627(+0.011)	0.251(+0.022)	0.635(+0.011)	
8	0.673(+0.016)	0.348(+0.033)	0.665(+0.016)	17	0.687(-0.030)	0.380(-0.062)	0.719(-0.041)	
9	0.610(-0.061)	0.236(-0.116)	0.645(-0.067)	18	0.673(-0.021)	0.350(-0.037)	0.678(-0.016)	
10	0.654(-0.006)	0.311(-0.012)	0.650(-0.018)	19	0.752(+0.004)	0.504(+0.007)	0.802(+0.002)	
11	0.630(+0.006)	0.258(+0.011)	0.671(+0.009)	20	0.713(-0.021)	0.430(-0.037)	0.714(-0.004)	
P Acc: 6		MCC: 6	AUC: 8	N Acc: 14		MCC: 14	AUC: 12	Z Acc: 0
						MCC: 0	AUC: 0	
α Acc: 0.009 MCC: 0.062 AUC: 0.057								

6.4 Support Vector Machines

The Support Vector Machine algorithm produced classifiers with the best performance for protein solubility classification in previous works and has, therefore, been considered the most promising technique also in this thesis. We denote the classifier built on sequence derived attributes by *SMOs* and the classifier built on merged datasets by *SMOm*. The latter is furthermore marked with *SMOm_{disease}*, *SMOm_{drug}*, *SMOm_{symptom}*, *SMOm_{enzyme}* and *SMOm_{compound}* which represent classifiers with added Disease, Drug, Symptom, Enzyme and Compound biomedical concept attributes, respectively.

6.4.1 Sequence derived attribute datasets

Table 6.8 provides the results of *SMOs* built on 20 different datasets where the values in bold show the best performance score for each metric. Note that SMO performed the best (as expected) on all sequence derived attribute datasets when comparing to *NBs* (see Table 6.1) and *J48s* (Table 6.5). In addition, it is the only method that has produced a classifier with accuracy over 80% (for **SdAd_eSol**), which agrees with the accuracy reported in [73]. **SdAd_eSol** also resulted in the highest MCC, AUC, Si Sol and Pr Insol values, while **SdAd_ClustEm17-M** offers the best Si Ins and Pr Sol results.

Table 6.8: Results for *SMOs*.

Attributes Dataset	Acc	MCC	AUC	Si Sol	Si Ins	Pr Sol	Pr Ins
SdAd_Natural-M	0.784	0.575	0.786	0.733	0.839	0.831	0.745
SdAd_Natural-D	0.607	0.215	0.607	0.590	0.625	0.629	0.586
SdAd_Natural-T	0.561	0.117	0.556	0.688	0.425	0.563	0.558
SdAd_Hydro-M	0.661	0.332	0.664	0.586	0.742	0.710	0.624
SdAd_Hydro-T	0.658	0.327	0.661	0.585	0.738	0.706	0.622
SdAd_ConfSimi-M	0.697	0.402	0.700	0.637	0.762	0.743	0.661
SdAd_ConfSimi-D	0.738	0.484	0.740	0.687	0.794	0.782	0.702
SdAd_ConfSimi-T	0.695	0.403	0.698	0.616	0.781	0.752	0.653
SdAd_BlosumSm-M	0.713	0.429	0.714	0.677	0.751	0.745	0.683
SdAd_BlosumSm-D	0.678	0.363	0.680	0.614	0.746	0.722	0.642
SdAd_ClustEm14-M	0.758	0.521	0.759	0.712	0.807	0.799	0.722
SdAd_ClustEm14-D	0.676	0.352	0.676	0.663	0.689	0.697	0.655
SdAd_ClustEm14-T	0.563	0.121	0.559	0.658	0.460	0.568	0.556
SdAd_ClustEm17-M	0.786	0.580	0.788	0.735	0.841	0.833	0.747
SdAd_ClustEm17-D	0.685	0.370	0.685	0.683	0.687	0.702	0.668
SdAd_ClustEm17-T	0.601	0.198	0.598	0.671	0.524	0.603	0.597
SdAd_PhysChem-M	0.746	0.502	0.749	0.686	0.812	0.797	0.706
SdAd_PhysChem-D	0.755	0.514	0.756	0.722	0.790	0.788	0.725
SdAd_Computed	0.738	0.479	0.739	0.719	0.760	0.763	0.715
SdAd_eSol	0.801	0.600	0.800	0.819	0.781	0.801	0.800
Best	0.801	0.600	0.800	0.819	0.841	0.833	0.800

6.4.2 Merged attribute datasets

This section contains results of each $SMOm$ classifier compared with $SMOs$. Again, we use all 20 merged datasets in each measurement since the $SMOs$ classifier built on SdAd_Natural-T performed better than the one built with Naïve Bayes. In general, $SMOm$ performed better on the majority of datasets. In addition, the differences for SdAd_eSol are always positive.

Table 6.9 shows the results for $SMOm_{drug}$ and $SMOm_{symptom}$. Both classifiers results in higher Acc, MCC and AUC values on the majority of the datasets and the α values show that these differences are statistically significant.

Table 6.9: Performance measurements for $SMOm_{drug}$ and $SMOm_{symptom}$ with Information Gain.

$SMOm_{drug}$							
Db	Acc	MCC	AUC	Db	Acc	MCC	AUC
1	0.793(+0.009)	0.594(+0.019)	0.795(+0.009)	11	0.756(-0.002)	0.519(-0.002)	0.758(-0.001)
2	0.616(+0.009)	0.235(+0.020)	0.617(+0.010)	12	0.680(+0.004)	0.361(+0.009)	0.681(+0.005)
3	0.569(+0.008)	0.143(+0.026)	0.571(+0.015)	13	0.591(+0.028)	0.179(+0.058)	0.588(+0.029)
4	0.658(-0.003)	0.328(-0.004)	0.661(-0.003)	14	0.782(-0.004)	0.572(-0.008)	0.784(-0.004)
5	0.669(+0.011)	0.348(+0.021)	0.672(+0.011)	15	0.695(+0.010)	0.391(+0.021)	0.695(+0.010)
6	0.708(+0.011)	0.425(+0.023)	0.711(+0.011)	16	0.598(-0.003)	0.195(-0.003)	0.598(0.000)
7	0.745(+0.007)	0.496(+0.012)	0.747(+0.007)	17	0.751(+0.005)	0.511(+0.009)	0.754(+0.005)
8	0.695(0.000)	0.401(-0.002)	0.698(0.000)	18	0.758(+0.003)	0.519(+0.005)	0.759(+0.003)
9	0.725(+0.012)	0.454(+0.025)	0.726(+0.012)	19	0.745(+0.007)	0.491(+0.012)	0.745(+0.006)
10	0.671(-0.007)	0.350(-0.013)	0.674(-0.006)	20	0.818(+0.017)	0.637(+0.037)	0.819(+0.019)
P Acc: 14 MCC: 14 AUC: 14			N Acc: 6 MCC: 6 AUC: 5			Z Acc: 0 MCC: 0 AUC: 1	
α Acc: 0.003 MCC: 0.004 AUC: 0.02							
$SMOm_{symptom}$							
Db	Acc	MCC	AUC	Db	Acc	MCC	AUC
1	0.788(+0.004)	0.584(+0.009)	0.790(+0.004)	11	0.755(-0.003)	0.517(-0.004)	0.757(-0.002)
2	0.621(+0.014)	0.244(+0.029)	0.622(+0.015)	12	0.682(+0.006)	0.366(+0.014)	0.683(+0.007)
3	0.563(+0.002)	0.141(+0.024)	0.568(+0.012)	13	0.586(+0.023)	0.169(+0.048)	0.582(+0.023)
4	0.656(-0.005)	0.324(-0.008)	0.659(-0.005)	14	0.780(-0.006)	0.567(-0.013)	0.782(-0.006)
5	0.662(+0.004)	0.334(+0.007)	0.665(+0.004)	15	0.695(+0.010)	0.391(+0.021)	0.696(+0.011)
6	0.706(+0.009)	0.419(+0.017)	0.708(+0.008)	16	0.622(+0.021)	0.242(+0.044)	0.619(+0.021)
7	0.743(+0.005)	0.492(+0.008)	0.745(+0.005)	17	0.750(+0.004)	0.507(+0.005)	0.752(+0.003)
8	0.697(+0.002)	0.406(+0.003)	0.700(+0.002)	18	0.758(+0.003)	0.519(+0.005)	0.759(+0.003)
9	0.720(+0.007)	0.444(+0.015)	0.721(+0.007)	19	0.749(+0.011)	0.499(+0.020)	0.749(+0.010)
10	0.670(-0.008)	0.349(-0.014)	0.673(-0.007)	20	0.819(+0.018)	0.639(+0.039)	0.819(+0.019)
P Acc: 16 MCC: 16 AUC: 16			N Acc: 4 MCC: 4 AUC: 4			Z Acc: 0 MCC: 0 AUC: 0	
α Acc: 0.09 MCC: 0.004 AUC: 0.005							

Table 6.10 shows results for $SMOm_{disease}$, $SMOm_{compound}$ and $SMOm_{enzyme}$. In general, these classifiers also outperformed the corresponding $SMOs$ on the majority of the datasets.

However the Wilcoxon signed-ranks test indicates that these differences are statistically not significant.

Table 6.10: Performance measurements for $SMOm_{disease}$, $SMOm_{compound}$ and $SMOm_{enzyme}$ with Information gain.

$SMOm_{disease}$							
Db	Acc	MCC	AUC	Db	Acc	MCC	AUC
1	0.785(+0.001)	0.577(+0.002)	0.787(+0.001)	11	0.756(-0.002)	0.518(-0.003)	0.758(-0.001)
2	0.618(+0.011)	0.239(+0.024)	0.619(+0.012)	12	0.678(+0.002)	0.357(+0.005)	0.679(+0.003)
4	0.548(-0.013)	0.138(+0.021)	0.532(-0.024)	13	0.584(+0.021)	0.164(+0.043)	0.579(+0.020)
5	0.657(-0.004)	0.324(-0.008)	0.660(-0.004)	14	0.777(-0.009)	0.561(-0.019)	0.779(-0.009)
6	0.656(-0.002)	0.322(-0.005)	0.659(-0.002)	15	0.690(+0.005)	0.381(+0.011)	0.690(+0.005)
7	0.700(+0.003)	0.407(+0.005)	0.702(+0.002)	16	0.614(+0.013)	0.224(+0.026)	0.611(+0.013)
8	0.742(+0.004)	0.490(+0.006)	0.743(+0.003)	17	0.747(+0.001)	0.502(0.000)	0.749(0.000)
9	0.697(+0.002)	0.404(+0.001)	0.699(+0.001)	18	0.756(+0.001)	0.516(+0.002)	0.758(+0.002)
10	0.714(+0.001)	0.431(+0.002)	0.715(+0.001)	19	0.745(+0.007)	0.492(+0.013)	0.746(+0.007)
11	0.673(-0.005)	0.353(-0.010)	0.675(-0.005)	20	0.815(+0.014)	0.631(+0.031)	0.816(+0.016)
P Acc: 14		MCC: 14	AUC: 13	N Acc: 6		MCC: 5	AUC: 6
				Z Acc: 8		MCC: 1	AUC: 1
α Acc: 0.172 MCC: 0.053 AUC: 0.197							
$SMOm_{compound}$							
Db	Acc	MCC	AUC	Db	Acc	MCC	AUC
1	0.790(+0.006)	0.588(+0.013)	0.792(+0.006)	11	0.751(-0.007)	0.509(-0.012)	0.753(-0.006)
2	0.616(+0.009)	0.234(+0.019)	0.617(+0.010)	12	0.680(+0.004)	0.361(+0.009)	0.680(+0.004)
4	0.556(-0.005)	0.118(+0.001)	0.559(+0.003)	13	0.594(+0.031)	0.185(+0.064)	0.589(+0.030)
5	0.655(-0.006)	0.320(-0.012)	0.658(-0.006)	14	0.777(-0.009)	0.562(-0.018)	0.779(-0.009)
6	0.660(+0.002)	0.330(+0.003)	0.663(+0.002)	15	0.694(+0.009)	0.389(+0.019)	0.694(+0.009)
7	0.702(+0.005)	0.410(+0.008)	0.704(+0.004)	16	0.617(+0.016)	0.231(+0.033)	0.615(+0.017)
8	0.743(+0.005)	0.492(+0.008)	0.745(+0.005)	17	0.747(+0.001)	0.502(0.000)	0.749(0.000)
9	0.694(-0.001)	0.397(-0.006)	0.697(-0.001)	18	0.758(+0.003)	0.519(+0.005)	0.759(+0.003)
10	0.716(+0.003)	0.436(+0.007)	0.718(+0.004)	19	0.742(+0.004)	0.486(+0.007)	0.743(+0.004)
11	0.671(-0.007)	0.351(-0.012)	0.674(-0.006)	20	0.815(+0.014)	0.630(+0.030)	0.815(+0.015)
P Acc: 14		MCC: 14	AUC: 14	N Acc: 6		MCC: 5	AUC: 5
				Z Acc: 0		MCC: 1	AUC: 1
α Acc: 0.135 MCC: 0.076 AUC: 0.070							
$SMOm_{enzyme}$							
Db	Acc	MCC	AUC	Db	Acc	MCC	AUC
1	0.790(+0.006)	0.587(+0.012)	0.792(+0.006)	11	0.753(-0.005)	0.513(-0.008)	0.755(-0.004)
2	0.614(+0.007)	0.231(+0.016)	0.615(+0.008)	12	0.680(+0.004)	0.361(+0.009)	0.681(+0.005)
4	0.534(-0.027)	0.073(-0.044)	0.517(-0.039)	13	0.585(+0.022)	0.165(+0.044)	0.581(+0.022)
5	0.654(-0.007)	0.319(-0.013)	0.657(-0.007)	14	0.778(-0.008)	0.564(-0.016)	0.781(-0.007)
6	0.662(+0.004)	0.334(+0.007)	0.665(+0.004)	15	0.693(+0.008)	0.386(+0.016)	0.693(+0.008)
7	0.703(+0.006)	0.412(+0.010)	0.705(+0.005)	16	0.599(-0.002)	0.197(-0.001)	0.598(0.000)
8	0.743(+0.005)	0.492(+0.008)	0.745(+0.005)	17	0.750(+0.004)	0.508(+0.006)	0.752(+0.003)
9	0.693(-0.002)	0.397(-0.006)	0.696(-0.002)	18	0.757(+0.002)	0.518(+0.004)	0.758(+0.002)
10	0.719(+0.006)	0.441(+0.012)	0.720(+0.006)	19	0.742(+0.004)	0.485(+0.006)	0.742(+0.003)
11	0.674(-0.004)	0.355(-0.008)	0.677(-0.003)	20	0.818(+0.017)	0.636(+0.036)	0.818(+0.018)
P Acc: 13		MCC: 13	AUC: 13	N Acc: 7		MCC: 7	AUC: 6
				Z Acc: 0		MCC: 0	AUC: 1
α Acc: 0.197 MCC: 0.197 AUC: 0.136							

6.5 Results overview

This section offers an overview of the results presented in the previous three sections. It also introduces the algorithm for building the classifier with highest performance scores for the eSol proteins. Table 6.11 shows an overview of the statistical significance of the results for each classifier and each biomedical concept group with Information Gain as the feature selection method. Results are presented with the following differences: “no statistically significant difference” (ND), “statistically significantly better” (SB) and “statistically significantly worse” (SW).

Table 6.11: Overview of the statistical significance of the results.

	Naïve Bayes			J48			SMO		
	Acc	MCC	AUC	Acc	MCC	AUC	Acc	MCC	AUC
Disease	ND	SB	ND	SW	SW	ND	ND	ND	ND
Drug	ND	ND	SB	ND	ND	ND	SB	SB	SB
Compound	ND	ND	SB	SW	SW	ND	ND	ND	ND
Symptom	ND	ND	SB	SW	ND	ND	SB	SB	SB
Enzyme	ND	ND	ND	ND	ND	ND	ND	ND	ND

Next, we also present detailed results for the best two classifiers where the first is built with only sequence derived attributes and the second is built on the attributes with added biomedical concepts. Both classifiers were built with SMO, where the former (*SMOs*) was built with the **SdAd_eSol** attribute dataset (see Table 6.8) while the latter (*SMOm*) was built with the **MAd_eSol_Symptom** attribute dataset (see Table 6.9, Db 20) with 30 selected attributes. Both classifiers use a modified polynomial kernel (See Section 2.4.4, Equation 2.14) where the output value $k(x, y)$ is normalized in the following way:

$$\frac{k(x, y)}{\sqrt{k(x, x)k(y, y)}} \quad (6.1)$$

Results for both classifiers are presented in Table 6.12. As the table shows, *SMOm* outperformed *SMOs* in all metrics except for Si Sol where there is a small difference in *SMOs*’ favour.

Table 6.12: Results for two best classifiers built on SdAd_eSol and MAd_eSol_Symptom.

	Acc	MCC	AUC	Si Sol	Si Ins	Pr Sol	Pr Ins
SMOm	0.819	0.639	0.819	0.816	0.822	0.832	0.806
SMOs	0.801	0.600	0.800	0.819	0.781	0.801	0.800

According to these results, we construct the algorithm for building $SMO_{m_{symptom}}$ for solubility classification of the eSol database, which results in the highest performance scores. The algorithm is as follows:

Algorithm 6.1: The algorithm for classifying the eSol proteins in biomedical concept space.

Build SMO classifier for protein solubility classification:

P : set of eSol protein names

P_i : name of a single protein

Q : FACTA query

$C_{symptom}$: a set of symptom biomedical concepts with their frequencies

C_j : a single symptom biomedical concept

a : a single attribute

a_n : the name of a single attribute a

a_v : the value of a single attribute a_j

A_{sd} : a set of sequence derived attributes (i.e., frequencies of Natural monomers with Mw and Ip)

A_{bc} : a set of biomedical concept attributes

A_m : a set of merged (combined) attribute datasets

FS_{ig} : the Information Gain feature selection method

SMO_{npk} : the SMO classification method using the normalized polynomial kernel

For each P_i from P (where $i = 1$ to $|P|$) do:

$Q = P_i$

Query FACTA with Q for frequencies of Symptom biomedical concepts

Save returned biomedical concepts and their frequencies into $C_{symptom}$

For each C_j from $C_{symptom}$ (where $j = 1$ to $|C_{symptom}|$) do:

Define new attribute a

$a_n =$ name of the concept C_j

$a_v =$ frequency of the concept C_j

Determine type k of C_i

Save a to the A_{bc}

$A_m = A_{sd} + A_{bc}$

Select 30 attributes from A_m with FS_{ig} and save them to A_m

Build classifier with SMO_{npk} with A_m

7

Discussion

7.1 Introduction

In this section we analyse the results obtained by the experiments detailed in Section 6 and see how they relate to the hypothesis of this thesis, introduced in Section 1.2.1. We also discuss the main problems that we have encountered during the progress of this thesis. In addition, we identify the scientific contributions that result from this thesis and were introduced in Section 1.2.2. Let us start with repeating the hypothesis which is as follows:

Hypothesis:

A new, more successful classifier for protein solubility classification can be built using biomedical concept attributes, which do not depend only on the primary structure of a protein.

The hypothesis was extended into two sub-hypotheses:

Sub-hypothesis 1:

With merging the most relevant sequence derived attributes and the most relevant biomedical concept attributes we improve protein solubility classification for some methods.

Sub-hypothesis 2:

Protein attributes derived from the protein's primary structure do not carry all information needed for optimal protein solubility classification.

7.2 Discussion of the results

To confirm the hypothesis, we performed an empirical study and evaluated the performance of three common classification methods, i.e., Naïve Bayes, Decision trees (J48) and Support Vector Machine (SMO). Each of the methods was used with (a) the sequence derived attributes that had shown to be most relevant in solubility classification, and (b) the most relevant biomedical attributes extracted using FACTA and merged with the most relevant sequence derived attributes from (a).

In general, the results show that adding relevant biomedical concept attributes influences the performances for all three used classification methods. In addition, the best classifiers (i.e., the classifiers with the highest Acc, MCC and AUC scores obtained by each method) were using the added biomedical concept attributes. These classifiers were $NBm_{compound}$ (Table 6.4), $J48m_{symptom}$ (Table 6.7) and $SMOm_{symptom}$ (Table 6.9). However, practically interesting is only the latter, since differences in $NBm_{compound}$ and $J48m_{symptom}$ were hardly noticeable (lower than 1%) in each metric.

A detailed comparison of $SMOm_{symptom}$ with the best classifier obtained by only sequence derived attributes ($SMOs$) has been presented in Table 6.12. Both classifiers were obtained by the SdAd_eSol sequence derived attributes, where $SMOm_{symptom}$ also used biomedical concept attributes from the Symptom group. The differences in the table show a slight improvement in the performance. In particular, the difference in the accuracy is almost 2% in $SMOm_{symptom}$'s favour, which means that this classifier classified correctly around 30 proteins more when compared to $SMOs$. An analysis of these proteins indicates that most of them were insoluble, which can be seen in the Sensitivity of Soluble proteins metric, where $SMOm_{symptom}$ performs worse than $SMOs$. More insoluble proteins being correctly classified is an important factor, since misclassified insoluble proteins waste researchers' time and resources on studying wrong proteins.

When testing the statistical significance of the results, only Naïve Bayes and SMO show improvements that are statistically significant, while the performance of J48 significantly degrades with some of the added attributes. Of course, it would be ideal if all classification algorithms showed statistically significant improvements. However, it is a well known fact in machine learning that there is no solution which is the best in every situation. This theory is also called *no free lunch* and was introduced in [7]. In addition, the sequence derived attribute datasets used in this thesis were mainly selected and optimized for the use with SVM and Naïve Bayes [34], since most previous research has used those two methods. Also, J48 offers customisation of several parameters that influence the final classification process. Since pattern search has been used in this thesis (see Section 2.5.2) to optimise parameters,

the optimal parameter values might have not been found. In addition, decision trees are prone to over-fit to learning data, which could also be the reason for their poor performance. Nevertheless, this work opens up further possibilities of researching decision trees in connection with protein solubility classification with biomedical concepts.

Considering the fact that the best overall classifier has been built with the merged attribute dataset, which contains the most relevant (i.e., selected by feature selection method) attributes, and taking into account the results of the statistical tests, we can confirm sub-hypothesis 1. In addition, we have shown that the new attributes increased the classifier's performance, thus, we can conclude that these new attributes offer important information for the classifier. This information was obviously not included in the most relevant sequence-derived attributes. As a result, we can confirm the sub-hypothesis 2. With the confirmation of the both sub-hypotheses, we can also confirm the main hypothesis.

The results also show some interesting findings. First, we have noticed that lower numbers of selected attributes contribute to better results, while higher numbers degrade the performance of classifiers. This behaviour has been illustrated on Figure 6.2 and it has also been confirmed with the number of selected attributes of the best $SMO_{m_{symptom}}$ classifier. As mentioned in Section 6.5, only 30 out of total 259 attributes were selected to obtain this classifier. We can conclude that the majority of biomedical concept attributes is irrelevant for protein solubility classification, which was expected, since FACTA has not been designed to mine only for concepts that are related to protein solubility. In addition, we believe that there is another reason for the high number of useless attributes. FACTA (and text mining techniques in general) is not 100% accurate when mining for relevant concepts. Some of the returned concepts are simply not related to the input query and they are usually hard to identify. We believe that, with improvement of text mining techniques, more useful attributes could be used for protein solubility classification.

The next interesting finding is that the Drug and the Symptom biomedical concept attribute groups step out when comparing the results (Table 6.11). This could be an interesting research topic for biologist and other experts since, as mentioned in Section 1, insoluble proteins many times result in different diseases. However, we can also notice that the Drug and the Symptom groups contain the lowest numbers of attributes amongst all biomedical concept attribute groups (Table 5.5). As mentioned in the previous paragraph, FACTA returns a large amount of useless attributes. As a result, these attributes can overshadow relevant attributes, and the latter can be ignored by feature selection methods. Datasets with higher numbers (e.g. Enzyme) of attributes are more likely to be sensitive to this behaviour.

During the progress of this thesis, we have encountered a few important problems and recognised that the protein solubility classification is a difficult and complex task. The first problem is finding an appropriate definition of protein solubility. Since different research groups use different definitions, careful research has to be done by computer scientists to understand these definitions. The second problem comes from publicly available protein databases which do not contain systematically documented information about solubility. As a result, it is often hard to identify the types of solubility that different databases include. In addition, these databases are often unbalanced, a factor that has to be considered during the design of the classification schemes. Finally, the third problem is choosing the right text mining techniques that would describe proteins with useful biomedical concepts and offer fast processing, so that the whole classification process would not be significantly hampered.

The scientific contributions of this thesis can be summarized as follows. Firstly, the thesis introduced the idea of using information from scientific literature for protein solubility classification. In addition, a method for extracting this information from medical literature and forming biomedical concept attributes was introduced. Also, an analysis of text mining techniques that can assist in this process was done. Secondly, this thesis recognised that most of the protein data sets available today do not document information about protein solubility well. We believe this information should be organized and systematically documented, thus, enabling computer algorithms to analyse and process it more efficiently. Thirdly, this thesis performed an original comparison of standard protein solubility classification methods with methods that use biomedical concepts. Finally, the results of this thesis have determined which biomedical concept groups increased the classifiers' performance scores the most. This would be interesting for biologist and other professionals who should investigate how solubility is connected with different symptoms or drug. In depth future research should be done to investigate these results.

8

Conclusion

In this thesis we presented an advanced classification scheme for protein solubility classification which uses new attributes in the form of biomedical concepts. In the first chapter we presented the motivation, introduced the expected goals and scientific contributions, and defined the hypothesis.

In the second section we presented a detailed description of machine learning concepts that built the basis for the new classification scheme. In particular, we focused on supervised classification techniques and described four unique classification techniques, each representing a distinctive way of building classifiers. We also presented the main problems in building classifiers, such as over-fitting and under-fitting of classifiers, and introduced methods for facing these problems, such as feature selection methods for reducing number of attributes and methods for optimizing classifiers' parameters.

In the third section we introduced the biological concepts needed for understanding the new classification scheme. Specifically, we described the basics of proteins and focused on protein structure. Four special levels of protein structure were described and the connection between them was given. In addition, the section explained different types of protein solubility and performed an analysis some of the databases that had been used for protein solubility classification. The section concludes with a detailed description of the eSol database, which had been used for the experiments performed in this thesis.

In the fourth section we introduced and performed an analysis of the text mining techniques that can be used to assist extracting new attributes from the eSol proteins. We described the basics steps of extracting useful patterns from unstructured text, such as tokenization and lemmatization. We also defined the term *biomedical concept* in the scope of the thesis and

described the process of extracting biomedical concepts from medical literature. Specifically, we introduced FACTA and illustrated the steps it uses for extracting names and frequencies for six different groups (i.e., Disease, Drug, Symptom, Compound and Enzyme) of biomedical concepts.

In the fifth section we described the experimental environment for our empirical study. First, we described the classification scheme with a detailed description of all its steps, particularly for the attribute extraction and selection steps. In addition, we also described a method for converting biomedical concepts into biomedical concept attributes. In the second part of this section, we described the Weka framework which had been used for implementing the classification scheme and for running the experiments.

In the sixth section we presented the results of each classification algorithm used. We compared the classifiers built using only sequence derived attributes with those built also using biomedical concept attributes. We used the Wilcoxon signed-ranks test to determine whether there is any statistically significant improvement in the classifier's performances.

In the seventh section we discussed the results and identified the scientific contributions of this thesis. The thesis confirmed the hypothesis given in the first section and concluded that knowledge and information from medical literature can be used as input for solubility classification techniques. Moreover, we demonstrated that the combination of text mining techniques and classification techniques results in classifiers with better performance in the protein solubility classification. All three techniques resulted in higher performance scores when combining some of the sequence derived attribute datasets with biomedical concept datasets. In addition, the Naïve Bayes and Support Vector Machine classifiers also showed that these improvements are statistically significant.

This thesis offers several directions for the future work. Firstly, it would be interesting for biologist and other experts to investigate why symptom and drug biomedical concepts result in the most improved protein solubility classification. Secondly, the use of biomedical concepts might also be considered in other areas of protein classification, such as protein functional classification [90], prediction of protein-protein interaction [8-10] or protein fold recognition [64]. Finally, although we succeeded in building an improved classifier, the classifier's performance scores are still not satisfactory and should be improved. Protein solubility classification is a very complex task and biomedical concepts should definitely not be the main attributes used in this process. However, as this thesis has shown, they can assist in the classification process, and with improvement of text mining techniques, this assistance might become more important.

References

- [1] Branden, C., Tooze, J., Introduction to Protein Structure, 2nd ed., Garland Publishing, 1999.
- [2] Chow, M., K., Amin, A., A., Fulton, K., F., Whisstock, J.C., Buckle, A. M., Bottomley, S. P., REFOLD: an analytical database of protein refolding methods, Protein Expression and Purification, 2006.
- [3] Baneyx, F., Recombinant protein expression in Escherichia coli, Current Opinion in Biotechnology, 1999.
- [4] Chow, M., K., Amin, A., A., Fulton, K., F., Fernando, T., Kamau, L., Batty, C., Louca, M., Ho, S., Whisstock, J. C., Bottomley, S., P., Buckle, A. M., The REFOLD database: a tool for the optimization of protein expression and refolding, Nucleic Acids Research, 2006.
- [5] Singh, S.M. and Panda, A.K., Solubilization and refolding of bacterial inclusion body proteins. J. Biosci. Bioeng., 2005.
- [6] Alpaydin, E. Introduction to Machine Learning. The MIT Press, 2004.
- [7] Wolpert, D., Macready, W., "No free lunch theorems for optimization." IEEE Transactions on Evolutionary Computation, 1(1), 67-82, 1997.
- [8] Bock, J., R., Gough, D., A., Predicting protein - protein interactions from primary structure. Bioinformatics, 17:455-460, 2001.
- [9] Bock, J., R., Gough, D., A., Whole-proteome interaction mining. Bioinformatics, 19:125-134, 2003.
- [10] Lo, S., L., Cai, C., Z., Chen, Y., Z., Chung, M., C, Effect of training datasets on support vector machine prediction of protein-protein interactions. Proteomics, 5:876-884, 2005.
- [11] Makrides, S.C., Strategies for achieving high-level expression of genes in Escherichia coli. Microbiol Rev., 60, 512-538, 1996.
- [12] Tresaugues, L., Collinet, B., Minard, P., Henckes, G., Aufrère, R., Blondeau, K., Liger, D., Zhou, C., Z., Janin J., Van Tilbeurgh, H., Quevillon-Cheruel, S., Refolding strategies from inclusion bodies in a structural genomics project. J. Struct. Funct. Genomics, 5, 195-204, 2004.
- [13] Christendat, D., Yee, A., Dharamsi, A., Kluger, Y., Savchenko, A., Cort, J., R., Booth, V., Mackereth, C., D., Saridakis, V., Ekiel, I., Kozlov, G., Maxwell, K., L., Wu, N., McIntosh, L., P., Gehring, K., Kennedy, M., A., Davidson, A., R., Pai, E., F., Gerstein, M., Edwards, A., M., Arrowsmith, C., H., Structural proteomics of an archaeon. Nat. Struct. Biol., 7, 903-909, 2000.

- [14] Bertone, P., Kluger, Y., Lan, N., Zheng, D., Christendat, D., Yee, A., Edwards, A., M., Arrowsmith, C., H., Montelione, G., T., Gerstein, M., SPINE: an integrated tracking database and data mining approach for identifying feasible targets in high-throughput structural proteomics. *Nucleic Acids Res.*, 29, 2884–2898, 2001.
- [15] Izard, J., Parker, M., W., Chartier, M., Duché, D., Baty, D., A single amino acid substitution can restore the solubility of aggregated colicin A mutants in *Escherichia coli*. *Protein Eng.*, 7, 1495–1500, 1994.
- [16] Pietrokovski, S., Henikoff, J. G., Henikoff, S., The Blocks database—a system for protein classification, *Nucleic Acids Research*, 24, 1, 197–200, 1996.
- [17] Xiaojing, Y., Yuan X., Yang F., Peng J., Buckles, B., P., Gene Expression Classification: Decision Trees vs. SVMs. “FLAIRS Conference, 92–97, 2003.
- [18] Murby, M., Samuelsson, E., Nguyen, T., N., Mignard, L., Power, U., Binz, H., Uhlén, M., Ståhl, S., Hydrophobicity engineering to increase solubility and stability of a recombinant protein from respiratory syncytial virus. *Eur. J. Biochem.*, 230, 38–44, 1995.
- [19] Murphy, L., R., Wallqvist, A., Levy, R., M., Simplified amino acid alphabets for protein fold recognition and implications for folding. *Protein Eng.*, 13, 149–152, 2000.
- [20] Dempster, A., P., Laird N., M., Rubin, D., B., Maximum likelihood from incomplete data via the EM algorithm. *J. Roy. Statist. Soc.*, 39, 1–38, 1977.
- [21] Kawashima, S. and Kanehisa, M., AAindex: amino acid index database. *Nucleic Acids Res.*, 28, 374, 2000.
- [22] Ambrose, C., McLachlan G.J., Selection bias in gene extraction on the basis of microarray gene-expression data, *Proc Natl Acad Sci USA* 2002, 99:6562–6566, 2002.
- [23] Smialowski P, Frishman D, Kramer S., Pitfalls of supervised feature selection., *Bioinformatics*, 26, 3, 440–443, 2010
- [24] Baldi, P., Brunak, S., Chauvin, Y., Andersen, C., A., Nielsen, H., Assessing the accuracy of prediction algorithms for classification: an overview, *Bioinformatics*, 16(5):412–24, 2000.
- [25] Fawcett, T., An introduction to ROC analysis. *Pattern Recognition Letters*, 27, 861–874, 2006.
- [26] Chakrabarti, P., Pal, D., The interrelationships of side-chain and main-chain conformations in proteins. *Prog. Biophys. Mol. Biol.*, 76, 1–102, 2001.
- [27] Lin, S., M, McConnell, P., Johnson, K., F., Shoemaker, J., MedlineR: an open source library in R for Medline literature data mining, 12;20(18):3659–61, *Bioinformatics*, 2004.
- [28] Chang, C., C., Lin, C.J., LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1--27:27, 2011.

- [29] Idicula-Thomas, S., Kulkarni, A., J., Kulkarni, B. D., Jayaraman, V., K., Balaji, P. V., A support vector machine-based method for predicting the propensity of a protein to be soluble or to form inclusionbody on overexpression in *Escherichia coli*, 22, 3, 278-284, *Bioinformatics*, 2006.
- [30] Smailowski, P., Martin-Galiano, A., J., Mikolajka, A., Girchick, T., Holak, T., A., Frishman, D., Protein solubility: sequence based prediction and experimental verification, *Bioinformatics*, 23, 19, 2536-2542, 2007.
- [31] Wilkinson, D., L., Harrison, R., G., Predicting the solubility of recombinant proteins in *Escherichia coli*, *Biotechnology*, 9, 443 - 448, 1991.
- [32] Goh, C., S., Lan, N., Douglas, S., M., Wu, B., Echols, N., Smith, A., Milburn, D., Montelione, G., T., Zhao, H., Gerstein, M., Mining the structural genomics pipeline: identification of protein properties that affect high-throughput experimental analysis. *J. Mol. Biol.*, 336, 2004.
- [33] Davis, G., D., Elisee, C., Newham, D., M., Harrison, R., G., New fusion protein systems designed to give soluble expression in *Escherichia coli*. *Biotechnol. Bioeng.*, 65, 382–388, 1999.
- [34] Magnan, C., N., Randall, A., Baldi, P., SOLpro: accurate sequence-based prediction of protein solubility, *Bioinformatics*, 1;25(17):2200-7, 2009.
- [35] Ong, S., A., Lin, H., H., Chen, Y., Z., Li, Z., R., Cao, Z., Efficacy of different protein descriptors in predicting protein functional families, *BMC Bioinformatics*, 8, 2007.
- [36] Furey, T., Cristianini, N., Duffy, N., Bednarski, D., Schummer, M., and Haussler. D., “Support vector machines classification and validation of cancer tissue samples using microarray expression data”, *Bioinformatics*, vol. 16, pp. 906-914, 2000.
- [37] Langley, P., *Elements of machine learning*. San Francisco, CA: Morgan Kaufmann Publishers Inc., 1995.
- [38] Mitchell, T., *Machine Learning*. McGraw Hill, 1997.
- [39] Rose, G., D., Geselowitz, A., R., Lesser, G., J., Lee, R., H., Zehfus., M., H., Hydrophobicity of amino acid residues in globular proteins. *Science*, 229, 834–838, 1985.
- [40] Sewell, M., *Machine Learning*. 2006.
<http://machine-learning.martinsewell.com/machine-learning.pdf> (Accessed 8.7.2011).
- [41] Chung, W., E., Micheli-Tzanakou., *Classifiers: An overview*, in *Supervised and Unsupervised Pattern Recognition: Feature Extraction and Computational Intelligence*, avtor E. Micheli-Tzanakou, 3-60. Boca Raton, FL: CRC Press Inc., 2000.
- [42] Russell, S., J., Norvig, P., *Artificial Intelligence: A Modern Approach* (2nd ed.). Upper Saddle River, NJ: Prentice Hall, 2003.
- [43] Quinlan, J., R., *Induction of Decision Trees*, *Machine Learning*, (1), 81-106 1986.

- [44] Quinlan, J., R., C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, 1993.
- [45] Breiman, L., Friedman, J., H., Olshen, R., A., Stone, C., J., Classification and regression trees. Belmont, CA., ZDA: Wadsworth International Group, 1984.
- [46] Vapnik, V., N., The Nature of Statistical Learning Theory. Springer, 1995.
- [47] Chen, L., Oughtred, R., Berman, H., M., Westbrook, J., TargetDB: a target registration database for structural genomics projects, *Bioinformatics*, 2004.
- [48] Li, Z., R., Lin, H., H., Han, L., Y., Jiang, L., Chen, X., Chen, Y., Z., PROFEAT: a web server for computing structural and physicochemical attributes of proteins and peptides from amino acid sequence, *Nucleic Acids Research*, 2006.
- [49] Hall, M, Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I., H., The WEKA Data Mining Software: An Update; *SIGKDD Explorations*, Volume 11, Issue 1, 2009.
- [50] Tsuruoka, Y., Tsujii, J., Ananiadou, S., FACTA: a text search engine for finding associated biomedical concepts, *Bioinformatics*, Vol. 24, No. 21, pp. 2559-2560, 2008.
- [51] Witten, I., Frank, E., *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd Edition, Morgan Kaufmann, 2005.
- [52] Duda, R., O., Hart, P., E., and Stork, D., G., *Pattern Classification (2nd Edition)*. Wiley- Interscience, 2000.
- [53] Demsar, J., Statistical comparisons of classifiers over multiple data sets, *Journal of Machine Learning Research*, 7:1–30, 2006.
- [54] Antonie, M., L., Zaiane, O., R., Holte, R., C., "Learning to Use a Learned Model: A Two-Stage Approach to Classification," *Sixth International Conference on Data Mining (ICDM'06)*, pp. 33-42, 2006.
- [55] Dong, G., Pei, J., *Sequence Data Mining*, Springer, 2007.
- [56] Trevino, S., R., Scholtz, J., M., Pace, C., N., Measuring and increasing protein solubility, *Journal of Pharmaceutical Sciences*, 2008.
- [57] Mishne, G., Glance, N., *Predicting Movie Sales from Blogger Sentiment*, 2006.
- [58] Idicula-Thomas, S., Balaji, P. V., Understanding the relationship between the primary structure of proteins and their amyloidogenic propensity: clues from inclusion body formation, *Protein engineering, design & selection : PEDS*, 2005.
- [59] Wilcoxon, F., Individual comparisons by ranking methods. *Biometrics*, 1:80–83, 1945.
- [60] Perez-Iratxeta, C., Pérez, A., J., Bork, P., Andrade, M., A., Update on XplorMed: a web server for exploring scientific literature. *Nucleic Acids Res.*, 31, 3866–3868, 2003.
- [61] Wheeler, D., L., Barrett, T., Benson, D., A., Bryant, S., H., Canese, K., Church, D., M., DiCuccio, M., Edgar, R., Federhen, S., Helmberg, W., Database resources of the

- National Center for Biotechnology Information. *Nucleic Acids Res.*, 33, D39–D45, 2005.
- [62] Maier, H., Döhr, S., Grote, K., O'Keeffe, S., Werner, T., Hrabé de Angelis, M., Schneider, R., *LitMiner and WikiGene: identifying problem-related key players of gene regulation using publication abstracts.* *Nucleic Acids Res.*, 33, W779–W782, 2005.
- [63] Dorian, P., *Data Preparation for Data Mining (The Morgan Kaufmann Series in Data Management Systems)*, Morgan Kaufmann Publishers, 1999.
- [64] Dubchak, I., Muchnick, I., Mayor, C., Dralyuk, I., Kim, S., H., *Recognition of a protein fold in the context of the Structural Classification of Proteins (SCOP) classification.* *Proteins*, 35:401-407, 1999.
- [65] Lindorff-Larsen, K., Røgen, P., Paci, E., Vendruscolo, M., Dobson, C., M., *Protein folding and the organization of the protein topology universe,* *Trends in Biochemical Sciences*, 2005.
- [66] Dill, K., A., Ozkan, S., B., Weikl, T., R., Chodera, J., D., Voelz, V., A., *The protein folding problem: when will it be solved?*, *Current opinion in structural biology*, 2007.
- [67] Uversky, V., N., Gillespie, J., R., Fink, A., L., *Why are "natively unfolded" proteins unstructured under physiologic conditions?*, *Proteins*, 2000.
- [68] Shimizu Y, Inoue, A., Tomari, Y., Suzuki, T., Yokogawa, T., Nishikawa, K., Ueda, T., *Cell-free translation reconstituted with purified components.* *Nat Biotechnol* 19:751–755, 2001.
- [69] Ying BW, Taguchi H, Ueda H, Ueda T (2004) *Chaperone-assisted folding of a singlechain antibody in a reconstituted translation system.* *Biochem Biophys Res Commun*, 320:1359–1364.
- [70] Feldman, R., Sanger, J., *The text mining handbook*, Cambridge University Press, 2007.
- [71] Berman, H., M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T., N., Weissig, H., Shindyalov, I., N., Bourne, P., E., *The Protein Data Bank.* *Nucleic Acids Res.*, 28, 235–242, 2000.
- [72] *The UniProt Consortium, The Universal Protein Resource (UniProt)*, 2007.
- [73] Niwa, T., Ying, B., W., Saito, K., Jin, W. Z., Takada, S., Ueda, T., Taguchi, H., *Bimodal protein solubility distribution revealed by an aggregation analysis of the entire ensemble of Escherichia coli proteins,* *Proc. Natl. Acad. Sci. U.S.A.* 106, 4201-4206, 2009.
- [74] Kyte, J., and Doolittle, R., F., *A simple method for displaying the hydropathic character of a protein.* *J. Mol. Biol.* 157, 105-132, 1982.
- [75] Gasteiger E., Hoogland C., Gattiker A., Duvaud S., Wilkins M., R., Appel R., D., Bairoch A *Protein Identification and Analysis Tools on the ExPASy Serv.r.* (In) John M. Walker (ed): *The Proteomics Protocols Handbook*, Humana Press, pp. 571-607m 2005.

- [76] Platt, J., Machines using Sequential Minimal Optimization. In B. Schoelkopf and C. Burges and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, 1998.
- [77] Kohavi, R., John, H., G., Wrappers for feature subset selection, *Artificial Intelligence*, 1997.
- [78] Pearson, K., "On Lines and Planes of Closest Fit to Systems of Points in Space," *Philosophical Magazine* 2 (6): 559–572, 1901.
- [79] Deza, M., M., Deza, E., *Encyclopedia of distances*, Springer, 2009.
- [80] Pang, B., Lee, L., *Opinion Mining and Sentiment Analysis*, 2008.
- [81] Sebastiani, F., Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.
- [82] Niwa, T., Ying, B., W., Saito, K., Jin, W., Takada, S., Ueda, T., Taguchi, H., Intrusion detection in web applications using text mining, *Proc Natl. Acad. Sci.*, 106(11):4201-6, 2009
- [83] Clark, E., D., B., Refolding of recombinant proteins. *Curr. Opin. Biotechnol.*, 9, 1998.
- [84] Ghosh, S., Rasheedi, S., Rahim, S., S., Banerjee, S, Choudhary, R., K, Chakhaiyar, P., Ehtesham, N., Z., Mukhopadhyay, S., Hasnain, S., E., Method for enhancing solubility of the expressed recombinant proteins in *Escherichia coli*. *Biotechniques*, 37, 418, 420, 422–423, 2004.
- [85] Machida, S., Yu, Y., Singh, S., P, Kim, J., D., Hayashi, K., Kawata, Y., Overproduction of beta-glucosidase in active form by an *Escherichia coli* system coexpressing the chaperonin GroEL/ES. *FEMS Microbiol Lett.*, 159, 41–46, 1998.
- [86] Trevino, S., R., Scholtz, J., M., Pace, C., N., Measuring and increasing protein solubility, *J Pharm Sci.*, 97(10):4155-66, 2008.
- [87] Quinlan, J., R., Improved use of continuous attributes in c4.5. *Journal of Artificial Intelligence Research*, 4:77-90, 1996.
- [88] Burges C., J., C., *A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery*, 2, 1998.
- [89] Lehninger, A., Nelson, D., L. Michael, M., C., *Principles of Biochemistry*, Fourth Edition, W. H. Freeman & Co, 2004.
- [90] Cai, C., Z., Han, L., Y., Ji, Z., L., Chen, X., Chen, Y., Z., SVM-Prot: web-based support vector machine software for functional classification of a protein from its primary sequence. *Nuclei. Acid. Res.*, 31:3692-3697, 2003.

Biography

Name:	Simon Kocbek
Date and place of birth:	18.6.1980, Maribor
Education:	1995 Conclusion of primary school Tabor II in Maribor.
	1999 Conclusion of Druga Gimnazija Maribor (Second Grammar School Maribor) in Maribor.
	2005 Conclusion of at the Faculty of Electrical Engineering and Computer Science, University of Maribor, course Informatics.
	2006 Defence of the undergraduate thesis titled: Securing Web Services with digital signature and encryption, supervisor: izr. prof. dr. Matjaž B. Jurič
	2006 Beginning of the postgraduate studies, study course Computer Science at Faculty of Electrical Engineering and Computer Science in Maribor.
Employment:	2006 General Tax Administration Office, Ljubljana
	2006 - Faculty of Health Sciences, University of Maribor,
	2011 Maribor

Statements



Univerza v Mariboru

Fakulteta za elektrotehniko,
računalništvo in informatiko

IZJAVA O OBJAVI ELEKTRONSKE VERZIJE DOKTORSKE DISERTACIJE IN OSEBNIH PODATKOV, VEZANIH NA ZAKLJUČEK ŠTUDIJA

Ime in priimek doktoranda-ke: Simon Kocbek
Vpisna številka: 95028782
Študijski program: računalništvo in informatika
Naslov doktorskega dela:
Klasifikacija topnosti proteinov v prostoru biomedicinskih konceptov tekstovne
analize

Mentor-ica: prof. dr. Peter Kokol
Somentor-ica: izr. prof. dr. Maria Garcia de la Banda, Monash University

Podpisani soglašam z objavo doktorske disertacije v Digitalni knjižnici Univerze v Mariboru.

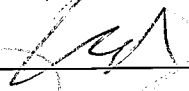
Tiskana verzija doktorske disertacije je istovetna elektronski verziji, ki sem jo oddal-a v Digitalno knjižnico Univerze v Mariboru.

Podpisani-a hkrati izjavljam, da dovoljujem objavo osebnih podatkov, vezanih na zaključek študija (ime, priimek, leto in kraj rojstva, datum diplomiranja, naslov diplomskega dela) na spletnih straneh in v publikacijah Univerze v Mariboru.

Datum in kraj:

Maribor, 7.11.2011

Podpis doktoranda-ke:





Univerza v Mariboru

Fakulteta za elektrotehniko,
računalništvo in informatiko

IZJAVA DOKTORSKEGA KANDIDATA

Podpisani-a Simon Kocbek,

vpisna številka 95028782

izjavljam,

da je doktorska disertacija z naslovom _____
Klasifikacija topnosti proteinov v prostoru biomedicinskih konceptov tekstovne
analize.

- rezultat lastnega raziskovalnega dela,
- da predložena disertacija v celoti ali v delih ni bila predložena za pridobitev kakršnekoli izobrazbe po študijskem programu druge fakultete ali univerze,
- da so rezultati korektno navedeni in
- da nisem kršil-a avtorskih pravic in intelektualne lastnine drugih.

Podpis doktorskega-e kandidata-ke:



Univerza v Mariboru

Fakulteta za elektrotehniko,
računalništvo in informatiko

IZJAVA KANDIDATOVEGA MENTORJA O USTREZNOSTI DOKTORSKE DISERTACIJE

Podpisani-a prof. dr. Peter Kokol, mentor-ica doktorskemu-i kandidatu-ki, izjavljam, da je doktorska disertacija z naslovom

Klasifikacija topnosti proteinov v prostoru biomedicinskih konceptov tesktovne analize.

ki jo je izdelal-a doktorski-a kandidat-ka Simon Kocbek, v skladu z odobreno temo, Pravilnikom o pripravi in zagovoru doktorske disertacije ter mojimi navodili in predstavlja izviren prispevek k razvoju znanstvene discipline.

Datum in kraj:

Maribor, 8.11.2011

Podpis mentorja-ice: