front page.pdf

Acknowledgment.pdf

Dedication.pdf

Content.pdf

introduction.pdf

Ch 1.pdf

Ch 2.pdf

Ch 3.pdf

Ch 4.pdf

Conclusion.pdf

bibliography.pdf

abstract.pdf

DEMOCRATIC AND POPULAR REPUBLIC OF ALGERIA
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

MOHAMED BOUDIAF UNIVERSITY - M'SILA
FACULTY OF MATHEMATICS AND INFORMATICS

DEPARTEMENT OF COMPUTER SCIENCE

Research Paper

Presented for obtaining: MASTER Degree
Field: Mathematics and Informatics
Branch: Computer Science
Specialty: Advanced Information System

By: Ahmed Majam

Topic

Prediction of Protein Structure Classes Using Support Vector Machine (SVM) Classifier

Publically defended on: 01 / 06 / 2016 Before a Jury composed of:

Mr. A. KHATTAF	**University of M'sila**	**Chairman**
Mr. YAGOUBI Rached	**University of M'sila**	**Supervisor**
Mr. R. MOKHTARI	**University of M'sila**	**Examiner**

Class : 2015 /2016

ACKNOWLEDGEMENT

I would like to express my deepest sense of gratitude towards my supervisor, prof MR. YAGOUBI Rached who has given me much suggestion, guidance and support. And thanks to My Prof MOKHTARI Rabeh for his advices, I would like to thank all the staff members of department of computer science for their extended cooperation and guidance. I also take this opportunity to give thanks to all others who have given me support for the project or in other aspects of my study at M'sila University.

DEDICATION

To my lovely family and best friends, and a special thanks to my friend Majid Al-Sharafi

Contents

CHAPTER II : Bioinformatics for prediction of structural classes of proteins

CHAPTER III : Support Vector Machine (SVM)

CHAPTER IV : Results and discusses

GENERAL INTRODUCTION

Living creatures bodies constitute billions of cells, each has a different essential function. Protein is an important component of the cell that itself has many functions.

In this research we will highlight the prediction of protein structures classes for its importance in our life. We highlight a collection of state of the art machine learning algorithm and data processing tools which was named the Weka.

This work is divided into four chapters each one discusses an essential part of the research.

In chapter one, we will talk about molecular biology especially the protein and how it is formed. Also, we will define the term of Bioinformatics, its objectives and the challenges it faces. At the end of this chapter we will talk about biological banks and some formats of them.

In chapter two, we will concentrate on protein structural classification. We will discuss the type of feature extracted. We will present a classification of different methods based on features extracted and their classification algorithmic. Finally, we will present the most used dataset which measure the accuracy.

Chapter three is presenting the SVM (support vector machine) and its importance in our study, and present the different kernel. Then, it highlights the data and its two cases that could be found at.

In chapter four, we will get to know the Weka, what it contains and how we use it. Finally, we will discuss the results which we got through the experiments we are going to make.

Prediction of protein structural classification is widely used in biological laboratories for its huge importance determining protein’s functions.

I.1 Introduction

Quantitation and quantitative tools are indispensable in modern biology. Most biological research involves application of some type of mathematical, statistical, or computational tools to help synthesize recorded data and integrate various types of information in the process of answering a particular biological question. For example, enumeration and statistics are required for assessing everyday laboratory experiments, such as making serial dilutions of a solution or counting bacterial colonies, phage plaques, or trees and animals in the natural environment [1].

In this chapter we will talk about biological molecules, then we will discuss the definition of Bioinformatics, some of its objectives and applications, finally we will talk about biological Bank and some bioinformatics data formats.

I.2 Molecular Biology

Molecular biology is a branch of science concerning biological activity at the molecular level. The field of molecular biology overlaps with biology and chemistry and in particular, genetics and biochemistry. A key area of molecular biology concerns understanding how various cellular systems interact in terms of the way DNA, RNA and protein synthesis function [2].

I.2.1 Cell

All living creatures are made of cells, that is to say the fundamental structural units limited by a membrane [see Figure I.1], containing genetic information and capable of independent living way when they are in a favorable environment [3].

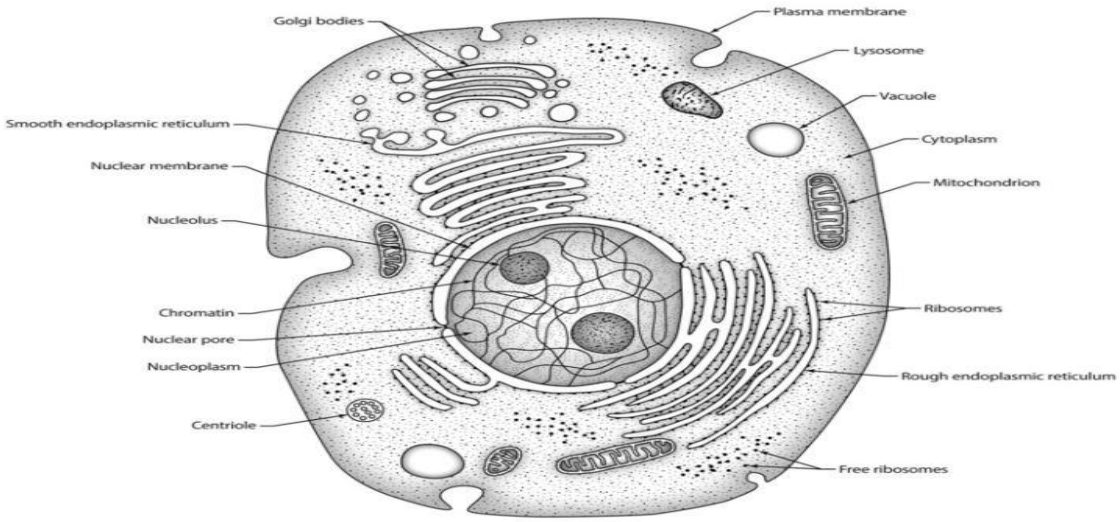

Fig I.1 Animal cell

I.2.1 DNA

The DNA is an acid of the family of nucleic acids, DNA is composed of four basic nitrogen-containing base. Each nucleotide is made up of a phosphate group, a sugar (of the deoxyribose type), and one of the four bases. The four bases are adenine (A), guanine (G), cytosine (C) and thymine (T). The nucleotides can be ranges into two categories. The bases C and T are said pyrimidine, and the bases A and G are called purine. When we speak of DNA, the representation that is made is often cell of a double helix structure [figure I.2] [4]. The two strands constituting this double helix forms are not independently. Each base of one strand is associated on the other strand a Supplemental base [4]. Complementarities bases are the following:

A ↔ T
C ↔ G

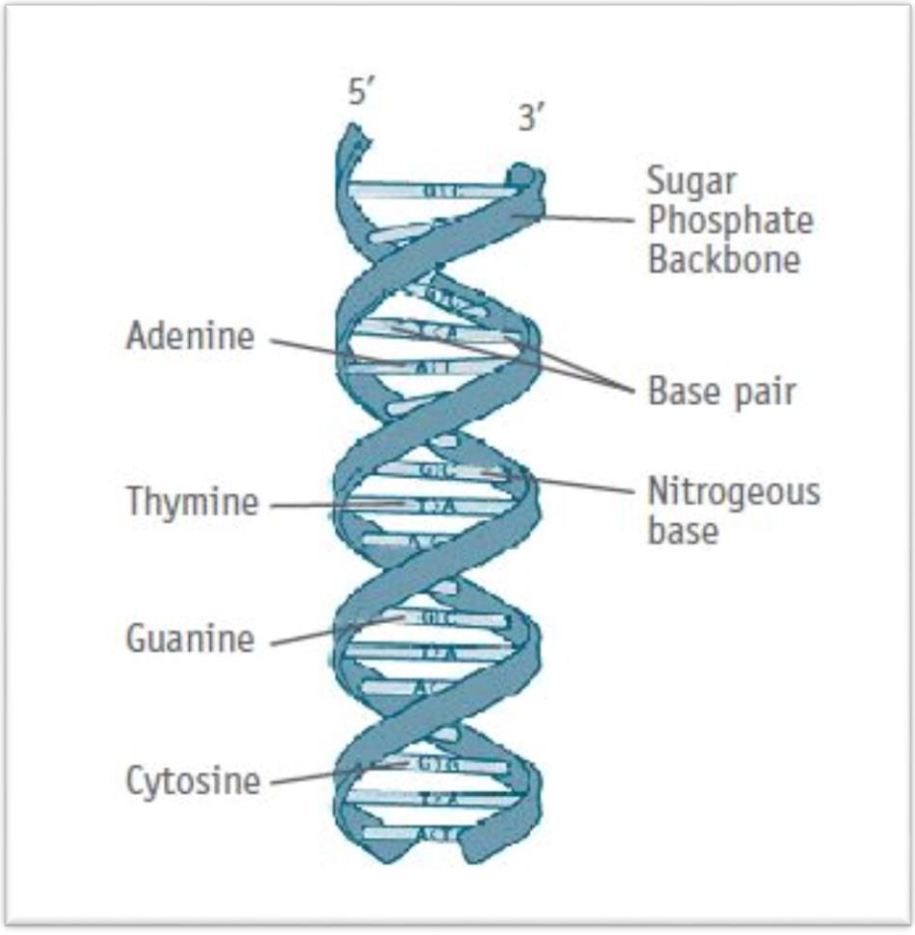

Fig I.2 DNA double helix structure [4]

I.2.2 RNA

RNA is similar in composition to DNA. It is a long linear molecule (polymer) that is made up of a limited number of monomers, the nucleotides. As in DNA, each nucleotide is composed of a sugar, a phosphate, and a base.

The cell contains different kinds of RNA, most importantly messenger RNA (mRNA), transfer RNA (tRNA) and ribosomal RNA (rRNA). These three RNA classes correspond to the three basic roles RNA plays in the cell.

- The Messenger RNA (mRNA) is used in eukaryotes to convey information nucleus to the cytoplasm genetic (substance of the cell that surrounds the nucleus).
- The transfer RNA (tRNA) is used in the RNA translation phase proteins.
- The ribosomal (rRNA) molecules are the major functional components of the molecular machines, the so-called ribosomes, which carry out the translation process [4].

I.2.3 Protein

Proteins are composed of small units. These units are the amino acids which are called the building blocks of protein. There are about 20 different amino acids, the list is given in the table 1. Each different protein is composed of various amino acids put together in varying order with almost limitless combinations. Most proteins are large molecules that may contain several hundred amino acids arranged in branches and chains [6].

Name	Code	Name	Code
Alanine	A	Leucine	L
Arginine	R	Lysine	K
Asparagine	N	Methionine	M
Aspartic acid	D	Phenylalanine	F
Cysteine	C	Proline	P
Glutamic acid	E	Serine	S
Glutamine	Q	Threonine	T
Glycine	G	Tryptophan	W
Histidine	H	Tyrosine	Y
Isoleucine	I	Valine	V

Tab I.1 List of 20 Amino Acids

In the early 1950s, Francis Crick suggested that there is a unidirectional flow of genetic information from DNA through ribonucleic acid (RNA) to protein, i.e. "DNA makes RNA makes protein". This is known as the central dogma of molecular biology [see figure I.3], since it was proposed without much evidence for the individual steps. Now these steps are known in detail: DNA is transcribed to an RNA molecule (messenger RNA [mRNA]), that contains the same sequence information as the template DNA, and subsequently this RNA message is translated into a protein sequence according to the genetic code [4].

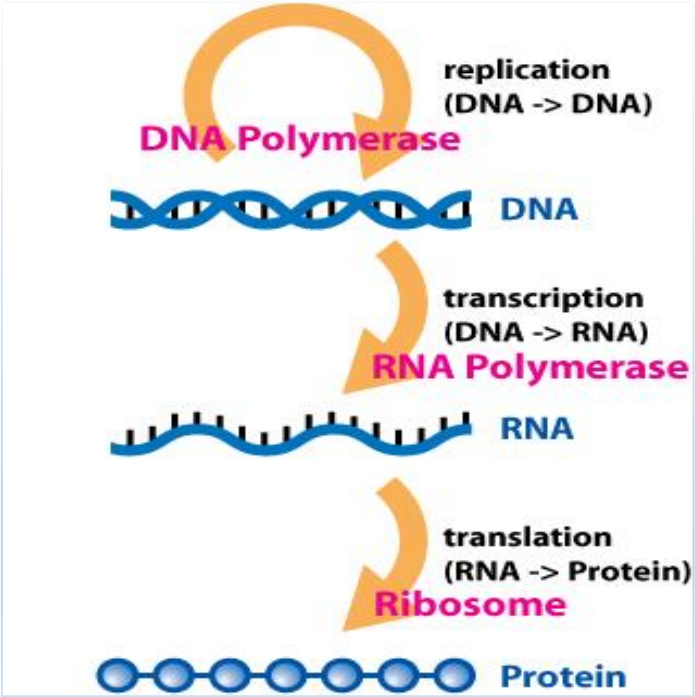

Fig I.3 The Central Dogma

Proteins are macromolecules and have four different levels of structure – primary, secondary, tertiary and quaternary.

1. Primary structure: There are 20 different standard L-α-amino acids used by cells for protein construction [see figure I.4]. Amino acids, as their name indicates, contain both a basic amino group and an acidic carboxyl group.

>1HTB:A|Homo sapiens (human) alcohol dehydrogenase

MSTAGKVIKCKAAVLWEVKKPFSIEDVEVAPPKAYEVRIKMVAVGICRTDD
HVVSGNLVTPLPVILGHEAAGIVESVGEGVTTVKPGDKVIPLFTPQCGKCRV
CKNPESNYCLKNDLGNPRGTLQDGTRRFTCRGKPIHHFLGTSTFSQYTVVD
ENAVAKIDAASPLEKVCLIGCGFSTGYGSAVNVAKVTPGSTCAVFGLGGVG
LSAVMGCKAAGAARIIAVDINKDKFAKAKELGATECINPQDYKKPIQEVLKE
MTDGGVDFSFEVIGRLDTMMASLLCCHEACGTSVIVGVPPASQNLSINPM
LLLTGRTWKGAVYGGFKSKEGIPKLVADFMAKKFSLDALITHVLPFEKINEG
FDLLHSGKSICTVLTF

Fig I.4 Protein Sequence for 1HTB taken from the Protein Data Bank

2. Secondary structure: Within the long protein chains (primary sequence), there are regions in which the chains are organised into regular structures known as alpha helices (α-helices) and beta-pleated sheets (β-sheets). Proteins do not stay in a linear form (primary structure) but ultimately fold into a tertiary structure, which the primary sequence encodes. The tertiary structure is made up of secondary structure elements.
   The two main types of secondary structure elements are α-helices and β-sheets.
   [Figure I.5] is an alpha-helix protein chain, which looks like a loosely coiled spring. The "alpha" means that if you look down the length of the spring, the coiling structure happens in a clockwise direction as it goes away from you. [Figure I.6] shows a beta-pleated sheet protein chain folded so the strands lie alongside each other. Many α-helix elements is called α-helices. Many β-strands make up a β-sheet [13].

Fig I.5 α-helix

Fig I.6 β- sheet segment with two β strands

3. Tertiary structure: The overall three-dimensional shape of an entire protein molecule is the tertiary structure [see Figure I.7]. The protein molecule will bend and twist in such a way as to achieve maximum stability or lowest energy state [6].

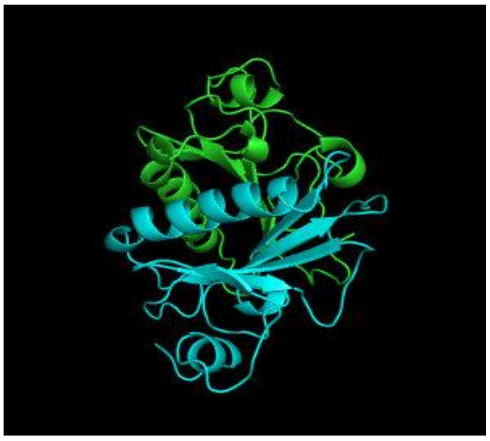

Fig I.7 Visualization of tertiary structure of protein

4. Quaternary structure: Many proteins are made up of multiple polypeptide chains, often referred to as protein subunits [see Figure I.8]. These subunits may be the same (as in a homodimer) or different (as in a heterodimer). The quaternary structure refers to how these protein subunits interact with each other and arrange themselves to form a larger aggregate protein complex [6].

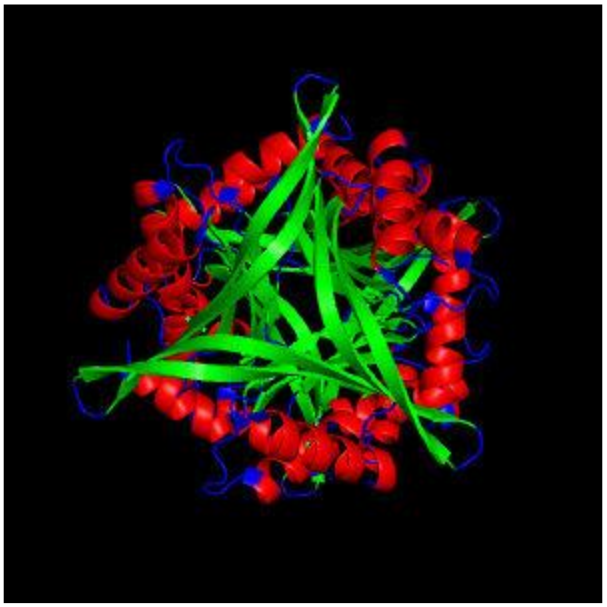

Fig I.8 Visualization of quaternary structure of protein

I.3 Molecular Genetic

I.3.1 The Gene

Historically, a gene is defined as a heritable unit of phenotypic variation. From a molecular standpoint, a gene is the linear DNA sequence required to produce a functional RNA molecule, or a single transcriptional unit. Genes can be assigned to one of two broad functional categories: structural genes and regulatory genes [4].

- Structural genes code for polypeptides or RNAs needed for the normal metabolic activities of the cell, e.g. enzymes, structural proteins, transporters, and receptors, among others.
- Regulatory genes code for proteins whose function is to control the expression of structural genes. With regard to molecular composition both classes of genes are similar.

I.3.2 The Genetic Code

The basic building blocks of DNA are the four nucleotides, the basic building blocks of proteins are the amino acids, of which there are 20 that naturally occur in proteins (the so-called proteinogenic amino acids) [4]. The genetic code is the correspondence between the sequence of the four bases in nucleic acids and the sequence of the 20 amino acids in proteins [4] [7].
The code has many properties:

- There are 64 codons, each of which is a triplet of nucleotide bases.
- Only 20 amino acids are used. These are called the standard amino acids.
- There are some rules, together with some exceptions:

a) XYU and XYC always code the same amino acid.
b) XYA and XYG often code the same amino acid.
c) In 8 out of the 16 possible cases, XY· encodes a single amino acid, where represents any of the four bases.

- The code is nearly universal. That is it seems that the vast majority of living organisms on Earth use this code. This particular code is known as the Canonical Genetic Code [7].

First letter	Second letter: U	Second letter: C	Second letter: A	Second letter: G	Third letter
U	UUU Phe UUC Phe UUA Leu UUG Leu	UCU Ser UCC Ser UCA Ser UCG Ser	UAU Tyr UAC Tyr UAA Stop UAG Stop	UGU Cys UGC Cys UGA Stop UGG Trp	U C A G
C	CUU Leu CUC Leu CUA Leu CUG Leu	CCU Pro CCC Pro CCA Pro CCG Pro	CAU His CAC His CAA Gln CAG Gln	CGU Arg CGC Arg CGA Arg CGG Arg	U C A G
A	AUU Ile AUC Ile AUA Ile AUG Met	ACU Thr ACC Thr ACA Thr ACG Thr	AAU Asn AAC Asn AAA Lys AAG Lys	AGU Ser AGC Ser AGA Arg AGG Arg	U C A G
G	GUU Val GUC Val GUA Val GUG Val	GCU Ala GCC Ala GCA Ala GCG Ala	GAU Asp GAC Asp GAA Glu GAG Glu	GGU Gly GGC Gly GGA Gly GGG Gly	U C A G

Tab I.2 The Genetic Code [7]

I.4 Bioinformatics

I.4.1 What is Bioinformatics

Bioinformatics is an interdisciplinary research area at the interface between computer science and biological science. A variety of definitions exist in the literature and on the World Wide Web, some are more inclusive than others. Here, we adopt the definition proposed by Luscombe et al. in defining bioinformatics as a union of biology and informatics: *bioinformatics* involves the technology that uses computers for storage, retrieval, manipulation, and distribution of information related to biological macromolecules such as DNA, RNA, and proteins. The emphasis here is on the use of computers because most of the tasks in genomic data analysis are highly repetitive or mathematically complex. The use of computers is absolutely indispensable in mining genomes for information gathering and knowledge building. Bioinformatics differs from a related field known as *computational biology*. Bioinformatics is limited to sequence, structural, and functional analysis of genes and genomes and their corresponding products and is often considered *computational molecular biology*. However, computational biology encompasses all biological areas that involve computation. For example, mathematical modeling of ecosystems, population dynamics, application of the game theory in behavioral studies, and phylogenetic construction using fossil records all employ computational

tools, but do not necessarily involve biological macromolecules [1].

I.4.2 Objective of Bioinformatics

The ultimate goal of bioinformatics is to better understand a living cell and how it functions at the molecular level. By analyzing raw molecular sequence and structural data, bioinformatics research can generate new insights and provide a "global" perspective of the cell. The reason that the functions of a cell can be better understood by analyzing sequence data is ultimately because the flow of genetic information is dictated by the "central dogma" of biology in which DNA is transcribed to RNA, which is translated to proteins. Cellular functions are mainly performed by proteins whose capabilities are ultimately determined by their sequences. Therefore, solving functional problems using sequence and sometimes structural approaches has proved to be a fruitful endeavor [1].

I.4.3 Some applications of Bioinformatics

Bioinformatics has not only become essential for basic genomic and molecular biology research, but is having a major impact on many areas of biotechnology and biomedical sciences. It has applications, for example, in knowledge-based drug design, forensic DNA analysis, and agricultural biotechnology. Computational studies of protein–ligand interactions provide a rational basis for the rapid identification of novel leads for synthetic drugs. Knowledge of the three-dimensional structures of proteins allows molecules to be designed that are capable of binding to the receptor site of a target protein with great affinity and specificity [1].

I.4.4 Top Bioinformatics Challenges

- Precise, predictive model of transcription initiation and termination: ability to predict where and when transcription will occur in a genome.
- Prediction of protein classification.
- Precise, predictive model of RNA splicing/alternative splicing: ability to predict the splicing pattern of any primary transcript.
- Precise, quantitative models of signal transduction pathways ability to predict cellular response to external stimuli.
- Determining effective protein-DNA, protein-RNA and protein-protein recognition codes.
- Accurate ab initio structure prediction.
- Rational design of small molecule inhibitors of proteins.

- Mechanistic understanding of protein evolution: understanding exactly how new protein functions evolve.
- Mechanistic understanding of speciation: molecular details of how speciation occurs
- Continued development of effective gene ontologies-systematic ways to describe the functions of any gene or protein.
- (Infrastructure and education challenge).
- Education: development of appropriate bioinformatics curricula for secondary, undergraduate, and graduate education [8].

I.5 Primary and Secondary Database

GenBank is just one member of a community of databases that includes three important protein databases: SWISS-PROT, the Protein Information Resource (PIR), and the Protein DataBank (PDB). SWISS-PROT and PIR can be considered secondary databases, curated databases that add value to what is already present in the primary databases. Both SWISS-PROT and PIR take the majority of their protein sequences from nucleotide databases. A small proportion of SWISS-PROT sequence data is submitted directly or enters through a journal-scanning effort, in which the sequence is (quite literally) taken directly from the published literature. This process, for both SWISSPROT and PIR, has been described in detail elsewhere (Bairoch and Apweiller, 2000; Barker et al., 2000.) As alluded to above, there is an important distinction between primary (archival) and secondary (curated) databases. The most important contribution that the sequence databases make to the scientific community is making the sequences themselves accessible. The primary databases represent experimental results (with some interpretation) but are not a curated review. Curated reviews are found in the secondary databases. GenBank nucleotide sequence records are derived from the sequencing of a biological molecule that exists in a test tube, somewhere in a lab. They do not represent sequences that are a consensus of a population, nor do they represent some other computer-generated string of letters. This framework has consequences in the interpretation of sequence analysis. In most cases, all a researcher will need is a given sequence. Each such DNA and RNA sequence will be annotated to describe the analysis from experimental results that indicate why that sequence was determined in the first place. One common type of annotation on a DNA sequence record is the coding sequence (CDS). A great majority of the protein sequences have not been experimentally determined, which may have downstream implications when analyses are performed. For example, the assignment of a product name or function qualifier based on a subjective interpretation of a similarity analysis can be very useful, but it can sometimes be

misleading. Therefore, the DNA, RNA, or protein sequences are the ‘‘computable’’ items to be analyzed and represent the most valuable component of the primary databases [9].

I.6 Bioinformatics Data Formats

In the field of bioinformatics there exists many different file formats that store DNA and protein sequence information. There is no one sequence format that is ideal: many are used in different contexts, and can often be converted from one to another for easier access or sharing [12].

I.6.1 Fasta Format

The FASTA file format was designed initially for the FASTA searching program, but it is used very widely for other tasks. The FASTA format is typically a sequence centric format in that it provides the sequence with minimal annotations. This fact makes it ideal for processing that requires the sequence and its unique id. Each record in the FASTA file consists of two main compulsory sections: the annotation and sequence sections. Annotation is identified by a line of text beginning with the greater-than symbol (>). The format specifies no other structure limitations on the annotation. However, there are many other pseudo standards for the annotations, such as comma (,) or pipe (|)-separated fields. Usually, annotation up to the first space is the unique identifier, but it is not required. Even the entire annotation can be missing (e.g., only a greater-than symbol is provided to separate the records) [11].

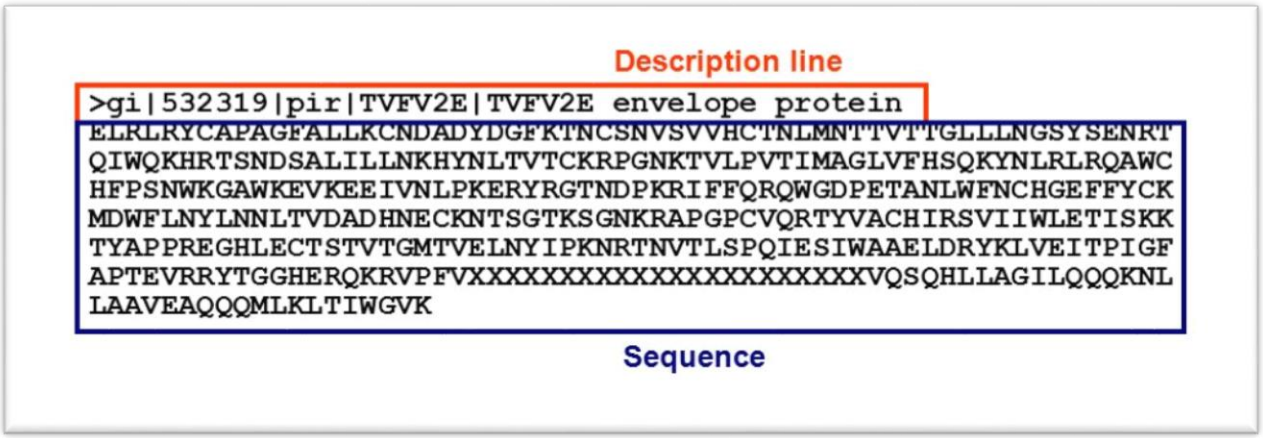

Fig I.5 A sample FASTA sequence record from a sequence DB

I.6.1 Protein DataBank (PDB) Format

Protein Data Bank (PDB) format is a standard for files containing atomic coordinates. Structures deposited in the Protein Data Bank at the Research Collaboratory for Structural Bioinformatics (RCSB) are written in this standardized format. Protein Data Bank format consists of lines of information in a text file. Each line of information in the file is called a *record*. A file generally contains several different types of records, which are arranged in a specific order to describe a structure [10].

Selected Protein Data Bank Record Types	
Record Type	
ATOM	atomic coordinate record containing the x,y,z orthogonal Angstrom coordinates for atoms in standard residues (amino acids and nucleic acids).
HETATM	atomic coordinate record containing the x,y,z orthogonal Angstrom coordinates for atoms in nonstandard residues. Nonstandard residues include inhibitors, cofactors, ions, and solvent. The only functional difference from ATOM records is that HETATM residues are by default not connected to other residues. Note that water residues should be in HETATM records.
TER	indicates the end of a chain of residues. For example, a hemoglobin molecule consists of four subunit chains which are not connected. TER indicates the end of a chain and prevents the display of a connection to the next chain.
SSBOND	defines disulfide bond linkages between cysteine residues.
HELIX	indicates the location and type (right-handed alpha, *etc.*) of helices. One record per helix.
SHEET	indicates the location, sense (anti-parallel, *etc.*) and registration with respect to the previous strand in the sheet (if any) of each strand in the model. One record per strand.

Fig I.6 Selected Protein data bank Record Types [10]

Older PDB files may not adhere completely to the newer format specification. From the standpoint of most users, the most notable differences between older and newer files occur in the fields following the temperature factor in ATOM and HETATM records [10].

I.6.2 GenBank Format

GenBank is the most complete collection of annotated nucleic acid sequence data for almost every organism [1]. Primary protein and nucleic acid sequence databases are so pervasive to our way of thinking in molecular biology that few of us stop to wonder how these ubiquitous tools are built. Understanding how these databases are put together will allow us to move forward in our understanding of biology and in fully harvesting the abstracted information present in these records. GenBank, the National Institutes of Health (NIH) genetic sequence database, is an annotated collection of all publicly available nucleotide and protein sequences. The records within GenBank represent, in most cases, single, contiguous stretches of DNA or

RNA with annotations. GenBank files are grouped into divisions; some of these divisions are phylogenetically based, whereas others are based on the technical approach that was used to generate the sequence information. Presently, all records in GenBank are generated from direct submissions to the DNA sequence databases from the original authors, who volunteer their records to make the data publicly available or do so as part of the publication process. GenBank, which is built by the National Center for Biotechnology Information (NCBI), is part of the International Nucleotide Sequence Database Collaboration, along with its two partners, the DNA Data Bank of Japan (DDBJ, Mishima, Japan) and the European Molecular Biology Laboratory (EMBL) nucleotide database from the European Bioinformatics Institute (EBI, Hinxton, UK). All three centers provide separate points of data submission, yet all three centers exchange this information daily, making the same database (albeit in slightly different format and with different information systems) available to the community at-large [11].

I.6.4 Swiss-Prot Format

An alternative annotation centric format to the GenBank format is the Swiss-Prot format. Swiss-Prot is operated and updated by the Swiss Institute of Bioinformatics (SIB). The Swiss-Prot data repository consists of only amino acid sequences that have been hand controlled for maximum quality and annotation. SIB believes that protein databases need to have three key aspects to measure them by: annotation, minimal redundancy, and integration with other databases. To improve the annotation they try to include as many data about a sequence as possible. To improve the redundancy they combine as many sequences as possible and note any discrepancies. In the past they have combined sequences that are across different source organisms as a method to reduce redundancy. However, they have noticed that this caused confusion in the annotation of species-specific information. So recently, they have separated the sequences that are common over multiple organisms and noted the duplication in the organism species section of the annotation. To improve the integration, they try to link as many other databases as possible. Linking to specialized databases also adds to the level of annotation that the database can contain. The task of annotation and verifying sequences is a rather time-consuming task and thus reduces the size of the database. So to overcome this, the scientists at the Swiss Institute of Bioinformatics (SIB) combined with EMBL to produce an automatic method of annotation. This is stored in another database called TrEMBL and is meant as a supplement to Swiss-Prot. As TrEMBL is a supplement to Swiss-Prot, it contains only sequences that are not already in Swiss-Prot. As the name suggests, the source of TrEMBL sequences, is the European Molecular Biology Laboratory (EMBL) database. To conform to

other databases standards and allow other databases (of the absorb locally type) to interact with the Swiss-Prot/TrEMBL databases, they produce regular updates. They produce four full yearly updates plus weekly difference updates which can be downloaded from their File Transfer Protocol (FTP) site [11].

I.7 Conclusion

We got to know the molecular biology, bioinformatics and the structures of proteins. In next chapter we will discuss the prediction of structural classes of proteins.

II.1 Introduction

Developments in molecular and structural biology during the last three decades, along with the development of large-scale genome technologies and the need to study complex biological systems, have led to the exponential growth and development of biological data produced. Bioinformatics is the application of computing to the organisation and analysis of biological data. The consequence is that computers are being used to collect, store and analyse biological data [13].

II.2 Protein Structural Classification

Structural class definitions were initially developed in 1980s and redefined multiple times since then, (see Table II.1). The main differences were in the thresholds used to define amount of strands for all-α proteins, and amount of helices for all-beta proteins. In 1986 Nakashima and colleagues defined five structural classes [14]. This was followed in 1995 by Chou who proposed classification into again five classes, but using different thresholds [15]. The change was due to Nakashima's classification, which set the thresholds for all-α proteins and all-β proteins that were not large enough to reflect the real features of the two structural classes. Chou also defined content of the secondary structures using the Dictionary of Secondary Structure of Proteins (DSSP) [16]. Another definition, which merges the α+β and the α/β classes into so called mixed class and thus considers only four classes, was proposed by Eisenhaber and colleagues in 1996 [17]. All above classifications consider irregular, which are also called ξ, proteins that are small in numbers and therefore are omitted from classification.

The threshold based classifications were deemed obsolete in the late 1990s and were replaced by the manually performed SCOP classification. SCOP database includes description of the structural and evolutionary relationships of proteins from the Protein Data Bank (PDB) [18]. The SCOP classifies proteins on multiple levels including structural classes, but also as belonging to different families, superfamilies and containing different domains. Domain is defined as a structurally conserved part of a protein sequence, and together with the entire sequences is currently a target of structure prediction. The SCOP's classification does not incorporate hardcoded rules for structural classes. Intuitively, it makes decisions based on structural elements that are located in individual domains that constitute the protein. Researchers claim that the SCOP classification is more "natural" and provides more reliable information to study protein structural classes when compared to classification based on the

percentage amounts of the secondary structures [19,20,21]. The SCOP classification currently includes 11 classes [22]: (1) all-α proteins; (2) all-β proteins; (3) α/β proteins; (4) α+β proteins; (5) multi-domain proteins; (6) membrane and cell surface proteins; (7) small proteins; (8) coiled coils proteins; (9) low resolutions proteins; (10) peptides; and (11) designed proteins. Usually, only the first four categories are considered for computational prediction purposes as they include significant majority of the protein sequences.

Structural class	Helix (α) threshold	Strand (β) threshold
all-α	> 15%	< 10%
all-β	< 15%	< 10%
α+β	> 15%	> 10%
α/β	> 15%	> 10%
all-α	≥ 40%	≤ 5%
all-β	≤ 5%	≥ 40%
α+β	≥ 15%	≥ 15%
α/β	≥ 15%	≥ 15%
all-α	> 15%	< 10%
all-β	< 15%	> 10%
α+β	> 15%	> 10%
all-α	Manually classified (SCOP database)	
all-β		
α+β		
α/β		

Tab II.1 Structural class thresholds [13].

A prior knowledge of structural classes of proteins has become quite useful from both an experimental and theoretical point of view. Knowledge of structural classes of proteins is important in many respects:-

- The knowledge of the structural class of a protein reduces the conformational search space during the search of the tertiary structure as it presents a description of the proteins overall folding process.
- Classification of structural classes of proteins enables the identification of common structural patterns of proteins and it shows that the arrangement of secondary structure elements along the sequence relates to three-dimensional properties of the protein.
- Secondary structure determination from the primary sequence is improved by incorporating knowledge of structural and vice versa, in some studies the predicted

secondary structure information has helped with the classification of structural classes of proteins.

- Reduce the gap between known structural class domains and the unavailability of experimental protein structure information, which is used to assign the structural class for the majority of known protein sequences [13].

The knowledge of the structural classes of proteins is also a useful property applicable to the wider area of proteomics. Such as protein localisation (where it resides within an organism cells) and what type the protein is i.e. enzyme or non-enzyme. Levit and Chothia developed the original concept of protein structural classes. From their work on globular proteins, they saw that protein structures naturally grouped into four main structural classes based on the gross amount of secondary structure elements found in tertiary structures. They devised a system that categorised proteins into one of the following four classes below [13]:

a) All-α: proteins with only small amount of beta-strands (Figure II.1)
b) All-β: proteins with only small amount of alpha-helices (Figure II.2)
c) α/β: proteins that include alpha-helices and beta-strands, where beta-strands are mostly parallel (Figure II.3)
d) α+β: proteins with both alpha-helices and beta-strands, where beta-strands are mostly anti-parallel (Figure II.4)

This classification is based upon majority secondary structure content present in a protein.

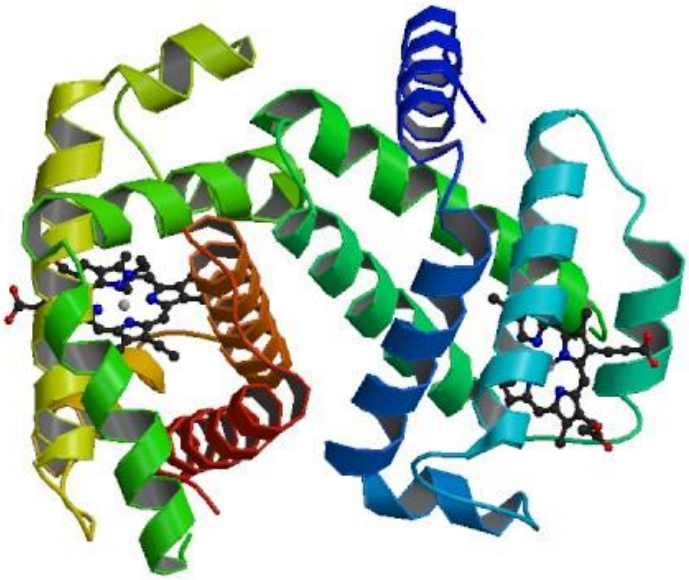

Fig II.1 all-α (Name: Hemoglobin a, PDB id: 2hbc) [23]

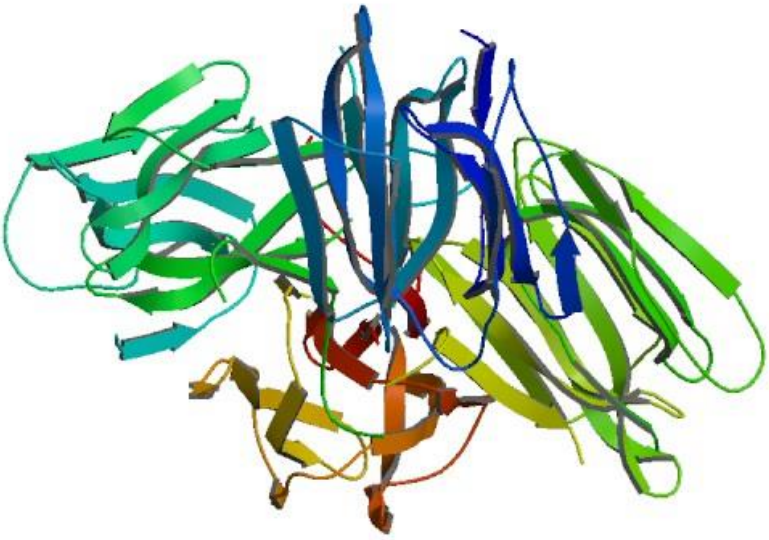

Fig II.2 all-β (Name: jacalin alpha chain, PDB id: 1ku8) [24]

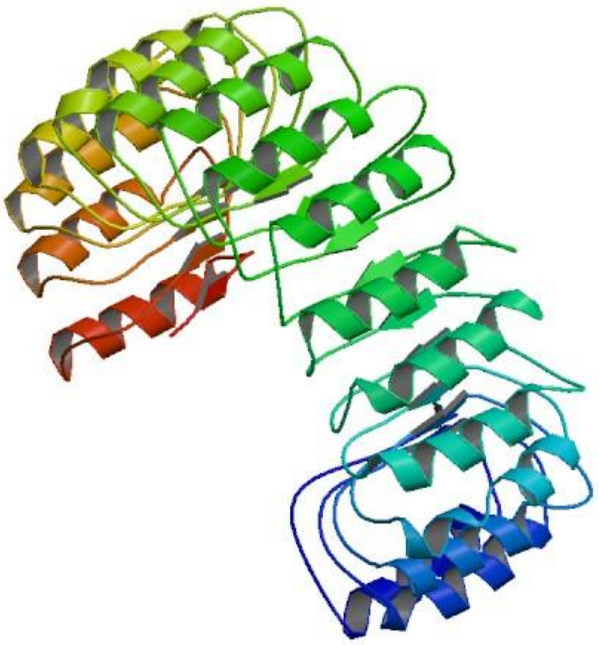

Fig II.3 α/β (Name: Ribonuclease inhibitor, PDB id: 1bnh) [25]

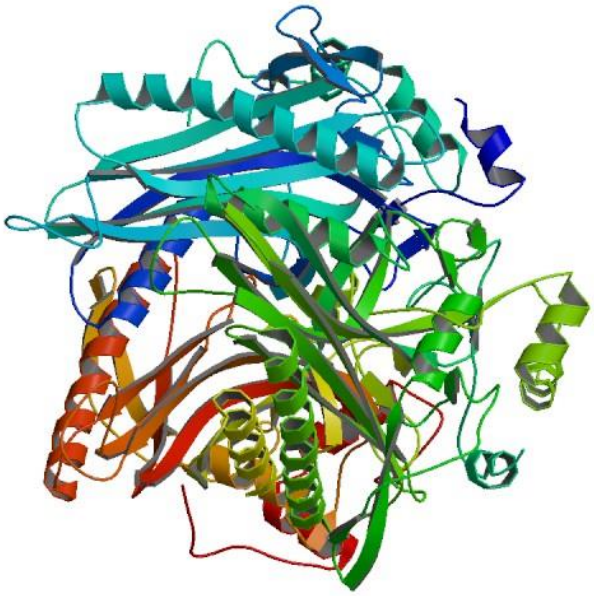

Fig II.4 $\alpha + \beta$ (Name: Pyruvoyl-dependent histidine decarboxylase, PDB id: 1pya) [26]

α-helices and β-strands are the structural elements found in secondary structure composition. The difference between α/β and α+β structural classes of proteins are how the α-helices and β-strands are arranged in the protein. In the α/β structural class, the α-helices commonly separated from the β-strands and alternate more frequently than α+β structural class. Compared to the α+β structural class the β-strands are usually interspersed. Majority of proteins are categorised into these main four classes, however, in addition to the main four groups there are several other structural classes in which very small numbers of proteins exists [13]:-

- Multi-domain proteins (alpha and beta) - folds consisting of two or more domains which belong to different structural classes.
- Membrane and cell surface proteins.
- Small proteins.
- Coiled coil proteins.
- Low resolution protein structures.
- Peptides.
- Designed proteins.

How does protein structure determine function? The tertiary structure of a protein infers the biological function of the protein. An analogy to protein structure linked to protein function is the key and lock, the correct key unlocks the correct door; the correct shape (structure) allows the inferring of function. Knowing the structural classes of proteins is an important aspect in tertiary structure classification as the structural class of a protein presents an intuitive description of its overall folding process. With many millions of conformations a protein can fold into the restriction the structural class imposes is a reduction in the search space to find the single conformation state the protein folds into and has high impact on its tertiary structure classification.

Several studies have proposed definitions of structural class thresholds; these have been revised a few times since the conception of structural classes of protein back in the 1970's. (Table II.1) contains thresholds of α and β structural elements, which were used to determine which structural class a proteins is categorised into. The difference between each system is the amount of helices found in all-β proteins and the amount of strands found in all-α protein.

However, the thresholds system has now been replaced by a manual classification method named the Structural Classification of Proteins (SCOP), which is a manual method replacing

the various thresholds as it became too rigid and other times too relaxed to categorise proteins into appropriate structural classes. The assignment of the structural class of proteins is now performed manually. The SCOP databases stores the structural classes of proteins that have been manually verified. SCOP has become one of the main data resources used in protein structural class research. The SCOP database has manually determined the structural class of many of the proteins from the PDB that have been experimentally verified. The PDB uses the assignment of structural classes of proteins done by SCOP into its own databank, not all proteins from the PDB are assigned a structural class yet, as SCOP is still going through PDB data and assigning each experimentally verified protein a structural class. Table II.2 contains the number of PDB and SCOP entries as of February 2013; the column named "No. of proteins…" contains the number proteins that have been experimentally classified by PDB and that have had a structural class assigned by SCOP. The protein datasets that will be used for the thesis analyses will come from the PDB and SCOP databases as they have experimentally determined protein structures where its structural classes are known [13].

Structural class	No. of proteins where class information is known
All alpha proteins	7627
All beta proteins	10672
Alpha and beta proteins (α/β)	11965
Alpha and beta proteins (α+β)	11053
Small proteins	2282
Multi-domain proteins (alpha and beta)	1199
Peptides	773
Other	1592
Total	47163

Tab II.2 Classification of proteins into structural classes [13]

The ASTRAL compendium for sequence and structure analysis provides databases and tools used in the research and analysis of protein structures and their sequences, the sequences partly derived from SCOP database. The other part of the data is derived from the coordinate files maintained and distributed by the PDB. The ASTRAL compendium is useful to extract large number of protein's amino acid sequences whose structural classes have been determined experimentally already. These sequences can form datasets to be used in bioinformatics studies [13].

II.3 Related works

Prediction of the protein structural classes is usually performed as a two steps procedure. First, sequences of different length are represented by a fixed length feature vector and next the feature values are fed into a classification algorithm. [67]. We will present a classification of different methods according to their extracted features and the classification algorithms.

II.3.1 Features Extracted From Amino Acid Sequence of Proteins

The earliest methods of classification of proteins used only features extracted directly from the amino acid sequences. In those methods, the researchers established a correlation between amino acid sequences and the corresponding structural classes. Zerin et al. [27] used the occurrence frequency of each of 20 amino acid and all possible combination of three consecutive amino acids known as triplets in the protein sequence. Using support vector machine classification algorithm their method achieved 71.4% prediction accuracy. Subsequent research showed that information related to only amino acid composition, such as the frequency of each amino acid or peptide, may have limited prediction ability, since the folding pattern of a proteins is the result of collective interaction among the residues in protein sequence [28]. To improve the accuracy of predictions, features representing amino acid position and order were introduced in the later research. Wu et al. [29] combined amino acid word frequency, word position and physiochemical properties of amino acid to represent proteins, where a word is a short sequence of amino acids of length *n* also referred to as "n-gram" pattern. They calculated the position information of amino acids based on the concept of measuring inter-nucleotide distances as described in [30-31]. They transformed the amino acid sequence into a numerical sequence which contains position information of each element. For each of the 20 amino acids they used the interval distance between the two nearest positions of that amino acid and calculated the probability of occurrence of that amino acid at that interval. They also calculated the 1-word frequency (frequency of word with length "1") of hydropathy states in the sequences after 17 transforming amino acid sequence of protein to hydropathy sequence, based on the hydropathy profile of amino acids.

Zhang et al. [32] constructed a 46 dimensional feature vector, where 20 values represent amino acid frequency, 20 values represent amino acid correlation at various distances, and 6 values represent frequency of hydrophobic amino acid couples. The calculations of the amino acid distance correlation are described by Equation (1) - (5) of [32]. Hydrophobic amino acids are those that avoid interaction with water. The distance correlation is relevant because amino

acids which are far apart in the sequence may be close neighbours after folding. They used the support vector machine algorithm based on a binary tree as described in [33]. Ding et al. [34] used the concept of pseudo amino acid composition (PseAA) introduced by Chou [35] to incorporate information about the order of amino acid residues in proteins as features. They used eight physiochemical properties like volume, polarity, and hydrophobic value to construct eight PseAA vectors to represent each protein. Each of these eight vectors was a 40 dimensional vector, where 20 values were the frequency of the 20 amino acids and 20 values were the correlation values between k-tier contiguous residues. Using each of these physiochemical property, they used Equations (2)-(6) of [35] to generate correlation values between k-tier (k=1,...,20) contiguous residues in the protein chain. The difference between Ding et al. [34] and chou's [35] method is that Ding et al. used eight different physiochemical property to generate eight PseAA vectors, whereas Chou [35] constructed only one PseAA vector using three physiochemical property values. For multiclass classification, Ding et al. [34] used dual layer fuzzy support vector machine (FSVM) as establish by Abe [36]. For each protein sample, eight PseAA vectors were fed into eight FSVM in the first layer. Outputs of the first layer generated by eight FSVM classifiers were again reclassified in the second layer. The high accuracy was achieved on dataset constructed by Chou [37]. There are a few more reports [38-40] based on extracting features from amino acid sequence of proteins [66].

II.3.2 Features Extracted from PSI-BLAST Profiles of Sequence

PSI-BLAST [41] (Position-Specific Iterative Basic Local Alignment Search Tool) profiles of sequences have also been used in structural class prediction methods as they reflect the evolutionary relationship among sequences [42-43]. PSI-BLAST [41] generates a position specific scoring matrix (PSSM) or profile from multiple sequence alignment which reflects how closely a query sequence is to the database of collected sequences. Taigang et al. [42] transformed the PSSM generated by PSI-BLAST into a fixed length feature vector by auto covariance (AC) transformation. They used AC transformation as it is a powerful statistical tool for analyzing sequence vectors in other areas of bioinformatics [44-47]. Their model, using a combination of PSSM and the AC method, showed good performance while reflecting evolutionary information and sequence order information at the same time.

II.3.3 Features Based on Functional Domains of Sequence

Functional domains are the regions in an amino acid sequence of protein that carry out a specific function. Proteins typically have several functional domains. Using these functional

domains as features in the structural class prediction problem, some researchers tried to capture the relationship among distant amino acids which is crucial for protein folding [48-49]. Chou et al. [48] used an integrated domain database [50] (InterPro database) which contains many sequences along with functional domain information. InterPro release 6.2 documents 7785 different functional domains (http://www.ebi.ac.uk/interpro). Chou et al. [48] represented each protein as a 7785 dimensional vector, where each feature is Boolean. A "1" represents the presence of a particular functional domain, and a "0" represents the absence of that functional domain in a protein. They suggest that functional domains of a protein correlate well with its structural class.

Amin et al. [49] followed Chou et al. [48] and used functional domains as class discriminating features. They used InterPro Release 30.0 which contains 21,178 functional domain entries. Of the 21,178 functional domains they only considered the domains which appear in the proteins of their dataset. Thus, their method used 2,400 functional domains as features. They also extracted features from predicted protein secondary structure. To reduce the dimension of the feature vector and select the most effective features they used the correlation based feature selection (CFS) method [54]. CFS is a filtering method to select from the original feature set a smaller set of non-redundant features which have powerful class discriminating ability [66].

II.3.4 Features Extracted from Predicted Protein Secondary Structure Sequence

Recently, many good methods have been developed using only features extracted from predicted protein secondary structure sequence [52-54]. The structural class of a protein mainly depends on its secondary structural content. Some researchers extracted features from predicted secondary structure sequence instead of amino acid sequence of protein. In these methods the researchers used secondary structure sequences predicted by methods like PSIPRED [55] and YASPIN [56]. Liu and Jia [52] constructed three novel features to differentiate between the $\alpha+\beta$ class and the α/β class more accurately. They used some previously used effective features from research [57- 58] like content of α-helix (H) and β-strand (E) in the sequence, maximum and average length of H and E segments, and composition moment vector of H and E in the sequence. One would expect the predicted states H and E to alternate more frequently in a protein belonging to the α/β class than in a protein belonging to the $\alpha+\beta$ class where α-helix and β-strands are isolated. Therefore, one of their newly developed features was the normalized alternating frequency of H and E. They also included two newly developed features based on

count of anti-parallel β-strands in the sequence, considering the fact that β-sheets in the α/β class proteins are usually composed of parallel β-strands whereas in the $\alpha+\beta$ class proteins, β-sheets are normally composed of antiparallel β-strands. They also showed that their newly constructed features had good impact in identifying the α/β and the $\alpha+\beta$ class proteins.

Along with some previously used features from [52,58], Zhang et al. [53] introduced some new features to capture the distribution of α-helix (H) and β-strand (E) in the sequence. They made a reduced representation of sequences using only H and E, while ignoring Coil (C). Then, using a transition probability matrix, they computed features based on probability of transition from H to E and E to H. They showed that their newly developed features based on transition probability matrix made good contribution in the overall. Ding et al. [54] also constructed several new features to extract information from predicted secondary structure sequences, such as the following: the variance of the length of H and E segments; variance of the positions of H and E in the secondary structure sequence; average length of H and E segments in the sequences, while ignoring coil segments. They also showed that their newly designed features are good for predicting the $\alpha+\beta$ and α/β class proteins compared with some established methods.

II.3.5 Features from both Amino Acid Sequence and Predicted Secondary Structure Sequence

Some successful methods used both amino acid sequences and predicted secondary structure sequences to extract features [43,49,58,59]. These methods try to incorporate useful class discriminating information from both amino acid sequence and predicted secondary structure sequence of protein. In 2008, Kurgan et al. [58] proposed a structural class prediction method popularly known as SCPRED. For this research, initially they extracted 2146 features from the amino acid sequence. These 2146 features included physiochemical values of amino acid based features, amino acid component based features like 1st and 2nd order composition moment vector, and property groups based features. The 20 amino acids can be subdivided into groups based on any one of several physiochemical properties. For example, according to electronic property, 20 amino acids are classified into following five groups: electron donor, weak electron donor, electron acceptor, weak electron accepter and neutral [58]. Features like composition percentage of electronic groups of amino acids, composition percentage of hydrophobic groups of amino acids were calculated. They also extracted 176 features from predicted protein secondary structure sequences which include maximum and average length of secondary structure segment, composition moment vector of secondary structural state. They reduced the

dimension of their initial feature set from 2322 to 9 by using Hall's [51] correlation based feature selection method. The algorithm chose 8 features from secondary structure sequence and 1 from amino acid sequence, confirming the class discriminating quality of features extracted from secondary structure sequences [66].

Mohammad and Hampapathalu [59] extracted features from secondary structure sequences, but also considered the solvent accessibility information of amino acid residues and residue pairs in the amino acid sequence. They used features like frequency of each amino acid to occur in a particular secondary structural states (H, E or C). Frequencies of amino acids pairs predicted as secondary structural state H, E and C were also measured for their model. They calculated the solvent accessibility state information for each amino acid in the protein from ACCpro [60]. Solvent accessibility of an amino acid residue is described as a binary value, either buried or exposed in terms of the degree of its interaction with the water molecules. They calculated features like frequency of 'buried' or 'exposed' residues. They also calculated the frequencies of amino acid pairs having solvent accessibility state 'buried', 'exposed' or 'partially buried'. Here they considered an amino acid pair to be in 'buried' or 'exposed' state only if both the residues were predicted in 'buried' or 'exposed' state, respectively, otherwise the pair was considered as in 'partially buried' state. Finally they checked their prediction model with different individual feature set and with the combination of these feature sets. They successfully showed that using information from both protein sequence and predicted secondary structure sequence could give better prediction than some other contemporary methods like SCPRED [58] and Kurgan and Chen [61].

Amin et al. [49] followed Kurgan et al. [58] and Yang et al. [62] to extract features from predicted secondary structure sequence of protein. Along with functional domain features they checked the contribution of features from secondary structure sequence in predicting structural classes. They used the CFS method to select the effective class discriminating features from initial feature set, resulting in only 77 functional domain features from the initial 2400 features, and 34 secondary structural features from the initial 110 features. This study too confirmed the effectiveness of secondary structural features in solving this problem [66].

Mizianty and Kurgan [43] used features based on the PSI-BLAST profile of proteins along with features from amino acid sequence and predicted secondary structure sequence of protein. After checking a combination of feature sets, they found that a combination of features from

PSIBLAST and predicted secondary structure sequence gave the best class discriminating performance for a twilight zone dataset.

II.3.6 Classification Algorithm

Soft computing techniques like Support Vector Machine (SVM), Artificial Neural Network (ANN) and Fuzzy Logic (FL) are machine learning techniques often used to construct classification models for protein structural class prediction. The strength of soft computing techniques to give human like expert decisions and to handle ambiguous and uncertain situations as well as to process approximate and flexible information with low solution cost make them an effective choice to be used by bioinformatics researchers who need to deal with a large amount of faulty uncertain data [66].

Support Vector Machine seems to be the most popular among all soft computing techniques to solve the protein structural classification problem as it has been used in many projects [27,29,42-43,49,52-54,58-59]. The basic idea behind this supervised machine learning method is to create a hyperplane which not only separate but also maximizes the distance between two classes. It then assigns the prediction label according to on which side of this hyperplane a test case falls. SVM solves the multiclass problem by creating either one-against-one binary classifiers or one-against-all binary classifiers. Amin et al. [49], Kurgan et al. [58] and Mohammad and Hampapathalu [59] developed one-against-one binary classifiers. Amin et al. used the initial predictions by binary classifiers to predict final class labels by pair-wise coupling technique as presented by [61]. Mohammad and Hampapathalu [59] used a voting scheme to assign the most probable class to the query protein sequence. Wu et al. [29] constructed one-against-all binary classifiers. Some methods combined SVM with other techniques to get a more efficient result [32] [34]. Zhang et al. [32] developed SVM based on binary tree methods whereas Ding et al. [34] combined fuzzy logic with SVM.

Some methods used Artificial Neural Networks (ANN) to solve the prediction problem [64-65]. They chose ANN due to its self-organizing and self-adaptability properties. After learning and training on representative proteins, it refers the relevant features of proteins, and then it can assign a query protein to a specific structural class.

Chou et al. [48] used an intricate sorting method to do class prediction, where a similarity score was calculated between the query protein and all proteins in the dataset. The query protein is assigned to the class of a protein in the dataset which is most similar to it as in nearest neighbour approaches.

II.4 Dataset resources

Data resources protein data bank (PDB) and structural classification of proteins (SCOP) are the main resources where protein amino acid sequences are obtained and constructed. These datasets contain the collection of protein amino acid sequences used to train and test predictive models. The datasets used or developed must be representative of the prediction of protein structural classes, taking into consideration of the homology level and sample sizes of each structural class as this impacts on predictive accuracy [13]. There have been many datasets developed for the prediction of protein structural classes, which are mainly derived from SCOP database.

The PDB database contains information about experimentally determined structures of proteins; it is the major resource for fully annotated proteins including structural class information, which is derived from the Structural Classification of Proteins (SCOP) database. Popular datasets constructed using PDB are listed in Table II.3.

SCOP is a manually annotated database and has been regarded as the most accurate classification of structural classes of proteins. The SCOP database contains proteins that are manually curated, annotated and classified into the structural classes providing information about folds evolutionary relationships, which can then be used in numerous protein structure related studies [13]. The current version of the SCOP database, v. 1.75, includes eleven structural classes, with the four major classes (all-α, all-β, α/β and α+β) covering approximately 90% of the entries in PDB [13].

Dataset Name	Structural Class			
	All-α	All-β	α/β	α+β
1189	223	294	334	241
25PDB	443	443	346	441

Tab II.3 Datasets constructed using PDB and SCOP [13]

II.5 Conclusion

In this chapter we talked about protein structural classification, its importance in our life, and we review the four structural classes. We discussed the types of feature extracted. We used the SVM to classify the protein structure.

In the next chapter, we will extend the SVM.

III.1 Introduction

In machine learning, support vector machines are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis [67]. SVMs are a useful technique for data classification. Although SVM is considered easier to use than Neural Networks, users not familiar with it often get unsatisfactory results at first.

Although users do not need to understand the underlying theory behind SVM, we briefly introduce the basics necessary for explaining our procedure. A classification task usually involves separating data into training and testing sets [68].

III.2 The Linearly Separable case

To explain the mystery of SVMs, let's first look at the simplest case a two-class problem where the classes are linearly separable. Let the data set D be given as *(X1, y1), (X2, y2),…, ($X_{|D|}$, $y_{|D|}$),*where Xi is the set of training tuples with associated class labels, yi. Each yi can take one of two values, either+1 or -1 (i.e., $yi \in \{+1, -1\}$), corresponding to the classes buys_computer = yes and buys_computer = no, respectively. To aid in visualization, let's consider an example based on two input attributes, A1 and A2, as shown in Figure III.1. From the graph, we see that the 2-D data are linearly separable (or "linear," for short) because a straight line can be drawn to separate all of the tuples of class +1 from all of the tuples of class -1. There are an infinite number of separating lines that could be drawn. We want to find the "best" one, that is, one that (we hope) will have the minimum classification error on previously unseen tuples. How can we find this best line? Note that if our data were 3-D (i.e., with three attributes), we would want to find the best separating plane. Generalizing to n dimensions, we want to find the best hyperplane. We will use the term "hyperplane" to refer to the decision boundary that we are seeking, regardless of the number of input attributes. So, in other words, how can we find the best hyperplane? An SVM approaches this problem by searching for the maximum marginal hyperplane. Consider Figure III.2, which shows two possible separating hyperplanes and their associated margins [68].

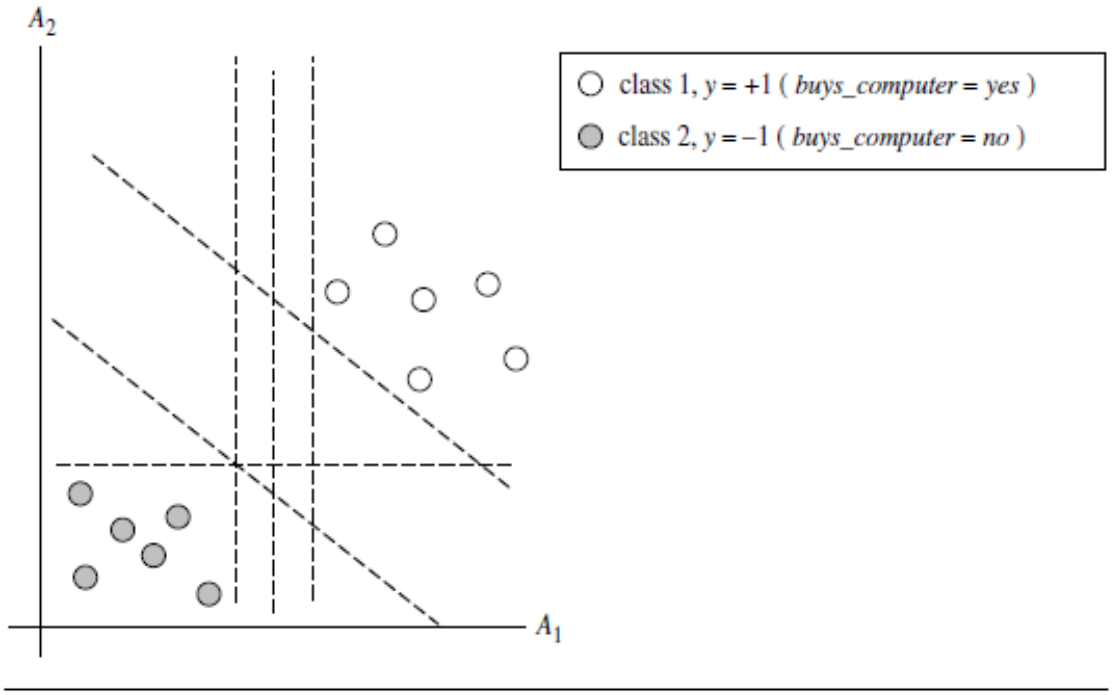

Fig III.1 The 2-D training data are linearly separable.

Both hyperplanes can correctly classify all of the given data tuples. Intuitively, however, we expect the hyperplane with the larger margin to be more accurate at classifying future data tuples than the hyperplane with the smaller margin. This is why (during the learning or training phase), the SVM searches for the hyperplane with the largest margin, that is, the maximum marginal hyperplane (MMH). The associated margin gives the largest separation between classes. Getting to an informal definition of margin, we can say that the shortest distance from a hyperplane to one side of its margin is equal to the shortest distance from the hyperplane to the other side of its margin, where the "sides" of the margin are parallel to the hyperplane. When dealing with the MMH, this distance is, in fact, the shortest distance from the MMH to the closest training tuple of either class [68].

A separating hyperplane can be written as:

$$W.\,X + b = 0 \qquad \text{(III.1)}$$

Where W is a weight vector, namely, $W = \{w_1, w_2, \ldots, w_n\}$; n is the number of attributes; and b is a scalar, often referred to as a bias. To aid in visualization, let's consider two input attributes, A_1 *and* A_2, as in Figure III.2 (b). Training tuples are 2-D, e.g., $X = (x_1, x_2)$, where x_1 *and* x_2 are

the values of attributes A_1 *and* A_2, respectively, for X. If we think of b as an additional weight, w_0, we can rewrite the above separating hyperplane as

$$w_0+w_1x_1+w_2x_2 = 0. \qquad \text{(III.2)}$$

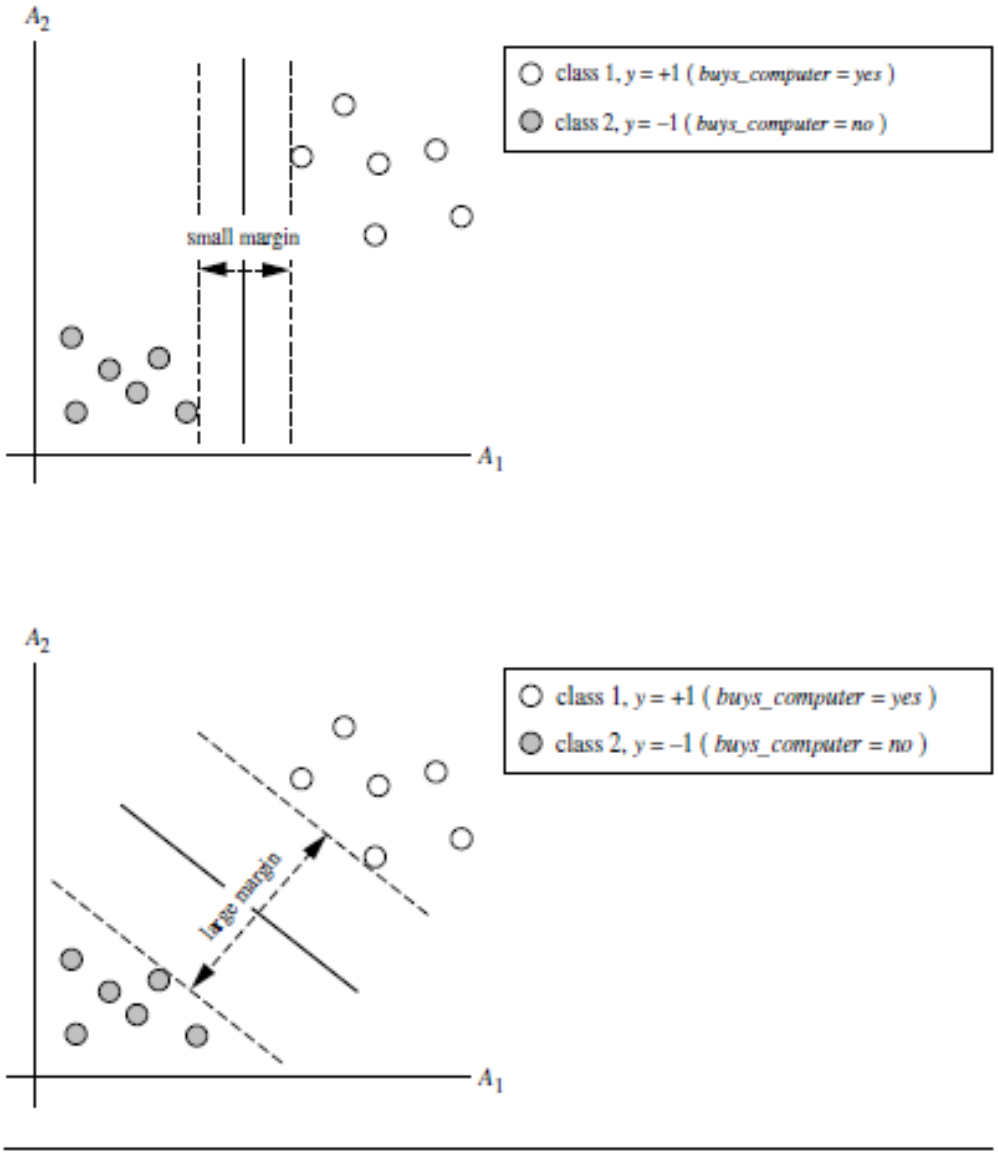

Fig III.2 Two possible separating hyperplanes and their associated margins.

The one with the larger margin should have greater generalization accuracy.

Thus, any point that lies above the separating hyperplane satisfies

$$w_0 + w_1x_1 + w_2x_2 > 0. \qquad \text{(III.3)}$$

Similarly, any point that lies below the separating hyperplane satisfies

$$w_0 + w_1x_1 + w_2x_2 < 0. \qquad \text{(III.4)}$$

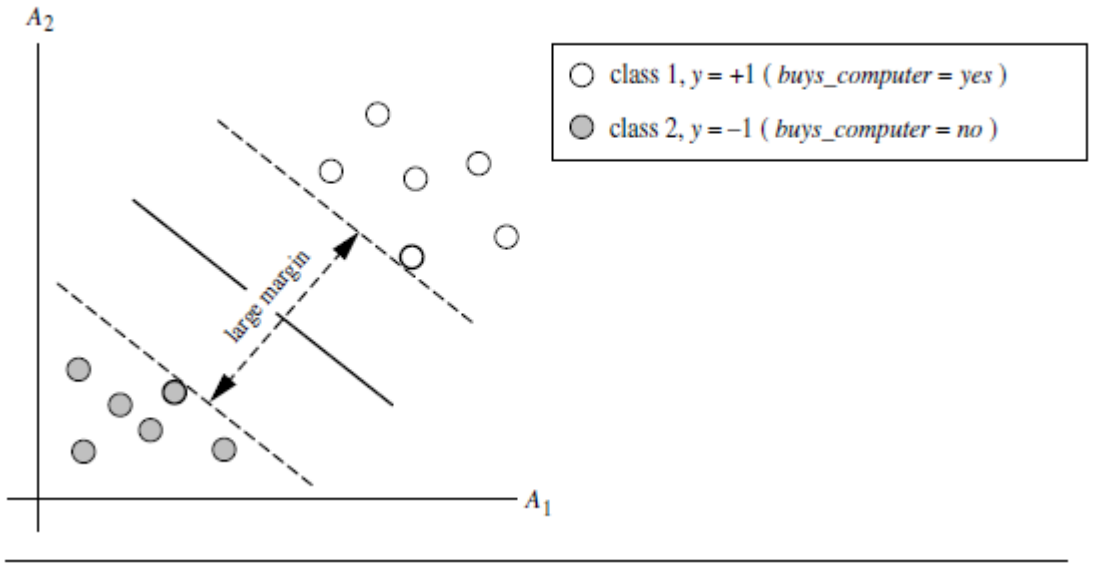

Fig III.3 The SVM finds the maximum separating hyperplane.

The one with maximum distance between the nearest training tuples. The support vectors are shown with a thicker border.

The weights can be adjusted so that the hyperplanes defining the "sides" of the margin can be written as:

$$H1 : w_0+w_1x_1+w_2x_2 \geq 1 \text{ for } yi = +1 \qquad \text{(III.5)}$$

$$H2 : w_0+w_1x_1+w_2x_2 \leq -1 \text{ for } yi = -1: \qquad \text{(III.6)}$$

That is, any tuple that falls on or above H_1 belongs to class +1, and any tuple that falls on or below H_2 belongs to class -1. Combining the two inequalities of Equations (5) and (6), we get:

$$yi\,(w_0 + w_1x_1 + w_2x_2) \geq 1, \ \forall i. \qquad \text{(III.7)}$$

Any training tuples that fall on hyperplanes *H1* or *H2* (i.e., the "sides" defining the margin) satisfy Equation (7) and are called support vectors. That is, they are equally close to the (separating) MMH. In Figure III.2, the support vectors are shown encircled with a thicker border. Essentially, the support vectors are the most difficult tuples to classify and give the most information regarding classification.

From the above, we can obtain a formulae for the size of the maximal margin. The distance from the separating hyperplane to any point on *H1* is $\frac{1}{||W||}$, where $||W||$ is the Euclidean norm of *W*, that is $\sqrt{W.W}$. By definition, this is equal to the distance from any point on *H2* to the separating hyperplane. Therefore, the maximal margin is $\frac{2}{||W||}$.

"So, how does an SVM find the MMH and the support vectors? "Using some "fancy math tricks," we can rewrite Equation (III.7) so that it becomes what is known as a constrained (convex) quadratic optimization problem. Once we've found the support vectors and MMH (note that the support vectors define the MMH!), we have a trained support vector machine. The MMH is a linear class boundary, and so the corresponding SVM can be used to classify linearly separable data. We refer to such a trained SVM as a linear SVM [68].

Based on the Lagrangian formulation mentioned above, the MMH can be rewritten as the decision boundary

$$d(X^T) = \sum_{i=1}^{l} yi\ \alpha i\ Xi\ \ X^T + b_0 \qquad \text{(III.8)}$$

Where y_i is the class label of support vector $\boldsymbol{X}_i$; X^T is a test tuple; α_i and b_0 are numeric parameters that were determined automatically by the optimization or SVM algorithm above; and l is the number of support vectors.

Interested readers may note that the α_i are Lagrangian multipliers. For linearly separable data, the support vectors are a subset of the actual training tuples (although there will be a slight twist regarding this when dealing with nonlinearly separable data).

Given a test tuple, X^T, we plug it into Equation (III.8), and then check to see the sign of the result. This tells us on which side of the hyperplane the test tuple falls. If the sign is positive, then X^T falls on or above the MMH, and so the SVM predicts that X^T belongs to class +1 (representing buys computer = yes, in our case). If the sign is negative, then X^T falls on or below the MMH and the class prediction is -1 (representing buys computer = no). Notice that the Lagrangian formulation of our problem (Equation (III.8)) contains a dot product between support vector *Xi* and test tuple X^T. This will prove very useful for finding the MMH and support vectors for the case when the given data are nonlinearly separable [68].

Before we move on to the nonlinear case, there are two more important things to note. The complexity of the learned classifier is characterized by the number of support vectors rather than the dimensionality of the data. Hence, SVMs tend to be less prone to overfitting than some

other methods. The support vectors are the essential or critical training tuples—they lie closest to the decision boundary (MMH). If all other training tuples were removed and training were repeated, the same separating hyperplane would be found. Furthermore, the number of support vectors found can be used to compute an (upper) bound on the expected error rate of the SVM classifier, which is independent of the data dimensionality. An SVM with a small number of support vectors can have good generalization, even when the dimensionality of the data is high.

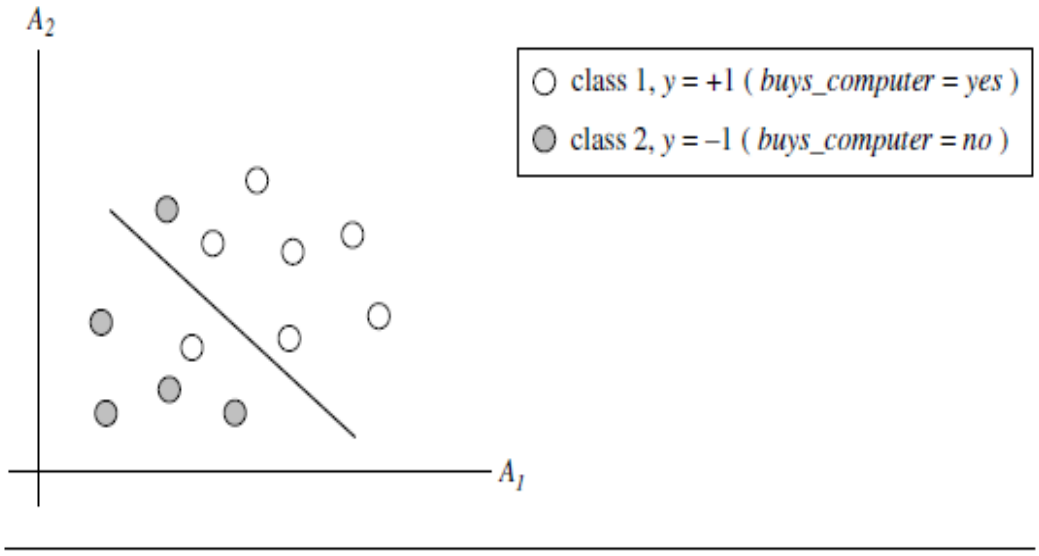

Fig III.4 simple 2-D case showing linearly inseparable data.

Unlike the linear separable data of Figure III.1, here it is not possible to draw a straight line to separate the classes. Instead, the decision boundary is nonlinear.

III.3 The Linearly Non-separable case

In the previously section, we learned about linear SVMs for classifying linearly separable data, but what if the data are not linearly separable, as in Figure III.4? In such cases, no straight line can be found that would separate the classes. The linear SVMs we studied would not be able to find a feasible solution here.

The good news is that the approach described for linear SVMs can be extended to create nonlinear SVMs for the classification of linearly inseparable data (also called nonlinearly separable data, or nonlinear data, for short). Such SVMs are capable of finding nonlinear decision boundaries (i.e., nonlinear hypersurfaces) in input space.

We obtain a nonlinear SVM by extending the approach for linear SVMs as follows. There are two main steps. In the first step, we transform the original input data into a higher dimensional space using a nonlinear mapping. Several common nonlinear mappings can be used in this step, as we will describe further below. Once the data have been transformed into the new higher space, the second step searches for a linear separating hyperplane in the new space. We again end up with a quadratic optimization problem that can be solved using the linear SVM formulation. The maximal marginal hyperplane found in the new space corresponds to a nonlinear separating hypersurface in the original space [68].

Example:- nonlinear transformation of original input data into a higher dimensional space. Consider the following example. A 3D input vector $X = (x_1, x_2, x_3)$ is mapped into a 6D space, Z, using the mappings $\phi_1(X) = x_1$, $\phi_2(X) = x_2$, $\phi_3(X) = x_3$, $\phi_4(X) = (x_1)^2$, $\phi_5(X) = x_1 x_2$, *and* $\phi_6(X) = x_1 x_3$. A decision hyperplane in the new space is $d(Z) = WZ + b$, where W and Z are vectors. This is linear. We solve for W and b and then substitute back so that the linear decision hyperplane in the new (Z) space corresponds to a nonlinear second-order polynomial in the original 3-D input space,

$$d(Z) = w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4 (x_1)^2 + w_5 x_1 x_2 + w_6 x_1 x_3 + b$$
$$= w_1 z_1 + w_2 z_2 + w_3 z_3 + w_4 z_4 + w_5 z_5 + w_6 z_6 + b$$

But there are some problems. First, how do we choose the nonlinear mapping to a higher dimensional space? Second, the computation involved will be costly. Refer back to Equation (8) for the classification of a test tuple, X^T. Given the test tuple, we have to compute its dot product with every one of the support vectors. In training, we have to compute a similar dot product several times in order to find the MMH. This is especially expensive. Hence, the dot product computation required is very heavy and costly.

Luckily, we can use another math trick. It so happens that in solving the quadratic optimization problem of the linear SVM (i.e. when searching for a linear SVM in the new higher dimensional space), the training tuples appear only in the form of dot products, $\phi(\boldsymbol{Xi}).\phi(\boldsymbol{Xj})$, where $\phi(\boldsymbol{X})$ is simply the nonlinear mapping function applied to transform the training tuples. Instead of computing the dot product on the transformed data tuples, it turns out that it is mathematically equivalent to instead apply a kernel function, $K(X_i, X_j)$, to the original input data. That is,

$$K(Xi, Xj) = \phi(Xi) * \phi(Xj). \qquad \text{(III.9)}$$

In other words, everywhere that $\phi(Xi) * \phi(Xj)$ appears in the training algorithm, we can replace it with $K(X_i; X_j)$. In this way, all calculations are made in the original input space, which is of potentially much lower dimensionality! We can safely avoid the mapping—it turns out that we don't even have to know what the mapping is! Later we will talk more about what kinds of functions can be used as kernel functions for this problem.

After applying this trick, we can then proceed to find a maximal separating hyperplane. The procedure is similar to that described in Section III.2, although it involves placing a user-specified upper bound, C, on the Lagrange multipliers, αi. This upper bound is best determined experimentally [68].

"What are some of the kernel functions that could be used?" Properties of the kinds of kernel functions that could be used to replace the dot product scenario described above have been studied. Three admissible kernel functions include:

Polynomial kernel of degree *h*: $K(Xi, Xj) = (Xi \,.\, Xj + 1)^h$ (III.10)

Gaussian radial basis function kernel: $K(Xi, Xj) = e\text{^}(-||Xi - Xj||^2/2\sigma^2)$ (III.11)

Sigmoid kernel: $K(Xi, Xj) = \tanh(\kappa Xi \,.\, Xj - \delta)$ (III.12)

Each of these results in a different nonlinear classifier in (the original) input space. Neural network aficionados will be interested to note that the resulting decision hyperplanes found for nonlinear SVMs are the same type as those found by other well-known neural network classifiers. For instance, an SVM with a Gaussian radial basis function (RBF) gives the same decision hyperplane as a type of neural network known as a radial basis function (RBF) network. An SVM with a sigmoid kernel is equivalent to a simple two-layer neural network known as a multilayer perceptron (with no hidden layers). There are no golden rules for determining which admissible kernel will result in the most accurate SVM. In practice, the kernel chosen does not generally make a large difference in resulting accuracy. SVM training always finds a global solution, unlike neural networks. So far, we have described linear and nonlinear SVMs for binary (i.e., two-class) classification. SVM classifiers can be combined for the multiclass case. A simple and effective approach, given *m* classes, trains *m* classifiers, one for each class (where classifier *j* learns to return a positive value for class *j* and a negative value for the rest). A test tuple is assigned the class corresponding to the largest positive distance.

Aside from classification, SVMs can also be designed for linear and nonlinear regression. Here, instead of learning to predict discrete class labels (like the $yi \in \{+1, -1\}$ above), SVMs for regression attempt to learn the input-output relationship between input training tuples, $\boldsymbol{X_i}$, and their corresponding continuous-valued outputs, $yi \in \mathcal{R}$. An approach similar to SVMs for classification is followed. Additional user-specified parameters are required.

A major research goal regarding SVMs is to improve the speed in training and testing so that SVMs may become a more feasible option for very large data sets (e.g., of millions of support vectors). Other issues include determining the best kernel for a given data set and finding more efficient methods for the multiclass case [68].

III.4 Conclusion

In this chapter, we got to know the cases when the data is linearly separable and linearly inseparable. We got to know the four kernels and their algorithms.

In the next chapter, we are going to discuss some tests to know the percentages of accuracy for each kernel and put them in comparison with each other.

IV.1 Introduction

Experience shows that no single machine learning scheme is appropriate to all data mining problems. The universal learner is an idealistic fantasy. As we have emphasized throughout this chapter, real datasets vary and to obtain accurate models the bias of the learning algorithm must match the structure of the domain [69].

IV.2 The Weka

The Weka workbench is a collection of state of the art machine learning algorithms and data preprocessing tools. It includes virtually all the algorithms described in previously chapter. It is designed so that you can quickly try out existing methods on new datasets in flexible ways. It provides extensive support for the whole process of experimental data mining, including preparing the input data, evaluating learning schemes statistically, and visualizing the input data and the result of learning. As well as a wide variety of learning algorithms, it includes a wide range of preprocessing tools.

Weka was developed at the University of Waikato in New Zealand, and the name stands for Waikato Environment for Knowledge Analysis. Outside the university the weka, pronounced to rhyme with Mecca, is a flightless bird with an inquisitive nature found only on the islands of New Zealand. The system is written in Java and distributed under the terms of the GNU General Public License [69].

Fig IV.1 The Weka

IV.2.1 What's in Weka?

Weka provides implementations of learning algorithms that you can easily apply to your dataset. It also includes a variety of tools for transforming datasets.

The workbench includes methods for all the standard data mining problems: regression, classification, clustering, association rule mining, and attribute selection. Getting to know the data is an integral part of the work, and many data visualization facilities and data preprocessing tools are provided. All algorithms take their input in the form of a single relational table in the ARFF format (we will describe it later) which can be read from a file or generated by a database query. One way of using Weka is to apply a learning method to a dataset and analyze its output to learn more about the data. Another is to use learned models to generate predictions on new instances. A third is to apply several different learners and compare their performance in order to choose one for prediction. The learning methods are called classifiers, and in the interactive Weka interface you select the one you want from a menu. Many classifiers have tunable parameters, which you access through a property sheet or object editor. A common evaluation module is used to measure the performance of all classifiers.

Implementations of actual learning schemes are the most valuable resource that Weka provides. But tools for preprocessing the data, called filters, come a close second. Like classifiers, you select filters from a menu and tailor them to your requirements. We will show how different filters can be used, list the filtering algorithms, and describe their parameters. Weka also includes implementations of algorithms for learning association rules, clustering data for which no class value is specified, and selecting relevant attributes in the data [69].

IV.2.2 How do you use it?

The easiest way to use Weka is through a graphical user interface called the Explorer. This gives access to all of its facilities using menu selection and form filling. For example, you can quickly read in a dataset from an ARFF file (or spreadsheet) and build a decision tree from it. But learning decision trees is just the beginning: there are many other algorithms to explore. The Explorer interface helps you do just that. It guides you by presenting choices as menus, by forcing you to work in an appropriate order by graying out options until they are applicable, and by presenting options as forms to be filled out. Helpful tool tips pop up as the mouse passes over items on the screen to explain what they do. Sensible default values ensure that you can

obtain results with a minimum of effort but you will have to think about what you are doing to understand what the results mean.

There are two other graphical user interfaces to Weka. The Knowledge Flow interface allows you to design configurations for streamed data processing. A fundamental disadvantage of the Explorer is that it holds everything in main memory when you open a dataset, it immediately loads it all in. This means that it can only be applied to small to medium-sized problems. However, Weka contains some incremental algorithms that can be used to process very large datasets. The Knowledge Flow interface lets you drag boxes representing learning algorithms and data sources around the screen and join them together into the configuration you want. It enables you to specify a data stream by connecting components representing data sources, preprocessing tools, learning algorithms, evaluation methods, and visualization modules. If the filters and learning algorithms are capable of incremental learning, data will be loaded and processed incrementally.

Weka's third interface, the Experimenter, is designed to help you answer a basic practical question when applying classification and regression techniques: which methods and parameter values work best for the given problem? There is usually no way to answer this question a priori, and one reason we developed the workbench was to provide an environment that enables Weka users to compare a variety of learning techniques. This can be done interactively using the Explorer. However, the Experimenter allows you to automate the process by making it easy to run classifiers and filters with different parameter settings on a corpus of datasets, collect performance statistics, and perform significance tests. Advanced users can employ the Experimenter to distribute the computing load across multiple machines using Java remote method invocation (RMI). In this way you can set up large-scale statistical experiments and leave them to run.

Behind these interactive interfaces lies the basic functionality of Weka. This can be accessed in raw form by entering textual commands, which gives access to all features of the system. When you fire up Weka you have to choose among four different user interfaces: the Explorer, the Knowledge Flow, the Experimenter, and the command-line interface. Most people choose the Explorer, at least initially [69].

IV.2.3 ARFF format

We now look at a standard way of representing datasets that consist of independent, unordered instances and do not involve relationships among instances, called an ARFF file.

Figure IV.2 shows an ARFF file for the weather data in, the version with some numeric features. Lines beginning with a % sign are comments. Following the comments at the beginning of the file are the name of the relation (weather) and a block defining the attributes (outlook, temperature, humidity, windy, play?). Nominal attributes are followed by the set of values they can take on, enclosed in curly braces. Values can include spaces; if so, they must be placed within quotation marks. Numeric values are followed by the keyword numeric.

```
% ARFF file for the weather data with some numeric features
%
@relation weather

@attribute outlook { sunny, overcast, rainy }
@attribute temperature numeric
@attribute humidity numeric
@attribute windy { true, false }
@attribute play? { yes, no }

@data
%
% 14 instances
%
sunny, 85, 85, false, no
sunny, 80, 90, true, no
overcast, 83, 86, false, yes
rainy, 70, 96, false, yes
rainy, 68, 80, false, yes
rainy, 65, 70, true, no
overcast, 64, 65, true, yes
sunny, 72, 95, false, no
sunny, 69, 70, false, yes
rainy, 75, 80, false, yes
sunny, 75, 70, true, yes
overcast, 72, 90, true, yes
overcast, 81, 75, false, yes
rainy, 71, 91, true, no
```

Fig IV.2 ARFF file for the weather data.

Although the weather problem is to predict the class value play? From the values of the other attributes, the class attribute is not distinguished in any way in the data file. The ARFF format merely gives a dataset; it does not specify which of the attributes the one that is supposed to be predicted is. This means that the same file can be used for investigating how well each attribute

can be predicted from the others, or to find association rules, or for clustering. Following the attribute definitions is an @data line that signals the start of the instances in the dataset. Instances are written one per line, with values for each attribute in turn, separated by commas. If a value is missing it is represented by a single question mark (there are no missing values in this dataset). The attribute specifications in ARFF files allow the dataset to be checked to ensure that it contains legal values for all attributes, and programs that read ARFF files do this checking automatically. In addition to nominal and numeric attributes, exemplified by the weather data, the ARFF format has two further attribute types: string attributes and date attributes. String attributes have values that are textual. Suppose you have a string attribute that you want to call description. In the block defining the attributes, it is specified as follows:

@attribute description string

Then, in the instance data, include any character string in quotation marks (to include quotation marks in your string, use the standard convention of preceding each one by a backslash, \). Strings are stored internally in a string table and represented by their address in that table. Thus two strings that contain the same characters will have the same value. String attributes can have values that are very long even a whole document. To be able to use string attributes for text mining, it is necessary to be able to manipulate them. For example, a string attribute might be converted into many numeric attributes, one for each word in the string, whose value is the number of times that word appears.
Date attributes are strings with a special format and are introduced like this:

@attribute today date

In the data section of the file, dates are specified as the corresponding string representation of the date and time. Although they are specified as strings, dates are converted to numeric form when the input file is read. Dates can also be converted internally to different formats, so you can have absolute timestamps in the data file and use transformations to forms such as time of day or day of the week to detect periodic behavior [69].

IV.3 Results and discusses

Our tests is made on 25pdb. We will make several tests using features extracted only from Amino Acids, and we will use different cost's values.

IV.3.1 Amino Acids Composition

One of the earliest methods and the most common ways to represent a protein's amino acid sequence is to compute amino acid composition of the protein. Amino acid composition calculates the fraction of each amino acid type in a protein sequence. Twenty descriptor values are computed for the 20 types of amino acids.

Amino acid composition calculates the fraction of each amino acid type in a sequence as defined in equation below:

$$f(r) = \frac{N_r}{N} \qquad \text{(IV.1)}$$

Where r = 1, 2, 3, . . . , 20, N_r is the number of amino acid of type r, and N is the length of the sequence [13].

In this work we will discuss the results by putting different cost's values.

Case 1: When cost = 1

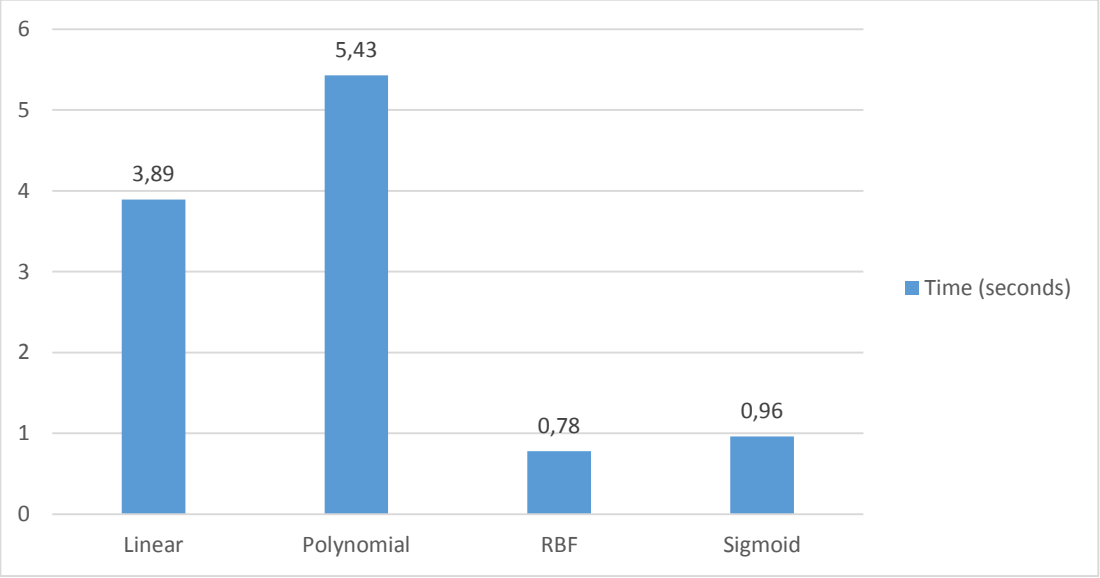

Fig IV.3 A graph shows the period of time that the four different kernels take

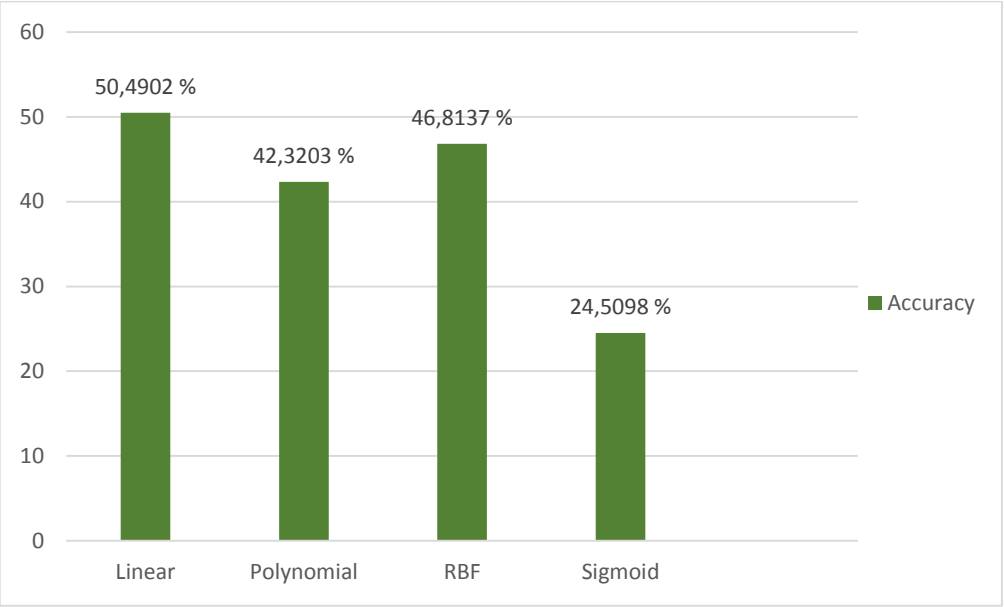

Fig IV.4 A graph shows the accuracy percentages of the four different kernels

In this case, we notice that the Polynomial takes the longest time while the RBF takes the shortest. In contrast with the best prediction's percentage which is the Linear the Sigmoid is the worst.

Case 2: When cost = 3

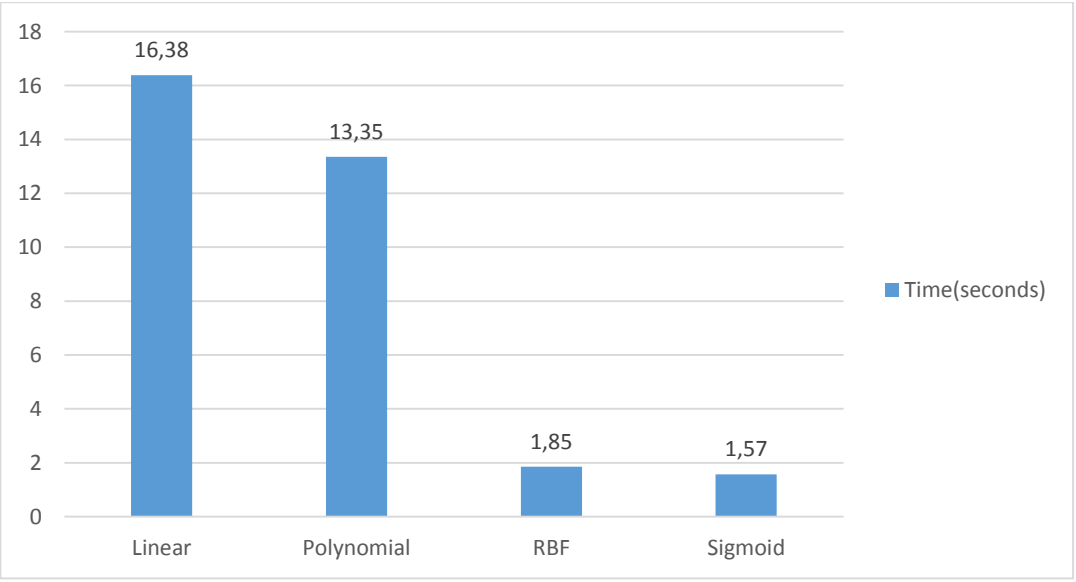

Fig IV.5 A graph shows the period of time that the four different kernels take

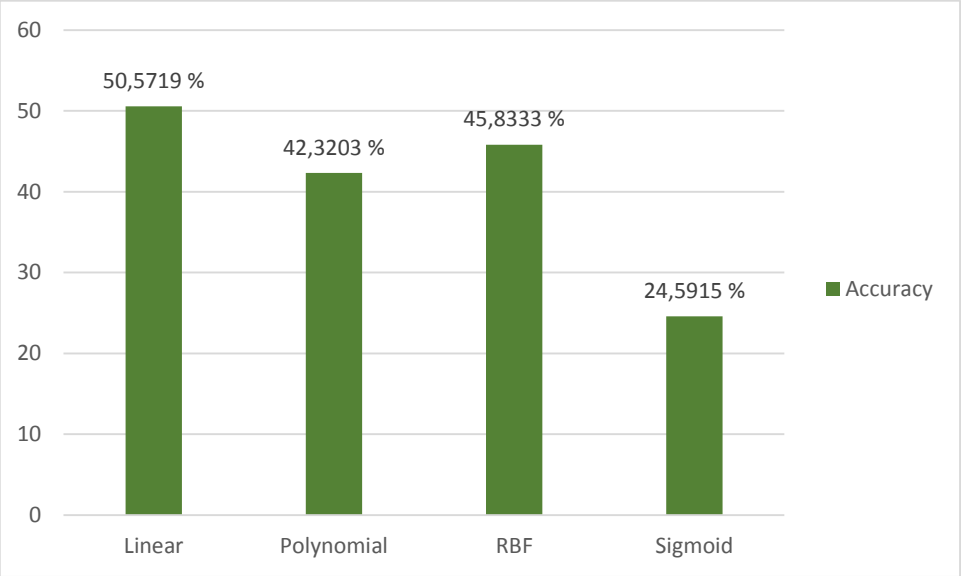

Fig IV.6 A graph shows the accuracy percentages of the four different kernels

In this case when we put cost =3, we notice that the Linear takes the longest time, yet it has the best prediction's percentage.

By putting the two cases in comparison we find out that the two accuracy's percentages of Linear kernel are slightly different.

IV.3.2 Dipeptide Composition

Dipeptide composition calculates the fractions of pairs of amino acids i.e. it will search for all AA, AC, AD, AE etc. and then next set of amino acids CA, CC, CD, CE and then next DA, DC, DD, DE etc. defined in equation below:

$$f(r,s) = \frac{N_{rs}}{N-1} \qquad \text{(IV.2)}$$

where *r*,*s* = 1, 2, 3, . . . , 20, and N_{rs} is the number of dipeptides of amino acid type *r* and *s*. Four hundred descriptor values are calculated for the 20 x 20 amino acid combinations [13].

Now, we are going to use different cost's values.

Case 1: When cost = 1

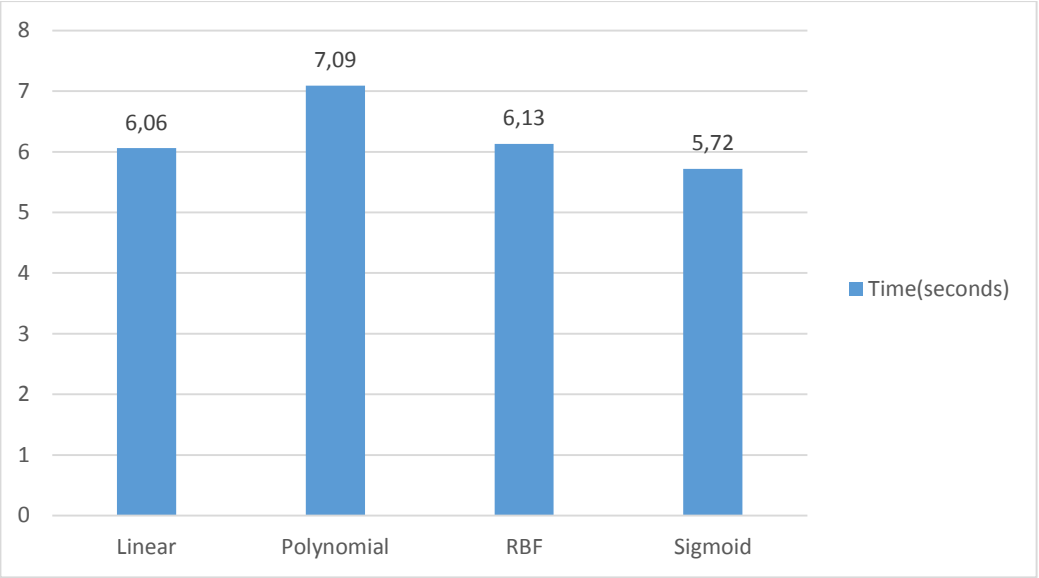

Fig IV.7 A graph shows the period of time that the four different kernels take

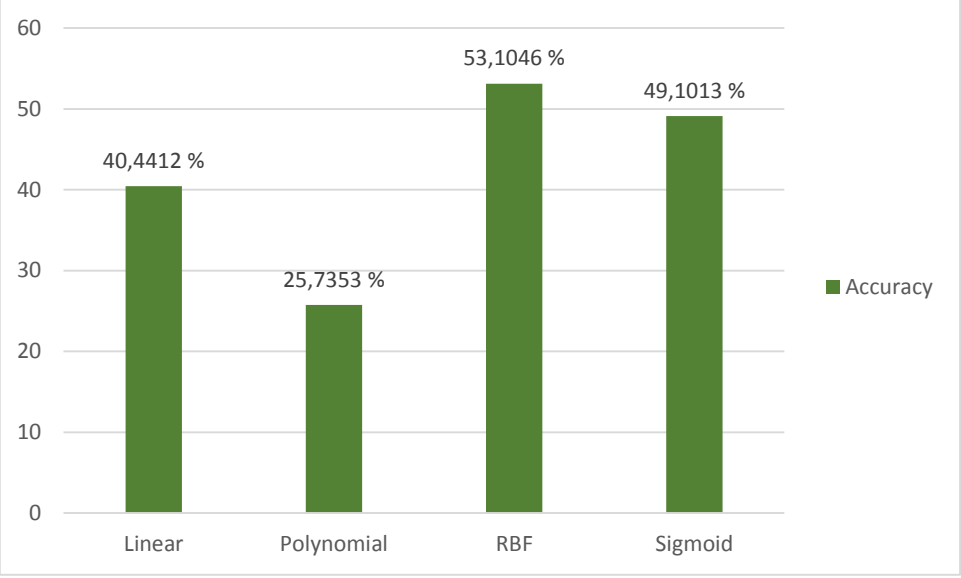

Fig IV.8 A graph shows the accuracy percentages of the four different kernels

In case 1 when the cost=1 the RBF takes longer time than the Linear and the Sigmoid but shorter than the polynomial. In the percentage the RBF scores the best prediction's percentage.

Case 2: When cost = 3

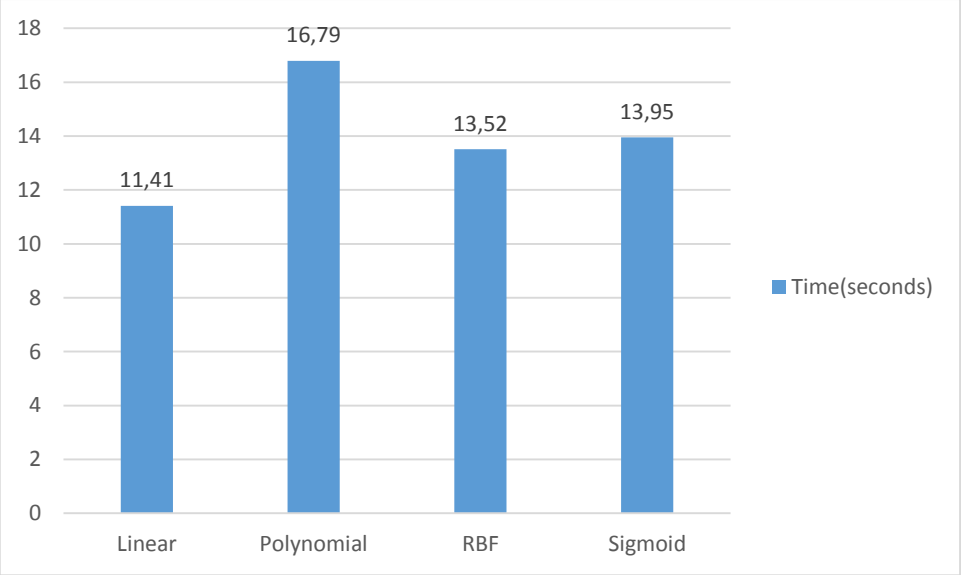

Fig IV.9 A graph shows the period of time that the four different kernels take

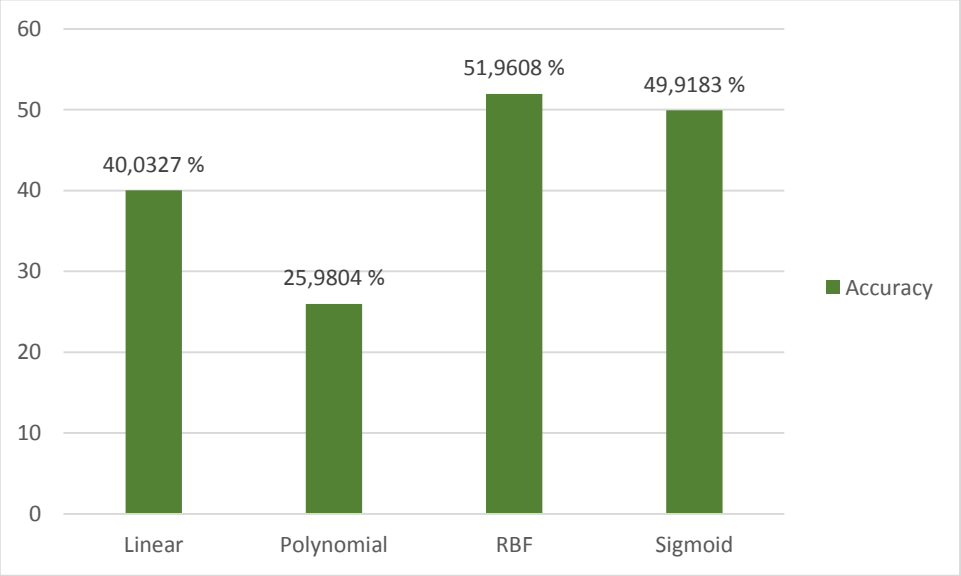

Fig IV.10 A graph shows the accuracy percentages of the four different kernels

In the second case when the cost=3 we notice that the Polynomial takes the longest time with the worst accuracy percentage.

In the two different cases (when cost = 1) and (when cost = 3) we figure out that the RBF kernel gives the best accuracy's percentage. When (cost = 1) it results with better percentages of accuracy than when (cost =3).

IV.3.3 Amino Acids Composition and Dipeptide Composition

In this section we gather Amino Acids Composition and Dipeptide Composition, and we will have 420 features as an outcome of the addition of the features.

Now we will review the results using (cost = 1) and (cost = 3) .

Case 1: When cost = 1

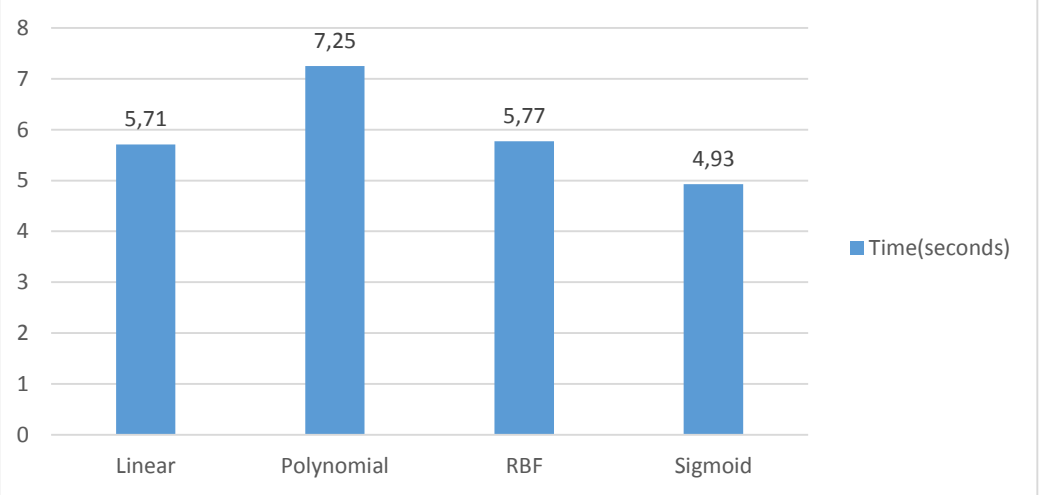

Fig IV.11 A graph shows the period of time that the four different kernels take

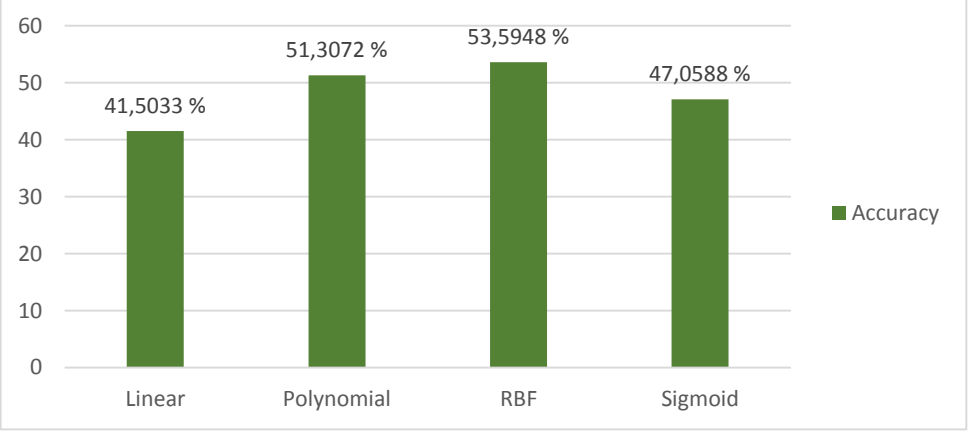

Fig IV.12 A graph shows the accuracy percentages of the four different kernels

Although the Polynomial takes longer time, its accuracy is better than the others regardless of the accuracy of RBF.

Case 2: When cost = 3

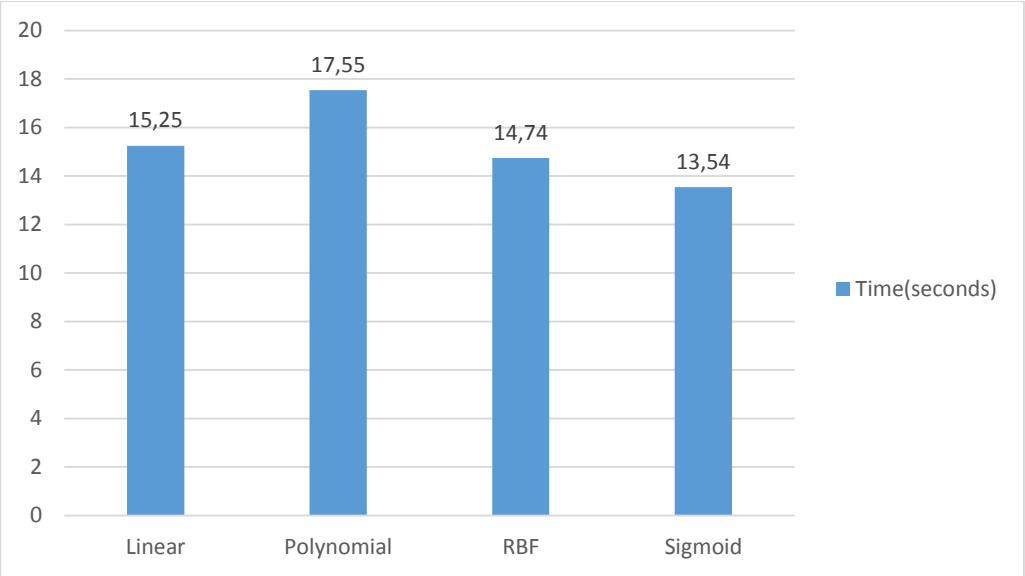

Fig IV.13 A graph shows the period of time that the four different kernels take

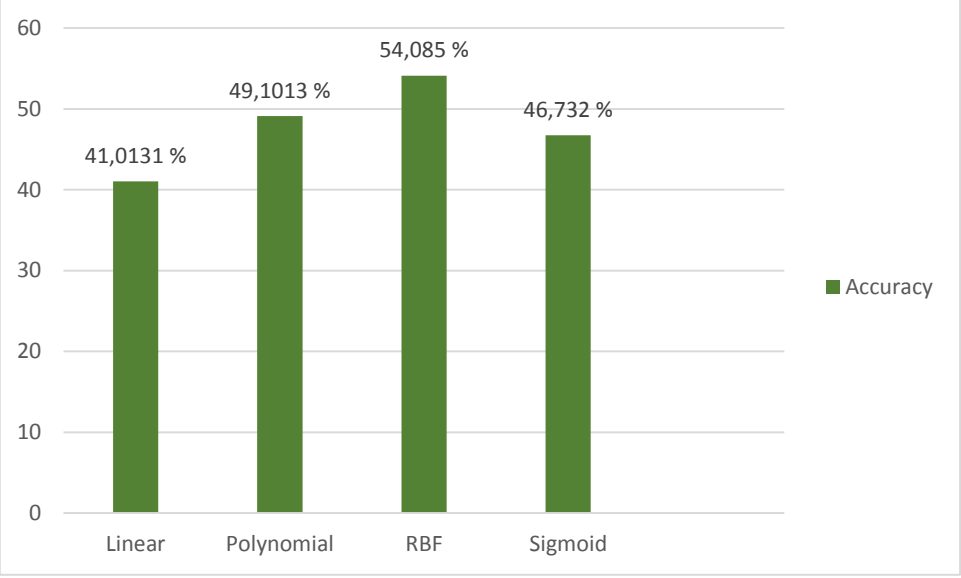

Fig IV.14 A graph shows the accuracy percentages of the four different kernels

In this case, RBF has the most accuracy percentage with a big difference with the others. In the two cases, the RBF kernel attains the highest accuracy's percentage. We also notice that the RBF when (cost =3) has a higher percentage than when (cost = 1).

Throughout the tests we made using various features which were:

- Amino Acids Composition.
- Dipeptide Composition.
- Amino Acids Composition and Dipeptide Composition.

and by using the two cases of cost's values, we conclude that the usage of RBF kernel is better than the others for its good outcome.

IV.4 Conclusion

In this chapter we got to know the Weka, what is in it, and how we use it. We have made several tests with different features set of Amino Acids, Dipeptide Composition and a gathering of them both.

GENERAL CONCLUSION

In this project of master, a soft computing technique like SVM was used to construct classification models for protein structural class prediction. We got to know the SVM, and we used it for the prediction of protein structure classes. Using SVM for prediction seems to be the most popular among all soft computing techniques to solve the protein structural classification problem.

There were old methods of classification of proteins which depended on only features extracted from Amino Acids sequences that has 20 features, and there was another method of classification of protein which relied on features extracted from Dipeptide composition that has 400 features.

Accuracy tests have used 25PDB dataset which is the most popular. We have made some tests on four different kernel kinds of SVM which were linear, polynomial, RBT, and sigmoid. To enhance the results, we have used different cost values in three different tests which were Amino Acids composition, Dipeptide composition, and a gathering of Amino Acids and Dipeptide composition. The results of these tests were similar to each other. We notice that the RBF kernel when (cost=3) scores the highest accuracy's percentages.

In my opinion, the good accuracy's percentages are very important regardless the time it may takes.

Bibliography

[1] Jin Xiong ”ESSENTIAL BIOINFORMATICS “,Cambridge university press,2006
[2] What is Molecular Biology?,”
http://www.news-medical.net/life-sciences/What-is-Molecular-Biology.aspx”,
[Accessed 30/4/2016] .
[3] D. Robert et B. Vian, ”´El´ements de biologie cellulaire 3e ´edition”, Doin, 2004
[4] Oliver Brandenberg, Zephaniah Dhlamini, Alessandra Sensi, Kakoli Ghosh, Andrea Sonnino“Introduction to Molecular Biology and Genetic Engineering “ Rome, 2011.
[5] Georgia C. Lauritzen “What is protein?“ Utah State University, 1992.
[7] Patricio Jeraldo “The Genetic Code“ May 5, 2006
[8] C. Burge, “Bioinformaticists Will Be Busy Bees,” Genome Technology, No. 17, January, 2002.
[9] Andreas D. Baxevanis, B.F. Francis Ouellette “A Practical Guide to the Analysis of Genes and Proteins, Second Edition”, 2001.
[11] Xiaohua Hu, Yi Pan “KNOWLEDGE DISCOVERY IN BIOINFORMATICS, Techniques, Methods, and Applications” John Wiley & Sons, 2007.
[13] Mr. Sundeep Singh Nanuwa, " Investigation into the role of sequence-driven-features and amino acid indices for the prediction of structural classes of proteins ", A thesis, De Montfort University, April 2013.
[14] H. Nakashima, K. Nishikawa, T. Ooi, The folding type of a protein is relevant to the amino acid composition, J. Biochem. 99 (1986) 153–162.
[15] K.C. Chou, A novel approach to predicting protein structural classes in a (20-1)-D amino acid composition space, Proteins 21 (1995) 319–344.
[16] W. Kabsch, C. Sander, Dictionary of protein secondary structures: pattern recognition of hydrogen-bonded and geometrical features, Biopolymers 22 (1983) 2577–2637.
[17] F. Eisenhaber, C. Frommel, P. Argos, Prediction of secondary structural content of proteins from their amino acid composition alone, II the paradox with secondary structural class, Proteins 25 (1996) 169–179.
[18] H.M. Berman, et al., The protein data bank, Nucleic Acids Res. 28 (2000) 235–242.
[19] A. Murzin, S. Brenner, T. Hubbard, C. Chothia, SCOP: a structural classification of protein database for the investigation of sequence and structures, J. Mol. Biol. 247 (1995) 536–540.
[20] K.C. Chou, C.T. Zhang, Prediction of protein structural classes, Crit. Rev. Biochem. Mol. Biol. 30 (1995) 275–349.
[21] K.C. Chou, G.M. Maggiora, Domain structural class prediction, Protein Eng. 11 (1998) 523–538.
[22] A. Andreeva, D. Howorth, S. Brenner, T. Hubbard, C. Chothia, A. Murzin, SCOP database in 2004: refinements integrate structure and sequence family data, Nucleic Acid Res. 32 (2004)

D226–D229.
[27] Zerrin Isik, Berrin Yanikoglu, and Ugur Sezerman. "Protein structural class determination using support vector machines." In Computer and Information Sciences-ISCIS 2004, pp. 82-89. Springer Berlin Heidelberg, 2004.
[28] Zhou, Guo-Ping. "An intriguing controversy over protein structural class prediction." Journal of Protein Chemistry 17, no. 8 (1998): 729-738.
[29] Wu, Li, Qi Dai, Bin Han, Lei Zhu, and Lihua Li. "Prediction of protein structural class using a combined representation of protein-sequence information and support vector machine." In Bioinformatics and Biomedicine Workshops (BIBMW), 2010 IEEE International Conference on, pp. 101-106. IEEE, 2010.
[30] Nair, Achuth Sankar S., and T. Mahalakshmi. "Visualization of genomic data using internucleotidedistance signals." Proceedings of IEEE Genomic Signal Processing (2005): 11-13.
[31] Afreixo, Vera, Carlos AC Bastos, Armando J. Pinho, Sara P. Garcia, and Paulo JSG Ferreira. "Genome analysis with inter-nucleotide distances." Bioinformatics 25, no. 23 (2009): 3064-3070.
[32] Zhang, T-L., and Y-S. Ding. "Using pseudo amino acid composition and binary-tree support vector machines to predict protein structural classes." Amino Acids 33, no. 4 (2007): 623- 629.
[33] Tang, Fa-ming, Zhong-dong Wang, and Mian-yun Chen. "On multiclass classification methods for support vector machines." Control and Decision 20, no. 7 (2005): 746.
[34] Ding, Yong-Sheng, Tong-Liang Zhang, and Kuo-Chen Chou. "Prediction of protein structure classes with pseudo amino acid composition and fuzzy support vector machine network." Protein and peptide letters 14, no. 8 (2007): 811-815.
[35] Chou, Kuo-Chen. "Prediction of protein cellular attributes using pseudo-amino acid composition." Proteins: Structure, Function, and Bioinformatics 43, no. 3 (2001): 246-255.
[36] Abe, Shigeo. "Fuzzy LP-SVMs for multiclass problems." In Proceedings of European Symposium on Artificial Neural Networks (ESANN'2004) Bruges, Belgium, pp. 429 - 434. 2004.
[37] Chou, Kuo-Chen. "A key driving force in determination of protein structural classes." Biochemical and biophysical research communications 264, no. 1 (1999): 216-224.
[38] Klein, Petr, and Charles Delisi. "Prediction of protein structural class from the amino acid sequence." Biopolymers 25, no. 9 (1986): 1659-1672.
[39] Bu, Wei-Shu, Zhi-Ping Feng, Ziding Zhang, and Chun-Ting Zhang. "Prediction of protein (domain) structural classes based on amino-acid index." European Journal of Biochemistry 266, no. 3 (1999): 1043-1049.
[40] Chou, Kuo-Chen. "A key driving force in determination of protein structural classes." Biochemical and biophysical research communications 264, no. 1 (1999): 216-224.

[41] Altschul, Stephen F., Thomas L. Madden, Alejandro A. Schaffer, Jinghui Zhang, Zheng Zhang, Webb Miller, and David J. Lipman. "Gapped BLAST and PSI-BLAST: a new generation of protein database search programs." Nucleic acids research 25, no. 17 (1997): 3389-3402.

[42] Liu, Taigang, Xingbo Geng, Xiaoqi Zheng, Rensuo Li, and Jun Wang. "Accurate prediction of protein structural class using auto covariance transformation of PSI-BLAST profiles." Amino acids 42, no. 6 (2012): 2243-2249.

[43] Mizianty, Marcin, and Lukasz Kurgan. "Modular prediction of protein structural classes from sequences of twilight-zone identity with predicting sequences" BMC bioinformatics 10.1 (2009): 414.

[44] Dong, Qiwen, Shuigeng Zhou, and Jihong Guan. "A new taxonomy-based protein fold recognition approach based on autocross-covariance transformation." Bioinformatics 25, no. 20 (2009): 2655-2662.

[45] Guo, Yanzhi, Lezheng Yu, Zhining Wen, and Menglong Li. "Using support vector machine combined with auto covariance to predict protein–protein interactions from protein sequences." Nucleic acids research 36, no. 9 (2008): 3025-3030.

[46] Guo, Yanzhi, Menglong Li, Minchun Lu, Zhining Wen, and Zhongtian Huang. "Predicting G-protein coupled receptors–G-protein coupling specificity based on autocross-covariance transform." Proteins: structure, function, and bioinformatics 65, no. 1 (2006): 55-60.

[47] Wu, Jiang, Meng-Long Li, Le-Zheng Yu, and Chao Wang. "An ensemble classifier of support vector machines used to predict protein structural classes by fusing auto covariance and pseudo-amino acid composition." The Protein Journal 29, no. 1 (2010): 62-67.

[48] Chou, Kuo-Chen, and Yu-Dong Cai. "Predicting protein structural class by functional domain composition." Biochemical and biophysical research communications 321, no. 4 (2004): 1007-1009.

[49] Ahmadi Adl, Amin, Abbas Nowzari-Dalini, Bin Xue, Vladimir N. Uversky, and Xiaoning Qian. "Accurate prediction of protein structural classes using functional domains and predicted secondary structure sequences." Journal of Biomolecular Structure and Dynamics 29, no. 6 (2012): 1127-1137.

[50] Apweiler, Rolf, Terri K. Attwood, Amos Bairoch, E. Birney, M. Biswas, P. Bucher, L. Cerutti et al. "The InterPro database, an integrated documentation resource for protein families, domains and functional sites." Nucleic acids research 29, no. 1 (2001): 37-40.

[51] Hall, Mark A. "Correlation-based feature selection for machine learning." PhD diss., The University of Waikato, 1999.

[52] Liu, Tian, and Cangzhi Jia. "A high-accuracy protein structural class prediction algorithm using predicted secondary structural information." Journal of theoretical biology 267, no.3 (2010): 272-275.

[53] Zhang, Shengli, Shuyan Ding, and Tianming Wang. "High-accuracy prediction of protein

structural class for low-similarity sequences based on predicted secondary structure." Biochimie 93, no. 4 (2011): 710-714.
[54] Ding, Shuyan, Shengli Zhang, Yang Li, and Tianming Wang. "A novel protein structural classes prediction method based on predicted secondary structure." Biochimie 94, no. 5 (2012): 1166-1171.
[55] Jones, David T. "Protein secondary structure prediction based on position-specific scoring matrices." Journal of molecular biology 292, no. 2 (1999): 195-202.
[56] Lin, Kuang, Victor A. Simossis, Willam R. Taylor, and Jaap Heringa. "A simple and fast secondary structure prediction method using hidden neural networks." Bioinformatics 21, no. 2 (2005): 152-159.
[57] Kurgan, Lukasz A., Tuo Zhang, Hua Zhang, Shiyi Shen, and Jishou Ruan. "Secondary structure-based assignment of the protein structural classes." Amino Acids 35, no. 3 (2008): 551-564.
[58] Kurgan, Lukasz, Krzysztof Cios, and Ke Chen. "SCPRED: accurate prediction of protein structural class for sequences of twilight-zone similarity with predicting sequences." BMC bioinformatics 9, no. 1 (2008): 226.
[59] Mohammad, Tabrez Anwar Shamim, and Hampapathalu Adimurthy Nagarajaram. "Svmbased method for protein structural class prediction using secondary structural content and structural information of amino acids." Journal of Bioinformatics and Computational biology 9, no. 4 (2011): 489-502.
[60] Cheng, Jianlin, Arlo Z. Randall, Michael J. Sweredoski, and Pierre Baldi. "SCRATCH: a protein structure and structural feature prediction server." Nucleic acids research 33, no. suppl 2 (2005): W72-W76.
[61] Kurgan, Lukasz, and Ke Chen. "Prediction of protein structural class for the twilight zone sequences." Biochemical and biophysical research communications 357, no. 2 (2007): 453-460.
[62] Yang, Jian-Yi, Zhen-Ling Peng, and Xin Chen. "Prediction of protein structural classes for low-homology sequences based on predicted secondary structure." BMC bioinformatics 11, no. Suppl 1 (2010): S9.
[63] Hastie, Trevor, and Robert Tibshirani. "Classification by pairwise coupling." The annals of statistics 26, no. 2 (1998): 451-471.
[64] Cai, Yu-Dong, and Guo-Ping Zhou. "Prediction of protein structural classes by neural network." Biochimie 82, no. 8 (2000): 783-785.
[65] Chandonia, John-Marc, and Martin Karplus. "Neural networks for secondary structure and structural class predictions." Protein Science 4, no. 2 (1995): 275-285.
[66] Syeda Nadia Firdaus, " PROTEIN STRUCTURAL CLASS PREDICTION USING PREDICTED SECONDARY STRUCTURE AND HYDROPATHY PROFILE", A thesis, (Master), Toronto- Canada, 2013.

[67] Lukasz A. Kurgan∗, Leila Homaeian, "Prediction of structural classes for protein sequences and domains—Impact of prediction algorithms, sequence representation and homology, and test procedures on accuracy", A thesis, Pattern Recognition 39 (2006) 2323 – 2343

[68] Jiawei Han , Micheline Kamber , “Data Mining Concepts and Techniques “ , second edition , Elsevier Inc., 2006.

[69] Eibe Frank, Ian H. Witten ‘’ Data Mining Practical Machine Learning Tools and Techniques‘’, Elsevier Inc., 2005.

[6] Protein structure "http://www.particlesciences.com/news/technical-briefs/2009/protein-structure.html " [Accessed 18/5/2016].

[10] http://www.biostat.jhsph.edu/~iruczins/teaching/260.655/links/pdbformat [Accessed 5/2/2016]

[12] Bioinformatics: Sequence File Formats, http://www.algosome.com/articles/bioinformatics-sequence-file-formats.html [Accessed 18/5/2016]

[23]"Information on 2hbc", http://www.rcsb.org/pdb/explore/explore.do?structureId=2hbc [Accessed 21/5/2016].

[24]"Information on 1ku8", http://www.rcsb.org/pdb/explore/explore.do?structureId=1ku8 [Accessed 21/5/2016].

[25]"Information on 2bnh", http://www.rcsb.org/pdb/explore/explore.do?structureId=2BNH [Accessed 21/5/2016].

[26]"Information on 1pya" http://www.rcsb.org/pdb/explore/explore.do?structureId=1pya [Accessed 21/5/2016].

ملخص

تتحكم البروتينات في جميع الوظائف البيولوجية للكائنات الحية. تتألف بنية البروتين من أربع فئات رئيسية، وكل فئة تؤدي وظيفة مختلفة وفقاً لطبيعتها. ونظراً للاستكشاف الكبير من سلاسل البروتين في بنوك المعلومات، يعتبر تحديد فئات بنية البروتين شيئاً صعباً من خلال الطرق التقليدية فيما يتعلق بالتكلفة والوقت. وبالنظر إلى أهمية فئات بنية البروتين، من المستحسن تطوير نموذج حاسوبي لمعرفة وتمييز تلك الفئات بدقة عالية. سنستخدم ثلاثة أنظمة استخراج للخصائص من الأحماض الأمينية لمعرفة المعلومات من تسلسل البروتين. يتم تقييم آداء النموذج المقترح باستخدام قاعدة بيانات البروتين. نسبة نجاح النموذج المقترح هو 54.085%.

كلمات مفتاحية: المعلومات الحيوية، تصنيف فئات بنية البروتين،SVM

Abstract

Proteins control all biological functions in living species. Protein structure is comprised of four major classes including all-α class, all-β class, α/β and α+β. Each class performs a different function according to their nature. Owing to the large exploration of protein sequences in the databanks, the identification of protein structure classes is difficult through conventional methods with respect to cost and time. Looking at the importance of protein structure classes, it is thus highly desirable to develop a computational model for discriminating protein structure classes with high accuracy. For this purpose, we propose a Support Vector Machine. Three features extraction schemes named Amino Acid Composition, Dipeptide Composition and a combination of them both are used to explore valuable information from protein sequences. The performance of the proposed model is assessed using the common dataset 25PDB. The success percentage of the proposed model is 54.085 %.

Key words: Bioinformatics, Support Vector Machine (SVM), Structural classes of protein.

Résumé

Les protéines contrôlent toutes les fonctions biologiques des espèces vivantes. La structure des protéines est composée de quatre grandes classes incluant all-α classe, all-β classe, α / β classe et α + β classe. Chaque classe a une fonction différente en fonction de leur nature. En raison de la grande exploration des séquences de protéines dans les banques de données, l'identification des classes de structure de protéines est difficile par les méthodes conventionnelles en ce qui concerne le coût et le temps. En regardant l'importance des classes de structures de protéines, il est donc hautement souhaitable de développer un modèle de calcul pour distinguer les classes de structure de protéines avec une grande précision. A cet effet, nous utilisons (SVM) Support Vector Machine. Trois systèmes d'extraction des caractéristiques du nom de composition d'acides aminés, de dipeptide Composition et une combinaison de tous les deux sont utilisées pour explorer des informations précieuses à partir de séquences de protéines. La performance du modèle proposé est évaluée en utilisant le 25PDB ensemble de données commun. Le pourcentage de réussite du modèle proposé est 54,085%.

Mots clé: Bioinformatiques, Support Vector Machine (SVM), les classes des structures de protéine.