Protein Structure Prediction using Machine Learning Algorithms and Feature Vectors

A report submitted to

RAMAIAH INSTITUTE OF TECHNOLOGY

Bengaluru

IS821 SENIOR PROJECT

as partial fulfilment of the requirement for

Bachelor of Engineering (B.E) in Information Science and Engineering

by

Affan Ahmed (USN- 1MS14IS003)
Anthony Prajwal P (USN- 1MS14IS015)
Keshav D Karanth (USN- 1MS14IS028)
Kartik Kapoor (USN- 1MS14IS138)

under the guidance of

Prof. S. R. Mani Sekhar (Dept. of ISE, RIT)

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

RAMAIAH INSTITUTE OF TECHNOLOGY

May 2018

Department of Information Science and Engineering

Ramaiah Institute of Technology

Bengaluru - 54

CERTIFICATE

This is to certify that Affan Ahmed (USN- 1MS14IS003), Anthony Prajwal P (USN- 1MS14IS015), Keshav D Karanth (USN- 1MS14IS028) and Kartik Kapoor (USN- 1MS14IS138) who were working for their **IS821 SENIOR PROJECT** under my guidance, have completed the work as per my satisfaction with the topic **Protein Structure Prediction using Machine Learning Algorithms and Feature Vectors**. To the best of my understanding the work to be submitted in dissertation does not contain any work, which has been previously carried out by others and submitted by the candidates for themselves for the award of any degree anywhere.

(Internal Guide)
Mr. S. R. Mani Sekhar
Professor, Dept. of ISE

(Head of the Department)
Dr. Vijaya Kumar B P
Professor & Head, Dept. of ISE

(Examiner 1)

(Examiner 2)

Name
Signature

Department of Information Science and Engineering
Ramaiah Institute of Technology
Bengaluru - 54

DECLARATION

We hereby declare that the entire work embodied in this **IS821 SENIOR PROJECT** report has been carried out by us at Ramaiah Institute of Technology under the supervision of Prof. S. R. Mani Sekhar. This Project report has not been submitted in part or full for the award of any diploma or degree of this or any other University.

Affan Ahmed (USN- 1MS14IS003)
Anthony Prajwal P (USN- 1MS14IS015)
Keshav D Karanth (USN- 1MS14IS028)
Kartik Kapoor (USN- 1MS14IS138)

Acknowledgements

The progress of this project would not have been possible without the unconditional support of many people, each of whom in their own capacities have helped us in carrying out this project. We would like to take this special opportunity to thank Dr. Vijaya Kumar B P, Professor and Head, Department of Information Science and Engineering, for permitting us to take up this project. We also wish to express our gratitude to our project guide, Prof. S. R. Mani Sekhar, who has been abundantly helpful and has offered invaluable assistance and exemplary guidance to us. We would also like to thank all the teaching and nonteaching staff of our department for their kind cooperation and assistance.

Abstract

Protein structure prediction is the deduction of the three-dimensional structures of proteins from their amino acid sequences, which involves the prediction of its folding and the secondary structure from its primary sequence.

New proteins on discovery are subject to structure prediction. A linear protein structure is derived from the protein in the form of chains which is the primary structure. The chain provides information about arrangement of the amino acids that is helix, strands, loops and turns. There are several algorithms that have been used to predict the structure of the protein.

Although plenty of protein sequence data is available online, it was not easy to acquire data that was pretty classified with labels. We scraped information from various websites allowing us to get class labels for our unsupervised data set.

Our attempt uses Feature Vectors methodology to generate a transformed view of the protein sequences, which are then used as input to classifiers like Random Forest and Naive Bayesian. With this we are able to compare and contrast the precision and accuracy of the predicted sequences. Results of the comparison will provide deep insights in methodologies used for structure prediction in the future. Our research yields the best result using the Random Forest Classifier with an astounding accuracy of 96%.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Computationally predicted three-dimensional structure of proteins has demonstrated the usefulness in many areas of biomedicine, ranging from approximate family healthcare measures to precise drug screening. However, or nearly 40 years the accuracy of the predicted models has been dictated by the availability of close structural templates. Progress has recently been achieved in refining low-resolution structures closer to the native ones; this has been made possible by combining knowledge-based information from multiple sources of structural templates. Unfortunately, no essential progress in the development of techniques for detecting remotely homologous templates and for predicting novel protein structures.

Determining the three-dimensional structure of protein molecules is a cornerstone for many aspects of modern biological research. Currently, over 7 million protein sequences are deposited in the UniProtKB/TrEMBL database but only ˜50,000 of them have experimentally solved structures. These numbers can be frustrating to molecular and cell biologists who need 3D models of proteins for their research: the chance of a protein domain to have a solved structure has dropped to 0.7% by the end of 2008; this number was 2% in 2004 and 1.2% in 2007 [1]. The high demand of the community for protein structures has placed

[1] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2673339

computer-based protein structure prediction, the only means to alleviate the problem, at an unprecedentedly critical position.

We believe, that our case study will help scientists make informed decisions while picking an approach for protein modelling and drug research.

1.2 Scope

The scope of this project is as follows :

- **Collection and Analysis of Data.**
- **Design and Development of Feature Vector.**
- **Prarallel Programming.**
- **Design of UI.**

Collection and Analysis of Data using Selenium for extracting protein data from websites by processing the HTML Web Page and extracting data for manipulation to a local storage. Once the protein sequences are stored locally, the application can run without an internet connection. Protein Data Banks[1] contain millions of sequences and the whole process was parallelised using Java and multithreading to increase computational performance. The sequence data that is collected from the internet had a mix of protein and non-protein data (such as DNA and RNA) which was filtered and cleansed for our requirement.

Design and Development of Feature Vector which is a measurable entity or characteristic that is used to describe an attribute or feature of the objects. Selection of feature vectors is key to improving the performance of the predictor algorithm, for structure prediction the features selected need to have direct correlation to the three dimentional structure of the protein. The four feature vectors most suitable for us are Hydrophobicity, Polarizability, Polarity and Van der Waals Volume. These features are combined in a matrix to improve the efficiency of the vector.

[1] https://www.rcsb.org/

Parallel programming Structure prediction is a computationally expensive and time consuming process. In order to overcome this, we designed a parallel programming predictor by sending sequences to multiple classifiers at once, this increases the overall performance and usability of the application as the users can compare the results returned by the classifiers simultaneously. Further, the data collection process was done parallelly by scraping multiple websites at once.

Design of UI that provides an interactive interface to run the sequence search either parallel or in sequence, based on the users requirements. The UI displays the feature vector along with the predicted structures. We kept the UI minimalistic and straight forward to aid in convenience to the users who may comprise of research students or scientists.

1.3 Objectives

We want to aid in the development of medical science. Drug development in particular requires knowledge of the binding sites of the candidate compounds, a well predicted structure helps in the computational screening and optimizing candidate compounds. Identifying the mechanism by which a protein functions and how it folds is of great curiosity for researchers and students alike. Taking this into account, we look to present a case study which will provide insights on the most accurate algorithm for structure prediction using feature vectors.

1.4 Proposed Model

Our application accepts primary protein sequences from the user, one at a time. This process can be automated by using a script to process multiple sequences. It application requires the user to choose which classifier they prefer to use for their analyse i.e Naive Bayes, Support Vector Machines or Random Forest.
Feature Vectors are generated from the entered sequence and then analysed to predict class labels for each entered sequence. The protein sequences vary in length and can be categorised based on various physical and stereo chemical properties, these properties determine the feature vector generation. The attributes used to

describe the sequence in our application include Polarity, Hydrophobicity, Polarizability and Van der Waals Volume.
Our application can return the predicted labels of multiple classifiers at once. This helps the user compare and understand the best algorithm to be used for prediction. Our research yields the best result using the Random Forest Classifier with an astounding accuracy of 96%.

1.5 Organisation of Report

In this report, we start with a Literature review that consolidates all the current knowledge, substantive findings including theoretical and methodological contributions to our problem. We then move onto the System analysis and design which gives us a description of all the components of the system and explains how they interact. Followed by Modelling and implementation section that will give a detailed overview of the implementation aspects of the project. We then take a look at Testing and results to discuss the findings we have made from the data we have collected. And we finish off with the Conclusion and future work section.

Chapter 2

Literature Review

Proteins are chains of Amino Acids aka building blocks. DNA holds the information and Protein do the work. The primary structure of a protein refers to the sequence of amino acids in the polypeptide chain. The primary structure is held together by peptide bonds that are made during the process of protein biosynthesis. The two ends of the polypeptide chain are referred to as the carboxyl terminus (C-terminus) and the amino terminus (N-terminus) based on the nature of the free group on each extremity.

The primary structure of a protein is determined by the gene corresponding to the protein. A specific sequence of nucleotides in DNA is transcribed into mRNA, which is read by the ribosome in a process called translation. The sequence of amino acids in insulin was discovered by Frederick Sanger[1], establishing that proteins have defining amino acid sequences. The sequence of a protein is unique to that protein, and defines the structure and function of the protein. The sequence of a protein can be determined by methods such as Edman degradation[2] or tandem mass spectrometry. Often, however, it is read directly from the sequence of the gene using the genetic code. Post-translational modification such as disulfide bond formation, phosphorylations and glycosylations are usually also considered a part of the primary structure, and cannot be read from the gene. For example, insulin is composed of 51 amino acids in 2 chains. One chain has 31 amino acids, and the other has 20 amino acids. (Refer to Figure 2.1)

[1] https://www.nobelprize.org/nobel_prizes/chemistry/laureates/1958/sanger-bio.html
[2] http://www.toplab.de/Edman%20degradation.pdf

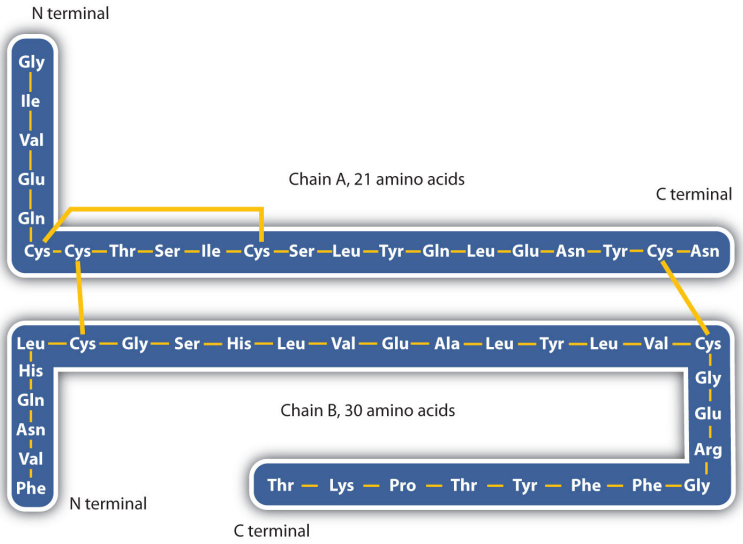

Figure 2.1: Primary Structure of Human Insulin

Secondary structure refers to highly regular local sub-structures on the actual polypeptide backbone chain. Two main types of secondary structure, the Alpha-helix and the Beta-strand or Beta-sheets, were suggested in 1951 by Linus Pauling[1] and coworkers. These secondary structures are defined by patterns of hydrogen bonds between the main-chain peptide groups. They have a regular geometry, being constrained to specific values of the dihedral angles and on the Ramachandran plot. Both the Alpha-helix and the Beta-sheet represent a way of saturating all the hydrogen bond donors and acceptors in the peptide backbone. Some parts of the protein are ordered but do not form any regular structures. (Refer to Figure 2.2) [2]

Tertiary structure refers to the three-dimensional structure of monomeric and multimeric protein molecules. The Alpha-helices and Beta-pleated-sheets are folded into a compact globular structure. The folding is driven by the non-specific hydrophobic interactions, the burial of hydrophobic residues from water, but the structure is stable only when the parts of a protein domain are locked into place by specific tertiary interactions, such as salt bridges, hydrogen bonds,

[1] http://www.pnas.org/content/100/20/11207.full

[2] https://2012books.lardbucket.org/books/introduction-to-chemistry-general-organic-and-biological/s21-04-proteins.html

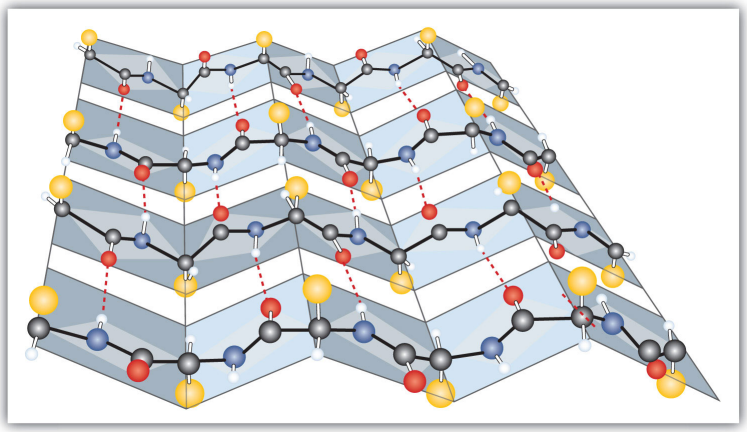

Figure 2.2: A Ball-and-Stick Model of the Beta-Pleated Sheet Structure in Proteins

and the tight packing of side chains and disulfide bonds. (Refer to Figure 2.3)[1]

Quaternary structure is the three-dimensional structure consisting of the aggregation of two or more individual polypeptide chains (subunits) that operate as a single functional unit. In this context, the quaternary structure is stabilized by the same non-covalent interactions and disulfide bonds as the tertiary structure. Complexes of two or more polypeptides (i.e. multiple subunits) are called multimers. Specifically it would be called a dimer if it contains two subunits, a trimer if it contains three subunits, a tetramer if it contains four subunits, and a pentamer if it contains five subunits. The subunits are frequently related to one another by symmetry operations, such as a 2-fold axis in a dimer. (Refer to Figure 2.4)

[1]https://2012books.lardbucket.org/books/introduction-to-chemistry-general-organic-and-biological/s21-04-proteins.html

Figure 2.3: A Ribbon Model of the Three-Dimensional Structure of Insulin

2.1 Hybrid Artificial Bee Colony and Particle Swarm Optimization

2.1.1 Introduction

Proteins are crucial in the biological processes, and their structure determines whether they can function well or not. Since the theory presented by Anfinsen that proteins space structure is entirely determined by the primary structure came out, it is possible for us to predict the structure of proteins through their primary structure without any experiment. In order to reach this target, the prediction problem can be formulated as an optimization problem that is set to find the lowest free energy conformation. In this paper[1], a hybrid Artificial Bee Colony (ABC) with Particle Swarm Optimization (PSO) Algorithm is used to solve this problem. Considering that the two algorithms have complementary characteristics, hence by combining them together results in better optimization results through this experimental approach.

[1]https://ieeexplore.ieee.org/document/6359433/

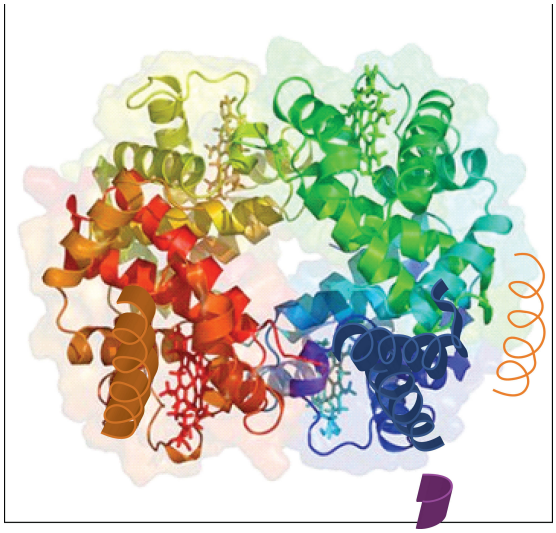

Figure 2.4: The Quaternary Structure of Hemoglobin

2.1.2 Advantages and Disadvantages

The ABC algorithm has several advantages compared with the other population based algorithms. The first advantage is that it is easy to use, highly flexible, robust and has a fast convergence. The second advantage is that it employs a fewer control parameters than the other algorithms such as Particle Swarm Optimization and Genetic Algorithm. The third advantage is that the ABC Algorithm hybridizes easily with the other optimization algorithms. The ABC algorithm also can handle stochastic cost objective function. However, the ABC algorithm has the disadvantages that it easily meets premature convergence in the later search period. The optimum value cannot be found sometimes. Hence the hybridization of both the Artificial Bee Colony (ABC) and Particle Swarm Optimization (PSO) Algorithm along with finding the lowest free energy conformation tackles several other disadvantages of both the algorithms separately so that the most efficient algorithm can be taken full advantage of.

2.1.3 Conclusion

This paper has proposed a new way to the protein secondary structure prediction, which is based on our proposed ABC-PSO model. Because of the complementary advantages of the two swarm intelligence algorithms, ABC-PSO has shown the superiority. The corresponding results of the experiments are also shown in this paper, which explain that ABC-PSO has a breakthrough in the short chain protein structure prediction.

2.2 Prediction of protein structural classes using Improved Support Vector Machine (ISVM)

2.2.1 Introduction

In general, the gap is broadening rapidly between the number of known protein sequences and the number of known protein structural classes. To overcome this crisis, it is essential to develop a computational prediction method for fast and precisely determining the protein structural class. Based on the predicted secondary structure information, the protein structural classes are predicted. To evaluate the performance of the proposed algorithm with the existing algorithms, four datasets, namely 25PDB, 1189, D640 and FC699 are used. In this work[1], an Improved Support Vector Machine (ISVM) is proposed to predict the protein structural classes. The comparison of results indicates that Improved Support Vector Machine (ISVM) predicts more accurate protein structural class than the existing algorithms.

The existing algorithms namely Support Vector Machine (SVM), Nave Bayes and the proposed Improved Support Vector Machine (ISVM) algorithms are compared to predict the protein structural classes. From the results produced, it is inferred that the ISVM algorithm provides better results than the existing algo-

[1]https://ieeexplore.ieee.org/document/7894709/

rithms.

2.2.2 Conclusion

Prediction of protein structural class for low-similarity sequences continues to be a challenging problem in Bioinformatics. Hence, the ISVM is proposed to further improve prediction accuracy. In the first step, the PSSM profile is converted into a fixed-length feature vector. Next, these features are ranked by ISVM based on their importance and the optimum top K features are input to an SVM classifier to perform the prediction. Four low similarity benchmark datasets are used to evaluate the performance of the proposed algorithm. The proposed algorithm obtains overall accuracies of 90%, 92%, 94% and 90% on D640, 1189, 25PDB and FC699 datasets, respectively. From the experimental results, it is clear that the proposed method significantly outperforms most of the existing methods.

2.3 Parallel Ant Colony Optimization using the HP Lattice Model

2.3.1 Introduction

The Protein Folding Problem studies the way in which a protein - a chain of amino acids - will 'fold' into its natural state. Predicting the way in which various proteins fold can be fundamental in developing treatments of diseases such as Alzheimer's and Cystic Fibrosis. Classical solutions to calculating the nal conformation of a protein structure are resource-intensive. The Hydrophobic-Hydrophilic (HP) method is one way of simplifying the problem. We introduce a novel method of solving the HP protein folding problem in both two and three dimensions using Ant Colony Optimizations and a distributed programming paradigm[1]. Tests Across a small number of processors indicate that the multiple colony distributed ACO (MACO) approach is scalable and outperforms

[1]https://ieeexplore.ieee.org/document/1420085/

single colony implementations. The Hydrophobic-Hydrophilic (HP) lattice was developed to simplify the problem whilst maintaining behavioral relevance. Until recently the 3D HP model has been largely ignored in favour of the 2D HP model due to the simplicity of the search space and easy visualization. The simplified HP models have been proven to be NP-Complete thus providing a perfect platform for evaluating and improving heuristic based algorithms that will assist future development of expanded protein folding problems. Much of the previous work in this field focused on developing and implementing solutions to 2D protein folding without considering 3D implementations. The paper proposes an algorithm that can solve the 3D HP protein folding problem by extending the solution to the 2D HP problem.

2.3.2 Conclusion

In running single processor implementations we terminated executing the test once no further improvements in the solutions were found. Both Multiple colony implementations outperformed the single colony implementation across five processors by a large margin. The tests run across five processors for multiple colonies resulted in execution times of less than a second. The use of multiprocessors on these test cases were not tested thoroughly. The framework developed for testing variable number of colonies for ACO for the HP folding problem shows acceptable 2D solutions which can be extended to 3D cases. Multiple colony solutions offer a definite improvement over the single colony implementation.

2.4 Naive Bayes with Feature Vectors

Among the various other approaches, the use of Naive Bayes classifier along with Feature Vectors was a standout as the Features provide a higher accuracy than other approaches. This has been proven by A. Chinmay, W. K. Sung who have compared and contrasted the use of Feature Vector selection along with various

machine learning algorithms likes BAYESPROT and SVM.[2] Feature Vectors are a more accurate measure for prediction as the features are directly correlated to the structure of the protein. This method allowed us to use our own Feature Vector in order to improve the accuracy of the predicted classes. We also used various other classifiers which provide a better understanding of which classifier works best with our Feature Vector. Table 2.1 shows the comparision of BAYESPROT and SVM using Feature Vectors.

Test Dataset		
Methods	**Accuracy(%)**	**No. of Classifiers**
BAYESPROT	58.8	3
SVM	45.2	2160
Cross Validation		
Methods	**Accuracy(%)**	**No. of Classifiers**
BAYESPROT	59.77	30
SVM	45.4	84240

Table 2.1: Comparative Results of BAYESPROT and SVM using Feature Vectors

[2]https://pdfs.semanticscholar.org/14e8/b8720a22b62aa81b39d510a954c0adbdccbe.pdf

Chapter 3

System Analysis and Design

3.1 Overview

In machine learning and statistics, classification is the problem of identifying to which of a set of categories a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known. An example would be assigning a given vehicle into bike or car classes or assigning a diagnosis to a given patient as described by observed characteristics of the patient (gender, blood pressure, presence or absence of certain symptoms, etc.).

Often, the individual observations are analyzed into a set of quantifiable properties, known variously as explanatory variables or features. These properties may variously be categorical - e.g. "A", "B", "AB" or "O", for blood type, ordinal - e.g. "large", "medium" or "small", integer-valued - e.g. the number of occurrences of a particular word in an email, real-valued (e.g. a measurement of blood pressure). Other classifiers work by comparing observations to previous observations by means of a similarity or distance function.

An algorithm that implements classification, especially in a concrete implementation, is known as a classifier. The term "classifier" also refers to the mathematical function, implemented by a classification algorithm, that maps input data to a category.

We shall now discuss three Classification techniques that we have used in brief.

Naive Bayes is a classification technique based on Bayes Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. Even if these features depend on each other or upon the existence of the other features, all of these properties independently contribute to the probability. Naive Bayes model is easy to build and particularly useful for very large data sets.

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples. In two dimensional space this hyperplane is a line dividing a plane in two parts where in each class lay in either side.

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees habit of over fitting to their training set.
Compared with other classification techniques, there are three advantages -

- For applications in classification problems, Random Forest algorithm will avoid the overfitting problem.
- For both classification and regression task, the same random forest algorithm can be used.
- The Random Forest algorithm can be used for identifying the most important features from the training dataset, in other words, feature engineering.

Lets begin with what System Analysis and Design is about and what it should contain.

3.2 System Analysis and Design

Systems Analysis and Design is an active field in which analysts learn novel approaches and various techniques for building a system more effectively and efficiently. The primary objective of systems analysis and design is to improve and optimize the development of the system. System development life cycle is a process which includes multiple phases such as planning, analysis, design, deployment, and maintenance. It starts from requirement analysis and goes till system implementation and maintenance.

3.2.1 System Analysis

System Analysis is the process of collecting and interpreting facts, identifying the problems, and decomposition of a system into its components.

System analysis consisted of finding the most optimal classification algorithms, the classifiers chosen need to provide the best results for the feature vectors that we have selected. We chose Support Vector Machines, Random forest and Naive Bayes over other classifiers like K-Means and Linear Regression; these classifiers are not optimal for our approach as they favor unsupervised learning.

3.2.2 System Design

System Design is a process of planning a new business system or replacing an existing one by defining its components or modules to satisfy the specific requirements. We started off by data collection which is parallelized to improve performance. Further the application can process three different classifiers simultaneously. The feature vector prediction requires attributes to be selected carefully, our literature survey helped us in picking the right attributes.

3.3 System Development Life Cycle

System Development Life Cycle (SDLC) is a conceptual model which includes policies and procedures for developing or altering systems throughout their life cycles. (Refer to Figure 3.1) SDLC includes the following activities:

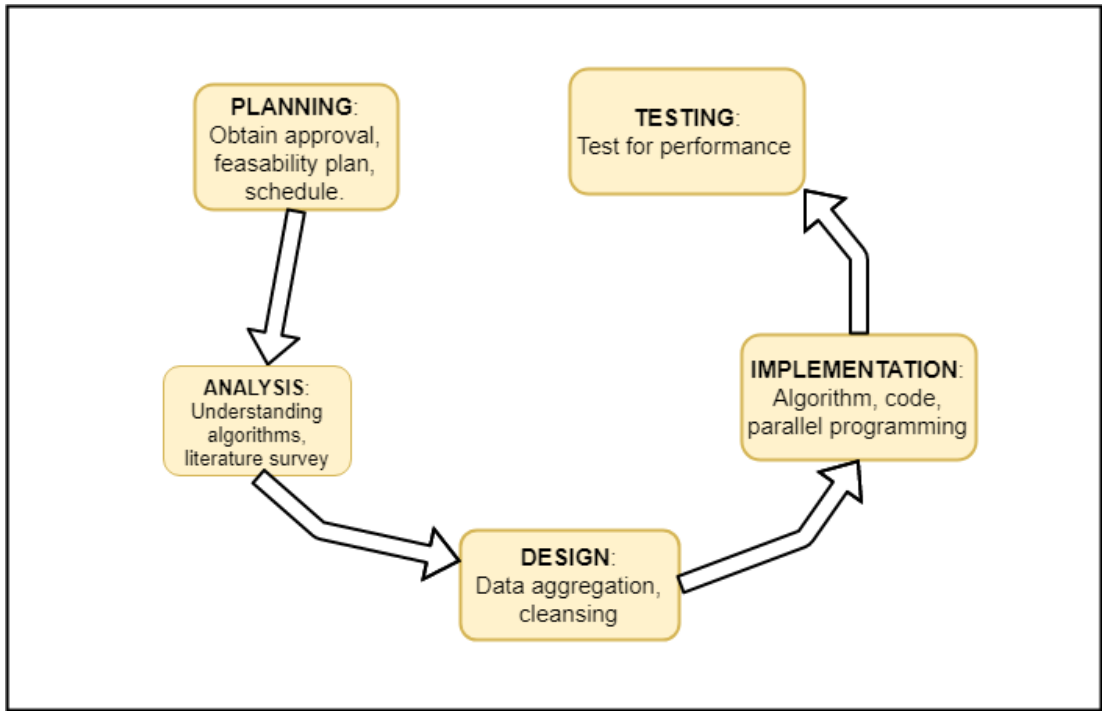

Figure 3.1: SDLC Phases

- Requirements
- Design
- Implementation
- Testing
- Deployment
- Operations
- Maintenance

For our system, we have followed a Gantt Chart which tries to qualitatively distribute the above tasks, as shown in Figure 3.2. The legend for the same is shown in Figure 3.3.

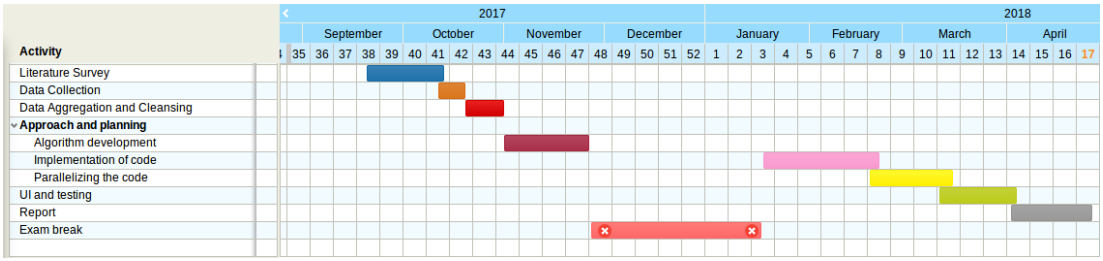

Figure 3.2: Gantt Chart

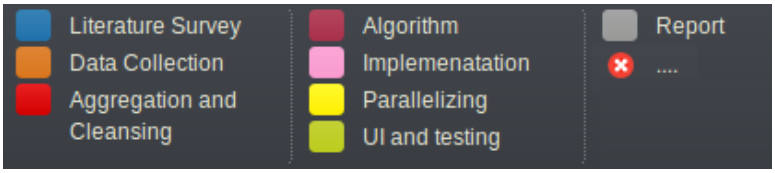

Figure 3.3: Gantt Chart Legend

3.4 Our System

From this section, we move on from the definitions of System Analysis and Design and delve into content which is specific to our system. We go in detail, starting from requirements and ending at Data Collection.

3.4.1 System Description

Before we get to the requirements, lets look at exactly what our system is trying to address. In System Description, we detail the Context of our System, its Characteristics, Assumptions, Constraints and Dependencies.

3.4.1.1 Context of System

The front end of the system was designed to simplify the ease of access to the user. It is interactive and simple to work with. Users provide the primary sequence to be classified

Coming to the backend, our application can run various classifiers based on user inputs. The classifier algorithms currently in use are Naive Bayes, Random Forest and Support Vector Machines. The novel features we add to the system is the parallel computation where a sequence can be tested for all three algorithms at

once, along with the use of Feature Vectors which are used as inputs to these classifiers.

This forms the context of our application.

3.4.1.2 System Characteristics

The main characteristics of our system include accuracy and precision of our application on the back-end. The data collection was also parallelised in order to reduce the time it takes to gather the supervised data set. We are able to achieve an efficiency of $O(n^2)$.

3.4.1.3 System Assumptions, Constraints and Dependencies

System Assumptions:

- The sequence should be in FASTA formats, limited to 21 amino acid sequences.
- The sequence entered should be a protein sequence and not a DNA or MRNA sequence.

System Constraints:

- Minimum size of the sequence should be 20 characters in length.
- Maximum size of the sequence should be 1231 characters in length.

System Dependencies:

- The application requires a system with multicore processing for the use of multiple threads.

3.4.2 Requirements Determination

The requirements for our system range over multiple domains. To simplify this, we have divided our requirements into two types, Functional Requirements and User Interface Requirements.

3.4.2.1 Functional Requirements

The Functional Requirements include the implementation of three different classifier algorithms which have to run parallely.

- A web scraper program which runs parallel to collect supervised data set. Which uses GOR IV structure prediction method.[1]
- Feature vector selection to improve accuracy of the classifiers.
- Algorithm to parallely trigger three classifiers.
- Results of the classifiers to be shown to the user.
- Presenting the generated feature vector to the user.

3.4.2.2 User Interface Requirements

- Simple user interface with textbox to accept primary protein sequence.
- Radio button to select a specific classifier.
- An option to run all three classifiers parallely.
- An option to display predictions of all the classifiers at the same time or individually.

3.4.3 System Design

We have designed a system which is shown in Figure 3.4 and Figure 3.5. The workflow of the system is a monolithic architecture. It describes a single tiered software application in which the user interface and data access code are combined into a single program from a single platform.
Monolithic applications are self contained and independent from other computing applications. The application is responsible for performing all necessary steps to complete a function. The system depends on a private data silo for all its data requirements instead of a large system or applications that work together.

[1] https://npsa-prabi.ibcp.fr/cgi-bin/npsa_automat.pl?page=/NPSA/npsa_gor4.html

The data set comprises of eight thousand cleansed sequences stored locally. The user interface takes a protein sequence as input and based on the feature vectors a scoring matrix is generated. The Feature Vector is used as input for the classifiers.
The predicted secondary sequence is displayed to the user. In case of multiple classifiers being used, all the predicted sequences are displayed to the user.

3.4.4 Components of the System

In this section, we go in detail about the components of the system, The components we will be discussing are listed below:

- **Java™ Application**
- **Generating Feature Vectors**
- **Algorithm**

3.4.4.1 Java™ Application

The java application is made using Swing. It acts as the front end of the system. The application is designed to work with an offline dataset without accessing internet.
There are two UI screens in the application:

- **Input Screen:** The input screen is displayed on startup. It consists of a textbox where the sequence is entered by the user, the entered text is validated for length and correctness. The text can be entered in upper case and lower case.
  The application lists three classifiers, namely Naive Bayes, Random Forest and Support Vector Machines. The user can pick any of these by using radio buttons and the selected classifier is run by the application. There is also an option to run all the classifiers parallelly, this returns the predicted output by all the classifiers at once.

Figure 3.6 and 3.7 shows the input screen from our application.

– **Output Screen:** The output screen displays the feature vector for the primary sequence (Refer to Figure 3.8) that the user has input to the application. The vector shown is used as the input to the classifier(s).
The predicted class labels are displayed recursively in the Output Section (Refer to Figure 3.9).

3.4.4.2 Generating Feature Vector

A feature vector is an n-dimensional vector having numerical values representing some object.
Our implementation involves a single dimensional feature vector that is unique to each protein sequence. A vector of length 105 is generated for each protein sequence based on values of certain physical and chemical properties. Our approach thereby converts a protein sequence of heterogeneous length to a feature vector of homogeneous length.
The twenty amino acids are segregated into three groups based on their values corresponding to the properties. The physical and chemical properties taken into account are Polarizability, Polarity, Hydrophobicity and Van Der Waals Volume. The twenty amino acids are grouped into three groups corresponding to their values for these properties.

The feature vector is made up of separate individual feature vectors that are as follows :

– **Composition Feature Vector** - The composition feature vector is calculated using

$$Ci = ((ni)/L)100;$$

where Ci represents the percent composition of each group

- **Transition Feature Vector (Tij)** - is represented by the percent frequency with which group i is followed by group j or vice versa where i, j take the values 1, 2 or 3 respectively.
- **Distribution Feature Vector** - The Distribution feature vector comprises of five numbers for each of the three groups, that are the fractions of the entire sequence, where the first residue of a given group is located, and where 25%, 50%, 75%, and 100% of those are located.
- **Percentage Frequency Feature Vector** - The Percentage Frequency feature vector is of length 20 and lists out the proportion of the 20 amino acids in the given protein sequence.
- **Calculations for Feature Vectors** - Each property has
  - 3 composition values with 4 properties - 3 x 4 = 12
  - 3 transition values with 4 properties - 3 x 4 = 12
  - 15 transition values with 4 properties - 15 x 4 = 60
  - Percentage frequency of each amino acid - 20 x 1 = 20
  - Length of protein - 1

  Feature Vector - 12 + 12 + 60 + 20 + 1 = 105

3.4.4.3 Algorithm

Algorithm **??** shows the Feature Vector Scoring process. The algorithm considers the following attributes to compute the vector (Refer Figure 3.10) :

- Hydrophobicity
- Polarizability
- Polarity
- Van der Walls Volumme

Algorithm 1: Feature Vector Scoring

Input: Primary Protein Sequence
Output: A vector score for the Sequence

```
1  procedure findFeatureVector(sequenceSQ)
2  begin
3      for each property do
4          divideAA ⟶ 3Groups
5          calcComposition(SQ) ⟶ returncompArr
6          calcTransition(SQ) ⟶ returntransArr
7          calcDistribution(SQ) ⟶ returndistArr
8      for each AA do
9          calcfrequencyofAAinSQ
10         returnPFArr
11     for each in array do
12         fv[]+ = array
13         returnfv
14     end
```

3.4.4.4 Classifier Algorithms

The following are the classifier algorithms we have used for -

- **Support Vector Machine Classifier** (Algorithm 3)
- **Naive Bayes Classifer** (Algorithm 4)
- **Random Forest Classifier** (Algorithm 5)

Algorithm 3: Support Vector Machines

Input: Feature Vector of size 1 x 105

Output: Predicted Protein Structure

$x \in Rn$, n is the dimension of the feature vector;

$y \in AlphaHelix, RandomCoil, ExtendedStrand(Hyperplanes)$;

$w, b \Longrightarrow SVMParameters$;

begin

  for *FeatureVectorRepresentation* $\Longrightarrow$ *N - dimensional Vector Space* **do**

    $if(wTx(i) + b <= -1)$then y(i) = -1;

    $elseif(wTx(i) >= 1)$then y(i) = 1;

  $MaximizeDistance(y(i), y(i), ..), wherei = 0, 1, 2, .n.$;

  end ;

Algorithm 4: Naive Bayesian Classifier

Input: Feature Vector of size 1 x 105

Output: Predicted Protein Structure

D: Set of Tuples.

X = $(x_1, x_2, x_3., x_n)$, Where x_i is the value of attribute A_i.

Classes $C_1, C_2, C_3 \longrightarrow \{AlphaHelix, RadomCoilandExtendedStrand\}$

Bayesian classifier predicts X $\in class$C$_i$ iff,

$P(C_i \mid X) > P(C_j \mid X) for\ 1 \leq j \leq m, j \neq i$

Maximum Posteriori Hypothesis,

$P(y \mid X) = \frac{P(X|y)P(y)}{P(X)}$

where, y is class variable and X is a dependent feature vector (of size 105).

Gaussian, hence, conditional probability is given by:

$P(X_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} e^{-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}}$

Algorithm 5: Random Forest

Input: Feature Vector of size 1 x 105
Output: Predicted Protein Structure

```
1  TrainingModel()
2  begin
3      k ⟹ RandomSelectFeatures() (From m features, k << m)
4      d ⟹ BestSplit(k)
5      i ⟹ 0
6      while (i < n) (n!=1) do
7          d ⟹ BestSplit(d)
8          BuildForest(d, i)
9          PredictionFromTrainedModel()
10         targetOutcome[n] ⟹
             RandomDecisionTree(k){AlphaHelix, RandomCoil, EtendedStrand}
11         votes[m] ⟹ CalculateVotes(targetOutcome[i]){i = 0, 1, n =
             105}{m << n}
12         max = 0
13         i = 0
14     while (i < n) do
15         if(max < votes[j]{j = 0, 1, ..m})
16         max ⟹ votes[j]
17         returnmax
```

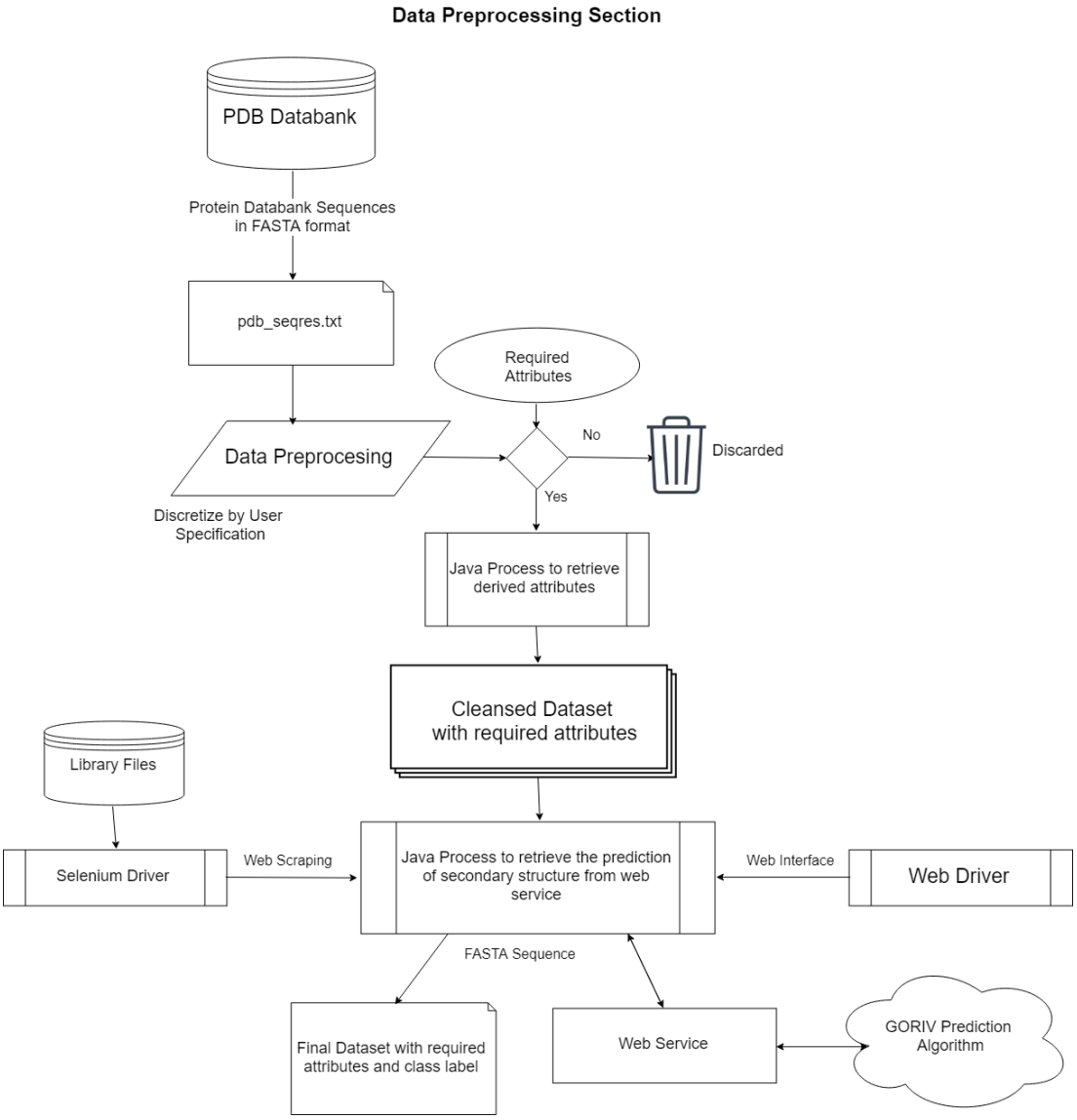

Figure 3.4: Data Preprocessing System Design

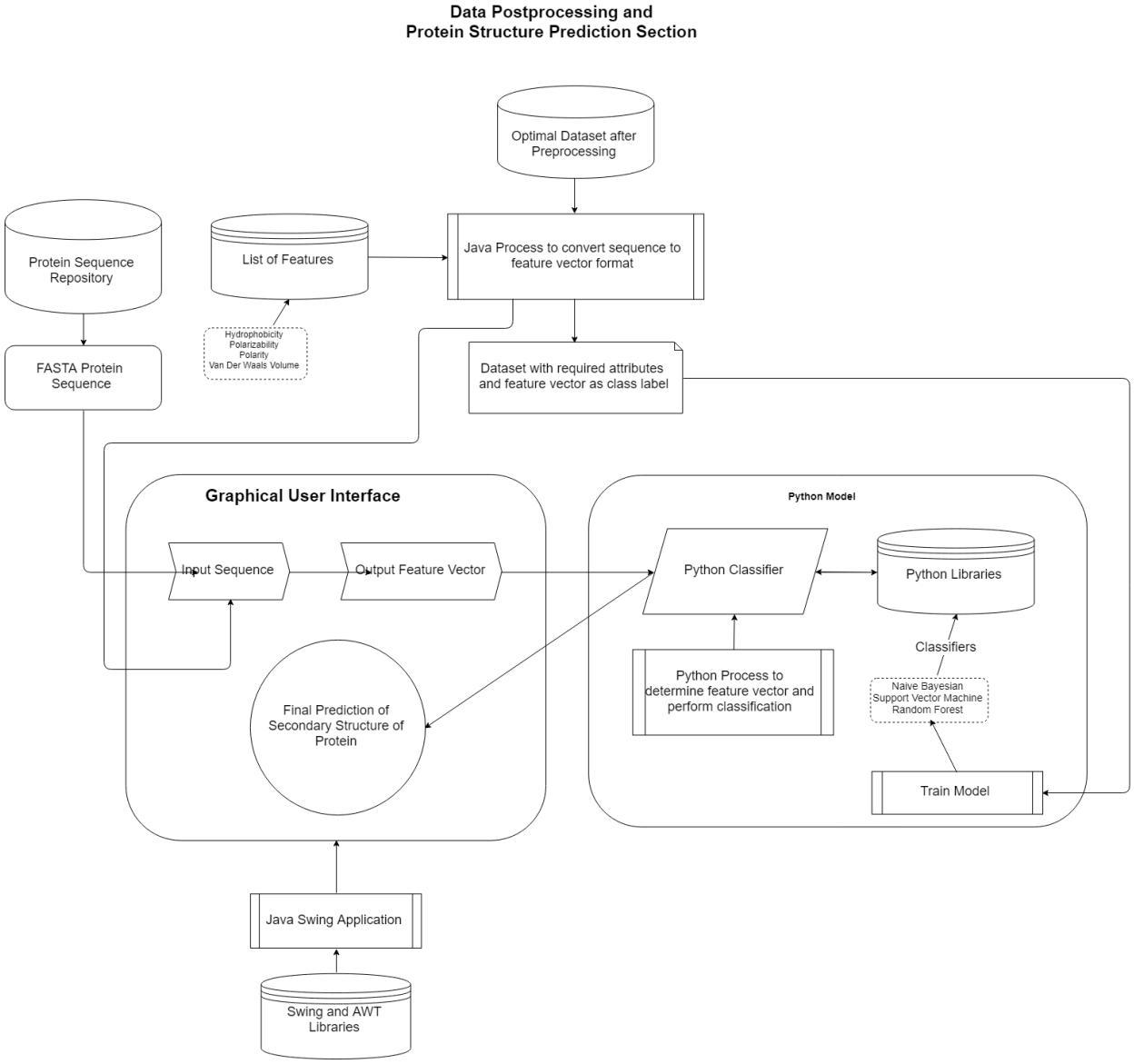

Figure 3.5: Data Postprocessing System Design

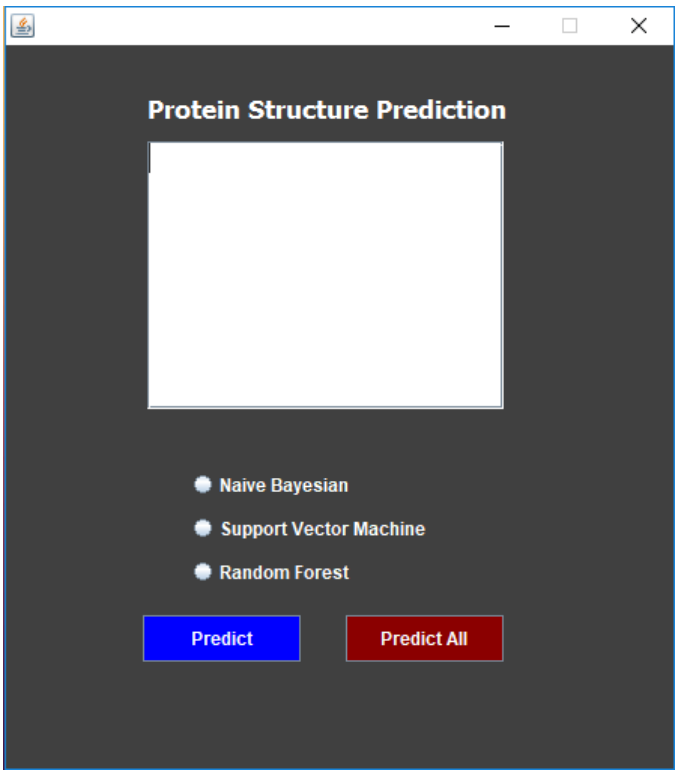

Figure 3.6: Screen 1 - Blank

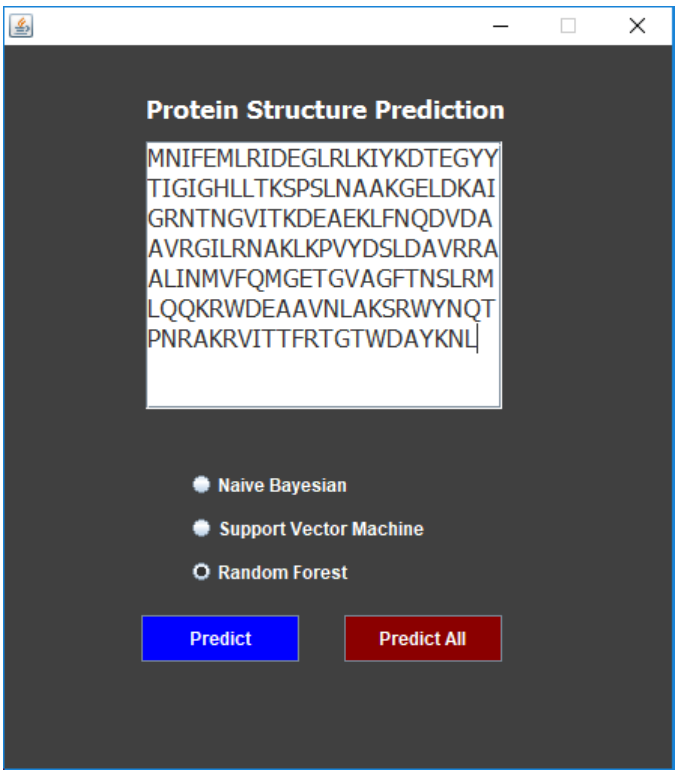

Figure 3.7: Screen 2 - with Sequence

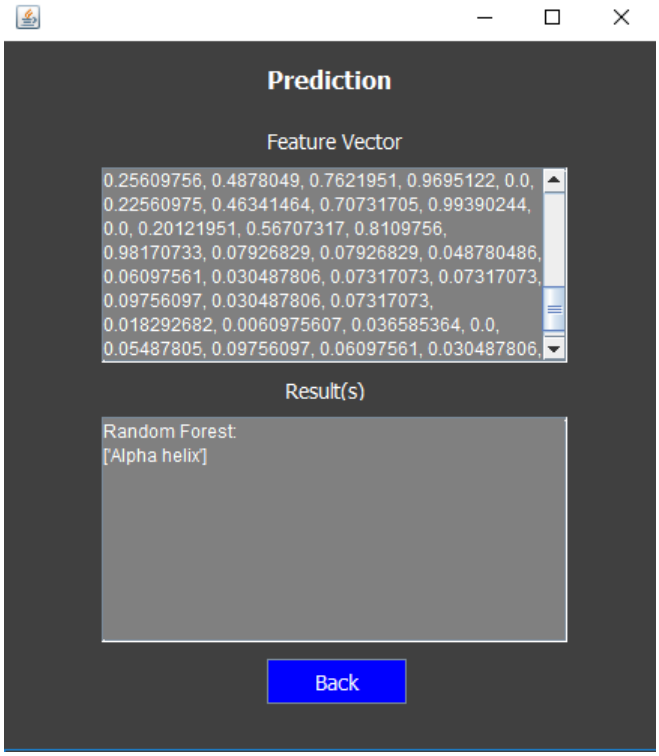

Figure 3.8: Screen 2 - output for Random Forest

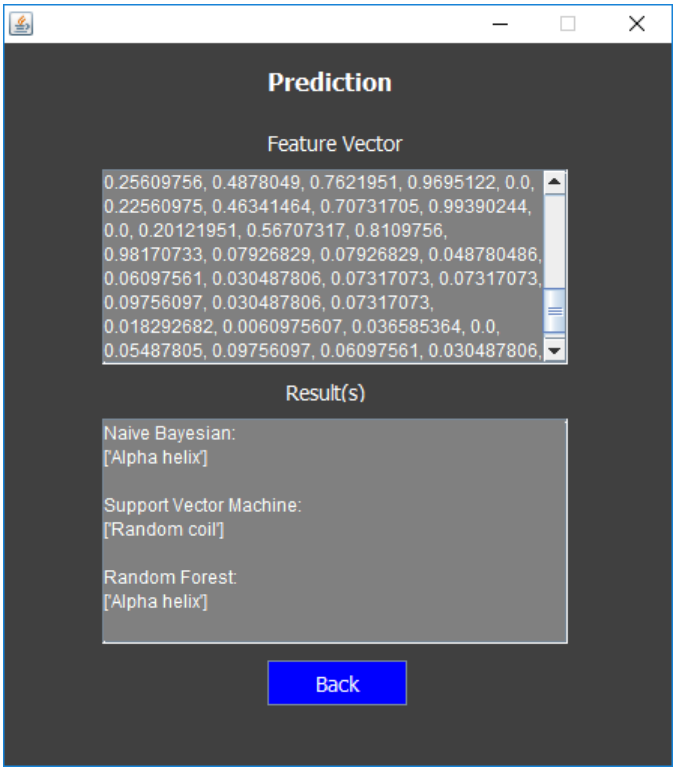

Figure 3.9: Screen 2 - output for all Classifiers

Attribute	Group 1	Group 2	Group 3
secondary structure	Helix	Strand	Coil
Hydrophobicity	Polar R,K,E,D,Q,N	Neutral G,A,S,T,P,H,Y	Hydrophobic C,V,L,I,M,F,W
Polarizability	(0-2.78) G,A,S,C,T,P,D	(2.95-4.0) N,V,E,Q,I,L	(4.43-8.08) M,H,K,F,R,Y,W
Polarity	(4.9-6.2) L,I,F,W,C,M,V,Y	(8.0-9.2) P,A,T,G,S	(10.4-13.0) H,Q,R,K,N,E,D
Van der Waals volume	(0-0.108) G,A,S,D,T	(0.128-0.186) C,P,N,V,E,Q,I,L	(0.219-0.409) K,M,H,F,R,Y,W

Figure 3.10: Feature Vector - Amino Acid attributes and corresponding groups

Chapter 4

Modeling and Implementation

4.1 Overview

From the previous section, we understood the requirements, design, workflow, accuracy and the data collected by the system. In this section, we describe in detail about how the system functions. We elaborate upon, the data model used, use case scenarios, workflow sequence, actual data transmission between channels.

4.2 Data Model

Data models define how the logical structure of a database is modeled. Data Models are fundamental entities to introduce abstraction in a DBMS. Data models define how data is connected to each other and how they are processed and stored inside the system.

4.2.1 Our Model

* **Initial Model** For our data model, we initially used an unsupervised data set from the Protein Data bank which needed a lot of

data preprocessing before actually using the data present in the data set. The data set contained about four hundred thousand sequences[1] which were a mixture of DNA, RNA and Protein sequences. We had to initially extract the protein sequences from the data set since those records were the only meaningful records for this project. The other parameters were Sequence ID, Sequence Type, Sequence Length, Sequence Name and the Primary Sequence itself. Our model requires the Class Label, the Secondary Structure of Protein. This unsupervised dataset had to be converted into a supervised dataset.

* **Intermediate Model** We researched online and found a web service that allowed us to extract the secondary structure from the primary sequence. This online web service displayed the results of the sequence with their frequency percentages and also whether they were a particular secondary structure type. The algorithm used isolated the amino acids and categorized them into particular secondary structure sequence. The percentages in the output is responsible for categorizing them into the final secondary structure.

  The particular web service took about ten seconds per sequence to receive the results if done manually. The dataset used had about eight thousand protein sequences to handle. So this manual process took around two hours for two hundred sequences. We had to automate this process so as to achieve any progress at all. We further used Selenium driver for web scraping. This allowed us to automate the retrieval process and increase efficiency of the conversion. The total time it took to retrieve and analyze all the sequence and to create a class label for each and every sequence out of the eight thousand or so protein sequences was about six to seven hours. This was a tremendous step forwards. Since this was not optimal performance, we converted the script so as to run on

[1] https://www.rcsb.org/

six different threads on six different Google Chrome tabs so as to achieve parallelization. The whole retrieval and data aggregation process took about two hours.

The script was written in Java so as to be platform independent and also because Selenium works best when coded on Java. The only changes that have to be made is that the appropriate Web-driver and Selenium driver must be used in tandem to work on machines other than Windows.

* **Final Model** After the conversion of the unsupervised dataset into a supervised data set, we now had to start the prediction process. We used three classifiers, namely Naive Bayesian, Support Vector Machine and Random Forest for predicting the secondary structure of protein. We chose these classifiers as they operate extremely different from each other and we wanted to figure out which was the best classifier to be used for our dataset with our particular input.

  The attributes sent to these classifiers were mainly the feature vectors and of course the class label. These classifiers take upto 75% of the dataset for training and the rest for testing to provide an insight of how accurate the predictions turn out to be.

  The Graphical User Interface uses the same classifiers but takes only one sequence at a time instead all the eight thousand sequences. There are options to either get predictions from all the classifiers or individually.

4.2.1.1 Dataset Format

For our dataset, we follow the following format.[1]

A sample dataset of protein sequences extracted from the protein data bank can be seen in Figure 4.1

– **Initial Dataset** - This data is directly obtained from the data bank. It requires preprocessing in order to be useful in sequence prediction.

[1] https://npsa-prabi.ibcp.fr/cgi-bin/npsa_automat.pl?page=/NPSA/npsa_gor4.html

```
TRANSDUCIN (GAMMA SUBUNIT)
PVINIEDLTEKDKLKMEVDQLKKEVTLERMLVSKCCEEFRDYVEE
RSGEDPLVKGIPEDKNPFKE
Random coil: 56.92
PHOSDUCIN
MEKAKSQSLEEDFEGQASHTGPKGVINDWRKFKLESEDSDSVAHS
KKEILRQMSSPQSRDDKDSKERFSRKMSVQEYELIHKDKEDENCL
RKYRRQCMQDMHQKLSFGPRYGFVYELESGEQFLETIEKEQKITT
IVVHIYEDGIKGCDALNSSLICLAAEYPMVKFCKIKASNTGAGDR
FSSDVLPTLLVYKGGELLSNFISVTEQLAEEFFTGDVESFLNEYG
LLPEKEMHVLEQTNMEEDME
Alpha helix: 45.31
```

Figure 4.1: Sample dataset

pdb_seqres.txt

101m_A mol:protein length:154 MYOGLOBIN

MVLSEGEWQLVLHVWAKVEADVAGHGQDILIRLFKSHPETLEKFD
RVKHLKTEAEMKASEDLKKHGVTVLTALGAILKKKGHHEAELKPL
AQSHATKHKIPIKYLEFISEAIIHVLHSRHPGNFGADAQGAMNKA
LELFRKDIAAKYKELGYQG

(Protein Sequence ID, Class, Molecule Type, Protein, Length of the sequence, Name of the sequence, Primary Protein Sequence)

– **Useful attributes**
Protein Sequence ID, Type, Primary Protein Sequence

– **Cleansed Dataset** - Considers only Protein Sequence after discarding RNA and DNA sequences from the data bank.

pdb_NS.txt

MYOGLOBIN

MVLSEGEWQLVLHVWAKVEADVAGHGQDILIRLFKSHPETLEKF

DRVKHLKTEAEMKASEDLKKHGVTVLTALGAILKKKGHHEAELK
PLAQSHATKHKIPIKYLEFISEAIIHVLHSRHPGNFGADAQGAMN
KALELFRKDIAAKYKELGYQG
(Name of the sequence, Primary Protein Sequence)

– **Intermediate Supervised Dataset** - Intermediate dataset is an unsupervised dataset obtained directly from the Protein Data Bank. It does not have a class label for the sequence of amino acids and various other parameters.

ContentTest.txt
MYOGLOBIN
MVLSEGEWQLVLHVWAKVEADVAGHGQDILIRLFKSHPETLEKF
DRVKHLKTEAEMKASEDLKKHGVTVLTALGAILKKKGHHEAELK
PLAQSHATKHKIPIKYLEFISEAIIHVLHSRHPGNFGADAQGAM
NKALELFRKDIAAKYKELGYQG
Alpha helix: 75.32
(Name of the sequence, Primary Protein Sequence, Class Label (Secondary Structure Prediction), Percentage of the Secondary Structure Percentage in the Sequence)

– **Supervised Dataset** - The supervised dataset has been created by web scraping from ¡enter here the website from where we scraped¿ and the class label obtained is used as the class label for the amino acid sequence which along with the sequence length forms our supervised dataset.

ContentTest.csv
MYOGLOBIN,MVLSEGEWQLVLHVWAKVEADVAGHGQDILIRLF
KSHPETLEKFDRVKHLKTEAEMKASEDLKKHGVTVLTALGAILK
KKGHHEAELKPLAQSHATKHKIPIKYLEFISEAIIHVLHSRHPG
NFGADAQGAMNKALELFRKDIAAKYKELGYQG,Alpha helix

Name of the sequence, Primary Structure Sequence, Class Label (Secondary Structure Prediction)

– **Feature Vector Input** - The feature vector of length 105 is constructed by taking into properties of the amino acid sequence which include Hydrophobicity, Polarity, Polarizability and Van Der Waals Volume. Individual Composition, Transition and Distribution values for each of the properties are generated and together with the sequence length and percentage frequency of each amino acid are combined to constitute the feature vector.

[0.25609756, 0.42682928, 0.31707317, 0.40243903, 0.34146342, 0.25609756, 0.37804878, 0.3292683, 0.29268292,.., 0.085365854, 0.085365854, 0.0121951215, 0.0121951215, 0.024390243, 82.0]

An array of floating point values that are fed as attributes to the classifier(s).

4.3 System Implementation

In this section, we will dive into the intricacies of our system. We begin with the UML diagrams which describe our system.

4.3.1 UML Diagrams

UML stands for Unified Modeling Language. For our system, we have created a Class Diagram, Use Case diagram and Sequence diagram.

For our use case diagram (Refer Figure 4.2) , we have one actor, the user. The user first gets a valid protein sequence for which the secondary structure has to be predicted. He then enters the same in the textbox provided in the GUI. Here he can either select one classifier

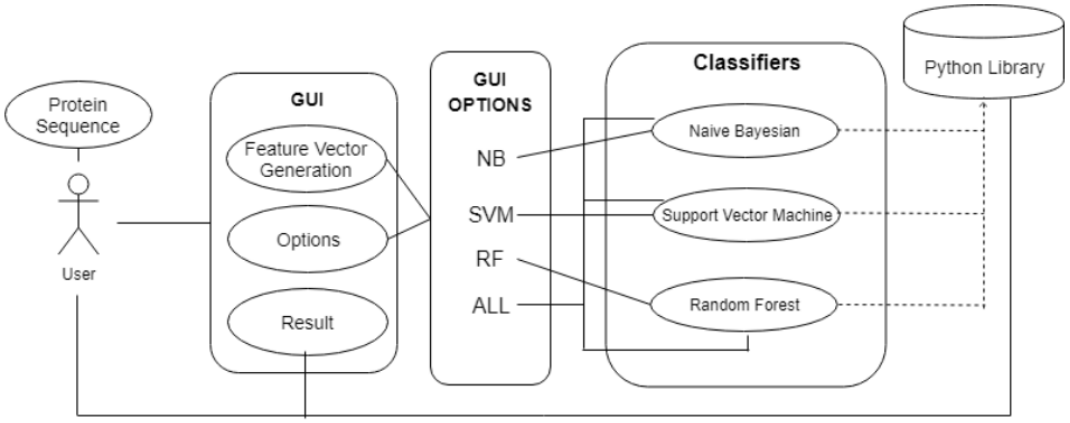

Figure 4.2: High Level Representation of System Design

to predict the structure or can get the predictions of all the three classifiers simultaneously.

Now the sequence, after validation, is converted into a vector which consists of floating point values. This vector is called a feature vector. The feature vector is then provided as input to the classifiers for prediction purposes. The vector is fed into the classifiers which use python libraries to provide the predicted result after executing their respective algorithms. The results are then received by the GUI which shows the feature vector as well as the predicted result for the given input sequence back to the user.

Now we will elaborate on the sequence diagram (Refer Figure 4.3). Initially the user has a protein sequence for which the prediction has to be made. This sequence is first validated by the GUI logic itself where the sequence cannot consist of characters other than the characters allotted to the list of twenty one amino acids. The sequence must also satisfy the length constraints, that is the length of the sequence must be a minimum of twenty and a maximum of one thousand two hundred

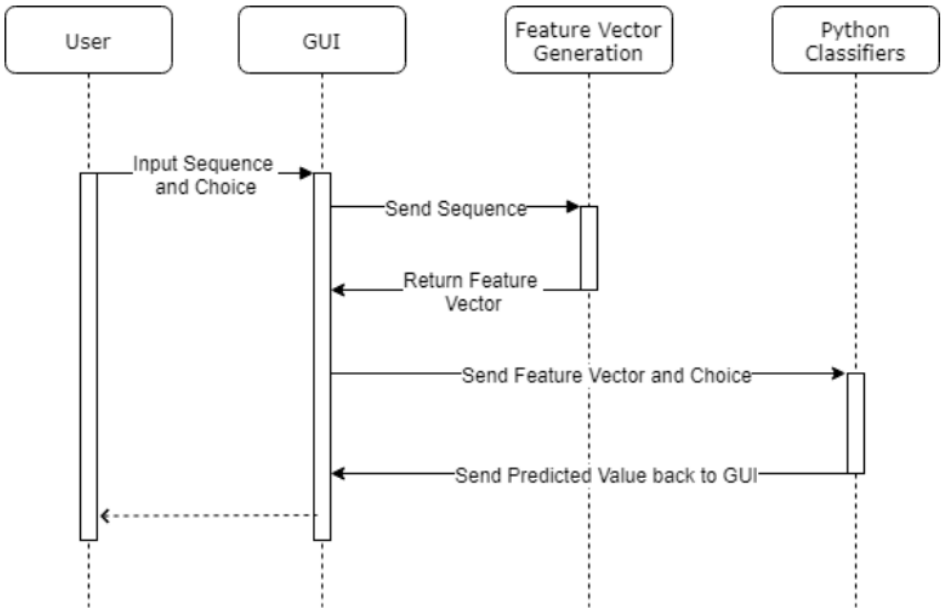

Figure 4.3: Sequence Diagram

and one. This sequence is then fed to the Java logic wherein a feature vector with floating point values is generated which will be the input to the Python Classifiers namely, Naive Bayesian, Support Vector Machine and Random Forest. The choice is also sent as a variable to the Java logic which is responsible for selecting the appropriate classifier.

The values fed into the classifier helps the algorithm of the particular classifier to classify the given feature vector into a specific secondary structure of protein which is then received by the GUI from which the user finally retrieves the data required.

Chapter 5

Results, Evaluation and Discussion

5.1 Overview

The heart of our application is our Feature Vector algorithm. The use of Feature Vector is to generate a Vector containing corresponding values to the chemical properties of the protein sequence. There is a lot of ongoing research in this domain. With our algorithm, take a step towards improving accuracy of predicting . Thus, it becomes essential to test our Feature Vector algorithm along with various classifiers.
We shall now discuss how the accuracy and results varied across the classifiers.

5.2 Gaussian Nave Bayers Classifier

In order to built the classifier for protein structure prediction our initial approach was to use the most popular classifier, Naive Bayers classifier. We chose Naive Bayers classifier because of its good performance in multi class prediction.

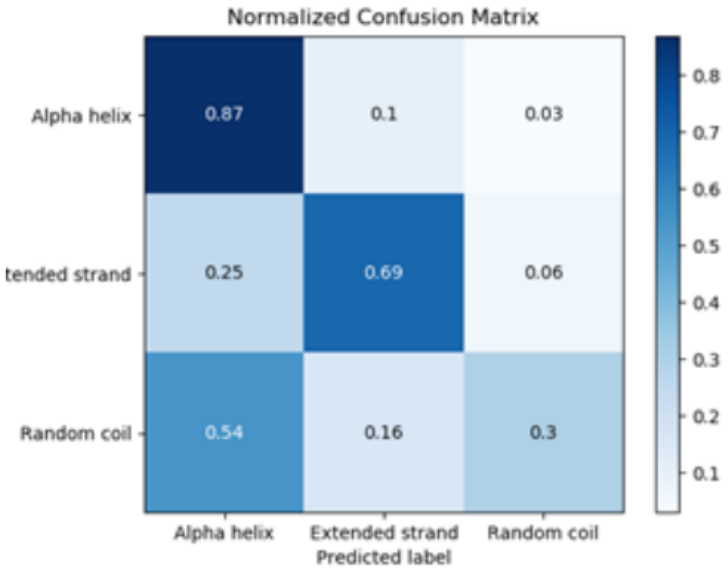

Figure 5.1: Normalized Confusion Matrix for Naive Bayes

The Confusion Matrix for the Gaussian Nave Bayers Classifier with the predicted values normalized between 0 and 1 is shown in Figure 5.1. From the matrix it can be analyzed that the classifier does a good work in predicting 87% of Alpha helix structure correctly. Also, the classifiers performance in predicting the Extended Strand sequences correctly is above average at 69%. However, when it comes to predicting the Random Coil sequences the classifier Performance is very bad with only 3 percent of the testing sequences predicted correctly.

5.2.1 Graphs of Dataset versus Accuracy and Precision for the Nave Bayers Classifier

The two graphs generated from the dataset (Refer 5.2 and 5.3) with splits of 60% of the dataset used for training in the first case and 75% of the dataset used for training in the other case. From the Graphs it is evident that the fluctuation in values of accuracy and precision is

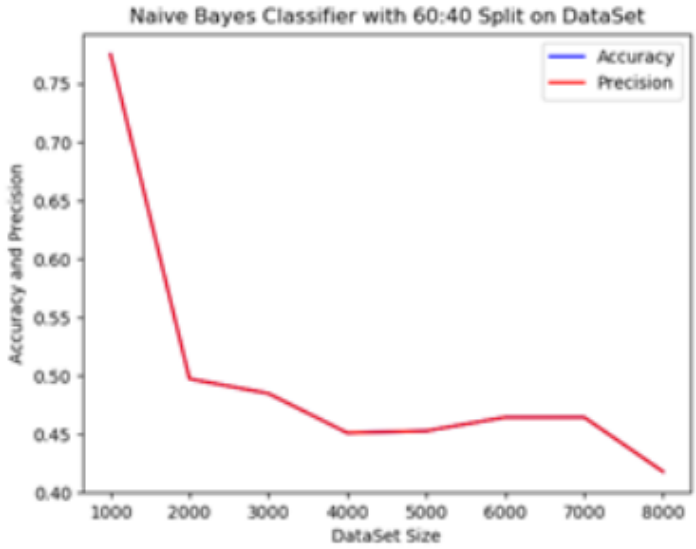

Figure 5.2: Results of Naive Bayes with 60:40 split on dataset

the same with the variation in the size of datasets. The Accuracy and precision both varies within the range size of 0.35 with the variation in the size of datasets from 1000 to 8000. The observed variations is against the expected behavior that the increase in size of training dataset should lead to increase in accuracy and precision. From this we can conclude that our Feature Vector does not suit well for the Gaussian Nave Bayers Classifier.

5.3 Support Vector Machine Classifier

The drawback of Naive Bayers classifier led us to the implementation of Support vector Machine(SVM) classifier. SVM is very effective in cases when the number of dimensions are very large. As a result, SVM was our next choice to handle our large dimensional feature vector.

The Confusion Matrix for the Support Vector Machine Classifier with the predicted values normalized between 0 and 1 is shown in Figure 5.4. From the matrix it can be analyzed that the classifier does an

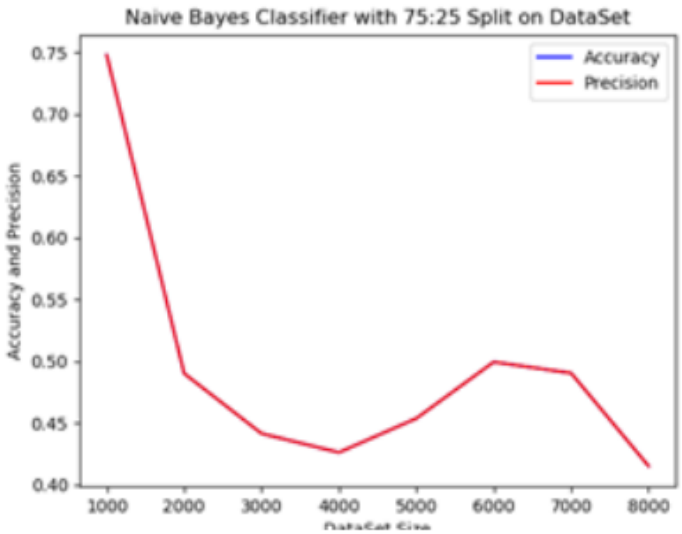

Figure 5.3: Results of Naive Bayes with 75:25 split on dataset

excellent work in predicting the structure of Extended Strand and Random Coil correctly for almost all the testing sequences. However, the performance of the classifier fails when it comes in predicting the structure of Alpha helix sequences. Consequently, only 28% of the testing sequences are predicted correctly as Alpha helix. With a overall accuracy of 76% SVM was a good classifier for our feature vector.

5.3.1 Graphs of Dataset versus Accuracy and Precision for the Support Vector Machine Classifier

The two graphs generated from the dataset (Refer 5.5 and 5.6) with splits of 60% of the dataset used for training in the first case and 75% of the dataset used for training in the other case.
The graphs explicate the variations in accuracy and precision with the increase in the size of dataset from 1000 to 8000. The Accuracy and precision both varies within range size of 0.02 with the variation in the size of dataset. This behavior is parallel to the expected behavior as both accuracy and precision increase with the increase in size

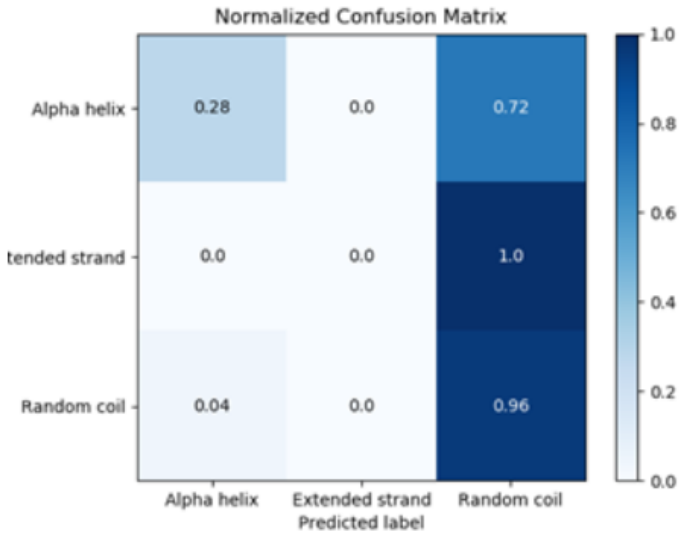

Figure 5.4: Normalized Confusion Matrix for Support Vector Machines

of dataset. Although there may be a slight decrease in the values of accuracy or precision with the increase in the dataset size, this may be neglected due to the reason that this small variations can sometimes occur as a result of overfitting. Overall, the classifier has a good accuracy of 76% and precision of 74%. Thus, SVM is a suitable classifier for our feature vector.

5.4 Random Forest Classifier

In our prospect for a better classifier to improve the accuracy we implemented Random Forest Classifier. In light of Random Forest Classifiers ability to deal with multiple features which may be correlated it would help us in handling our closely interrelated feature vectors. Thee Confusion Matrix for the Random Forest Classifier with the predicted values normalized between 0 and 1 is shown in Figure 5.7. From the matrix it can be analyzed that the classifier does an excellent work in correctly predicting all the three structures for almost all the testing sequences.

Figure 5.5: SVM Classifier with 60:40 split on DataSet

5.4.1 Graphs of Dataset versus Accuracy and Precision for the Random Forest Classifier

The two graphs generated from the dataset (Refer 5.8 and 5.9) with splits of 60% of the dataset used for training in the first case and 75% of the dataset used for training in the other case.
The Accuracy and precision both varies within range size of 0.03 with the variation in the size of dataset. From the graphs we can see that we have almost a linear relationship. This behavior sharply coincides with our expected behavior as both accuracy and precision increase with the increase in size of dataset. As a result, the classifier does a excellent work in predicting all the three structures. Overall, the classifier has a good accuracy of 96% and precision of 95%.
Thus, Random Forest is the most suitable classifier for our feature vector.

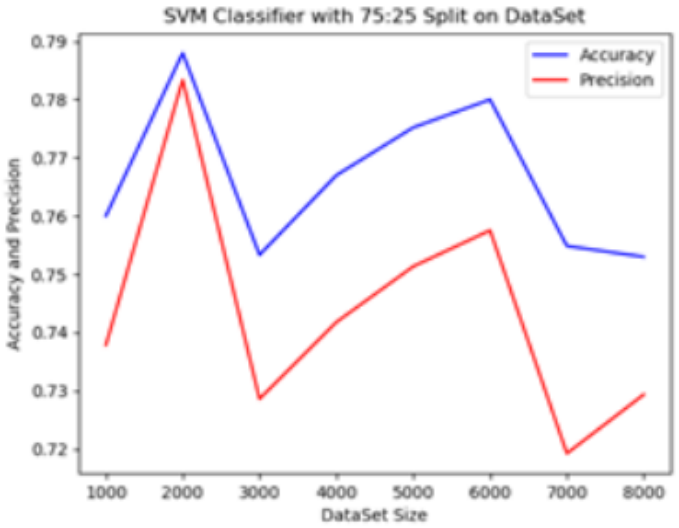

Figure 5.6: SVM Classifier with 75:25 split on DataSet

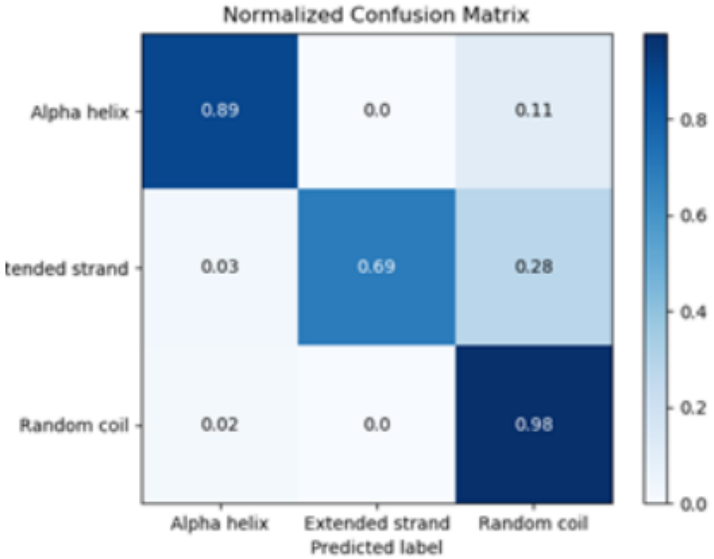

Figure 5.7: Normalized Confusion Matrix for Random Forest

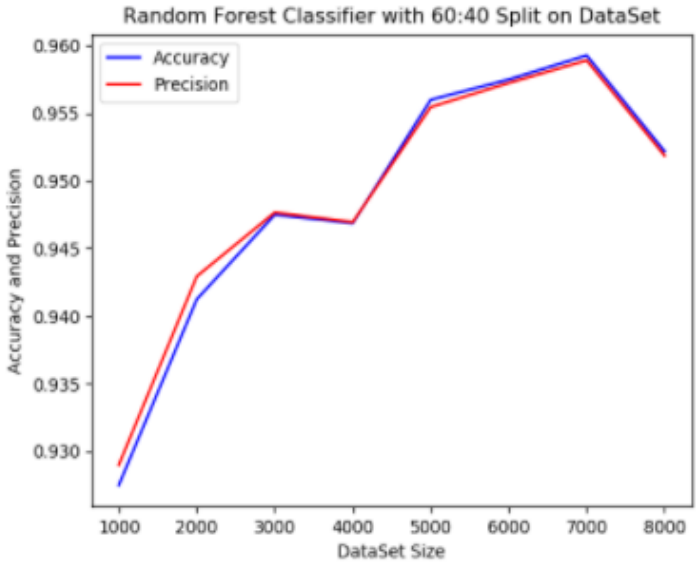

Figure 5.8: Random Forest Classifier with 60:40 split on DataSet

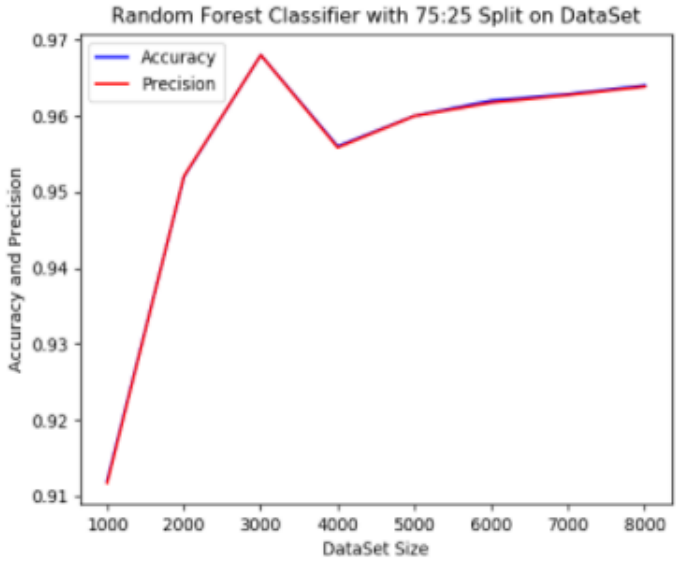

Figure 5.9: Random Forest Classifier with 75:24 split on DataSet

Chapter 6

Conclusion

We have built a Machine Learning model which predicts the two-dimensional structure of the protein from their amino acid sequences. Determining the three-dimensional structure of protein molecules is a primary focus for many aspects of modern biological research. Our model takes the primary protein sequence as input and outputs the three-dimensional structure to which the protein folds. This can be used in various Drug development which in particular requires knowledge of the binding sites of the candidate compounds, a well-predicted structure helps in the computational screening and optimizing candidate compound.

The unsupervised dataset to train our Machine Learning model is a linear chain of amino acids which forms the primary structure of the protein. We extracted information from various websites allowing us to get class labels for our unsupervised data set. We have generated a unique Feature Vectors for each protein sequences based on Polarity, Hydrophobicity, Polarizability and Van der Waals Volume. We then used various classification models which are used to analyze and predict class labels for each entered sequence.

Our initial approach of using Nave Bayesian classifier yielded an accuracy of approximately 47%. This low accuracy is the consequence of poor prediction of sequences which are Random Coil. In our endeavor to find a better classifier which suits our feature vector, we came up

Dataset Size	SVM		Naïve Bayer's		Random Forest	
	Accuracy	Precision	Accuracy	Precision	Accuracy	Precision
1000	0.773	0.745	0.775	0.775	0.930	0.929
2000	0.769	0.757	0.498	0.498	0.950	0.952
3000	0.758	0.730	0.485	0.485	0.944	0.944
4000	0.755	0.726	0.451	0.451	0.951	0.950
5000	0.780	0.757	0.453	0.453	0.950	0.950
6000	0.776	0.747	0.465	0.465	0.965	0.965
7000	0.760	0.727	0.465	0.465	0.957	0.957
8000	0.758	0.737	0.418	0.418	0.955	0.955

Figure 6.1: Accuracy and Precision of the three algorithms for 60:40 split

Dataset Size	SVM		Naïve Bayer's		Random Forest	
	Accuracy	Precision	Accuracy	Precision	Accuracy	Precision
1000	0.760	0.738	0.748	0.748	0.928	0.929
2000	0.788	0.783	0.490	0.490	0.960	0.960
3000	0.753	0.729	0.441	0.441	0.969	0.970
4000	0.767	0.742	0.426	0.426	0.961	0.961
5000	0.775	0.751	0.454	0.454	0.968	0.968
6000	0.780	0.758	0.499	0.499	0.962	0.962
7000	0.755	0.719	0.490	0.490	0.962	0.961
8000	0.753	0.729	0.415	0.415	0.963	0.962

Figure 6.2: Accuracy and Precision of the three algorithms for 75:25 split

with Support Vector Machine. SVM produced an overall accuracy of 78%. Hence showing competence with our developed feature vector. But still, the classifier fails when predicting the structure of most of the Alpha Helix sequences. Finally, the Random Forest classifier gives an excellent accuracy of 96% showing a strong capability of success with our developed Feature Vector. The accuracy and precision of the three splits are represented in Figure 6.1 and 6.2.

This Machine Learning model has been built and tested based on multiple classifiers. The most recent technologies have been used in the development of this application keeping in mind both the factors, efficiency and scalability.

Chapter 7

Future Work

The secondary structure of the protein gives a good indication of what the protein is majorly made up off. Weve developed a system with three classifiers that predict the secondary structure of protein and gives the bioinformaticians a one stop shop and access to different methods in predicting the actual structure that a sequence of amino acids would form.

The secondary structure is essential along with angles that the amino acids make with each other to give the final tertiary structure of the protein. This tertiary structure is the naturally occuring structure of the protein in nature. These various structures depend on the amino acids that form these chains and the twists and turns occur due to the interactions between these acids.

The tertiary structure of proteins is an important field is the drug making industry. Pharmacists and bioinformaticians study these structures and collaborate in making these drugs. Hence its extremely important to know minute details in these structures as minimal changes make a huge difference in the structure.

By getting the amino acid chains and passing it to the classifiers which have been trained on huge datasets the accuracy of certain classifiers has been found out to give better accuracy as compared to the others. Addition of other classifiers as well as implementation of the prediction of the tertiary structure of proteins in this system can make it

a comprehensive tool for bioinformaticians to predict the structures of proteins from their amino acids sequences and compare results of various machine learning algorithms for the same sequence.
The smallest of details can be added to classifiers to make them take into consideration all the factors that would result in a real world structure of the protein. Physical devices need not be used and precious time need not be wasted on identifying the structure of protein which is all taken care by the system trained on large datasets and working on a system running several machine learning algorithms simultaneously to make the human effort in prediction minimal.

Bibliography

1. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2673339

2. https://www.rcsb.org/

3. https://www.nobelprize.org/nobel_prizes/chemistry/laureates/1958/sanger-bio.html

4. https://2012books.lardbucket.org/books/introduction-to-chemistry-general-organic-and-biological/s21-04-proteins.html

5. https://npsa-prabi.ibcp.fr/cgi-bin/npsa_automat.pl?page=/NPSA/npsa_gor4.htm

6. Rinderknecht E, Humbel RE. Primary structure of human insulin-like growth factor II. FEBS Lett 1978; 89: 283286.

7. M. Li, H. Duan and D. Shi, "Hybrid Artificial Bee Colony and Particle Swarm Optimization Approach to Protein Secondary Structure Prediction," Proceedings of the 10th World Congress on Intelligent Control and Automation, Beijing, 2012, pp. 5040-5044. doi: 10.1109/WCICA.2012.6359433,

8. P. Manikandan and D. Ramyachitra, "Prediction of protein structural classes based on secondary structure sequence using Improved Support Vector Machine (ISVM)," 2016 IEEE Uttar Pradesh Section International Conference on Electrical, Computer and Electronics Engineering (UPCON), Varanasi, 2016, pp. 525-530. doi: 10.1109/UPCON.2016.7894709,

9. D. Chu, M. Till and A. Zomaya, "Parallel Ant Colony Optimization for 3D Protein Structure Prediction using the HP Lattice

Model," 19th IEEE International Parallel and Distributed Processing Symposium, 2005, pp. 193b-193b. doi: 10.1109/IPDPS.2005.326

10. R. King, M. Ouali, A. Strong, A. Aly, A. Elmaghraby, M. Kantardzic, D. Page, "Is it better to combine predictions?", Protein Engineering, vol. 13, pp. 15-19, 2000.