Gene Function Prediction Using Evolutionary K-Nearest Neighbor Algorithm

Hiroshi Madushani de Silva

148002J

Degree of Master of Science

Department of Computer Science and Engineering

University of Moratuwa

Sri Lanka

December 2017

Gene Function Prediction Using Evolutionary k- Nearest Neighbor Algorithm

Hiroshi Madushani de Silva

148002J

Thesis/Dissertation submitted in partial fulfillment of the requirements for the degree Master of Science in Research

Department of Computer Science and Engineering

University of Moratuwa

Sri Lanka

December 2017

Declaration

“I declare that this is my own work and this thesis/dissertation does not incorporate without acknowledgment any material previously submitted for a Degree or Diploma in any other University or Institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgment is made in the text.

Also, I hereby grant to University of Moratuwa the non-exclusive right to reproduce and distribute my thesis/dissertation, in whole or in part in print, electronic or another medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Signature: Date:

The supervisor/s should certify the thesis/dissertation with the following declaration.

The above candidate has carried out research for the Masters thesis Dissertation under my supervision.

Name of the supervisor:

Signature of the supervisor: Date :

Abstract

High-throughput gene annotation data are available in many popular model organism databases and repositories. These data are often incomplete and still evolving while the functions of the genes are unknown or partially known. As the manual curation process is costly and time-consuming, an in-silico method of predicting gene functions became a huge requirement in the industry of bioinformatics. Our approach is to use gene expression data that exist in data repositories rather than sequence data in order to predict the gene functions. In this paper, we have proposed a supervised machine learning algorithm combined with the genetic algorithm for function prediction. The k- Nearest Neighbor Algorithm is optimized using the genetic algorithm to find out the optimum k for a dataset. Also, the genetic algorithm gives a weight vector for the attributes in the dataset making an exceed performance of k- Nearest Neighbor Algorithm. GAKNN is a solution created for gene function prediction which analyze gene annotation data from different repositories and predict gene functions using the genetic algorithm optimized k- Nearest Neighbor classification algorithm. GAKNN provides a workspace for data pre-processing including data cleaning, feature selection, and missing data imputation followed by data analysis and data visualization. The software has been tested over two gene expression datasets from different sources to evaluate the accuracy. The datasets are from two different functional annotation schemes: Gene Ontology and FunCat. The data pre-processing methods available in GAKNN such as missing data imputation also tested with two gene expression datasets and results show that the use of Evolutionary k-Nearest Neighbor Imputation Algorithm gives better results than mean imputation and standard k- Nearest Neighbor Algorithm. The accuracies range from 60%- 88% in GAKNN for function prediction. The weights given for each attribute in the dataset and the optimum k by the genetic algorithm are also graphically represented in GAKNN.

Keywords: k- Nearest Neighbor, Genetic Algorithm, Gene Functions, Gene Annotation, Gene Expression Data

Acknowledgment

I express my gratitude to my parents for their immense dedication and especially I would like to dedicate every achievement of my life to my Mother Mrs. Sarojini de Silva.

My gratitude goes to my supervisor, Dr. Amal Shehan Perera who guided me throughout this research for his valuable assistance and patience.

I like to thank Dr. Rapthi de Silva, Dr. Chandana Gamage, and Dr. Chathura de Silva for their support and Dr. Chinthana Wimalasuriya, Dr. Surangika Ranathunge, and Prof. Nalin Wickramarachchi for their valuable feedback.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

GAKNN- Genetic Algorithm Optimized k- Nearest Neighbor

EvlkNN- Evolutionary k- Nearest Neighbor

kNNImputation- k- Nearest Neighbor Imputation

EvlkNNImputation- Evolutionary k- Nearest Neighbor Imputation

DNA- Deoxyribose Nucleic Acid

RNA- RiboNucleic Acid

GO- Gene Ontology

FunCat- Functional Catalog

CHAPTER 1

1. Introduction

This thesis provides a solution to predict functions of genes regardless of organisms using gene expression data. The solution is built upon the knowledge discovery process where the user is enabled to do pre-processing and optimization of data in order to predict the functional classes. The data pre-processing options in the solution includes feature selection and missing data imputation. Another part of the same study of predicting gene functions has a missing data imputation algorithm included in a data pre-processing step will be also discussed in detail in this thesis. A supervised learning algorithm is used to create a model as the initial step of prediction. The model is built upon a training dataset which has labeled functional classes in it. The trained model is then used as the base to do the predictions in the presence of uncertainty in gene expression data files. A detailed breakdown of the predicted gene functions by the solution gives the user a precise overview of accuracy for each functional class. The solution is developed for the convenience of biologists with the interpretation of each step using data visualization steps as much as possible using line charts and accuracy breakdown details including precision and recall for each functional class. The introduction to gene function prediction, the importance of identifying gene functions, the developed methodology, the data mining methods used in the solution, and the results will be discussed in detail in further chapters.

1.1. Gene Functions

The activities involved with genes are called gene functions. However, there are many categorizations/ definitions for the term “gene functions”. The gene functional hierarchical schemes available at present have assigned specific functional categories to each gene. Gene Ontology and FunCat are popular examples of functional categorization at present.

1.2. Problem Definition

Comprehensive genome databases provide several catalogs of information on genomes. There exists a potential for using gene annotation data, which includes phenotype, localization, protein class, complexes, enzyme catalogs, pathway information, chromosome location, and protein-protein interaction in predicting the functional class of genes. For biologists, it is practically useful to be able to know the functional class of genes via in-silico methods without having to conduct extensive and expensive biological experiments. In certain cases, it is useful to at least to know the probability of a certain gene's potential to be in a given functional class so that biologists can restrict the number of experiments they need to conduct.

The function prediction was done using sequence information over the years but the challenge was the sequence similarity alone could not predict the unknown functions of genes[1]. A significant number of genes in newly sequenced genomes used to reject the approach of sequence similarity in identifying functions. Therefore, sequence derived properties and additional information (genome context, protein structure information) were used to predict functions using machine learning algorithms[2]. As the number of sequenced genomes is also rapidly growing, the only way of annotating the gene products would be using a computerized approach.

GAKNN is a framework built upon the evolutionary k- nearest neighbor algorithm to predict gene functions using gene annotation data/ gene expression data stored in genome databases. GAKNN can be used to pre-process, optimize, predict, and analyze the gene functions of unlabeled functions in genes. GAKNN supports many functional category schemes including Gene Ontology and FunCat. Training a set of gene annotation data/ gene expression data with known function labels in it is the initial step of GAKNN where the trained model will be then used to do the prediction of unknown functions of genes. The learning algorithm is the genetic algorithm optimized k-nearest neighbor classification algorithm which falls into supervised machine learning.

1.3. Importance of predicting gene functions

Getting to know a function of a gene reveals many mysteries involved in a gene. The identified gene functions can be used to test the reaction of cells to drugs and ultimately leading to drug discovery. The predicted gene functions can be compared or tested to track the inheritance of a disease. Using the mutated genes alongside with healthy genes, we can identify the risks of developing a disease. Therefore, the identification of gene functions plays a vital role in deciding many medical involvements to come up with valuable discoveries and treatments.

1.4. Research Objectives

High throughput gene annotation data are available in many repositories like Gene Ontology, GeneBank, NCBI, and MIPS. These data are often incomplete and still evolving. A knowledge-based method that relies on these existing data can be used to predict gene functions. GAKNN is a computational solution which facilitates function prediction using any functional hierarchy (e.g- FunCat, Gene Ontology) for any species. The main objective of this research is to find gene functions of a relevant gene using the gene annotation data. Also, the time taken for manual curation process and detecting gene functions will be reduced by using a machine learning approach towards gene function prediction. Also, sequence alone cannot predict the functions of genes in many newly sequenced genomes where the functions are not known, our approach of using Genetic Algorithm Optimized k- Nearest Neighbor Classification for function prediction will be an efficient solution.

1.5. Gene Function Annotation

Once the genes are sequenced, it is vital that the identified genes are annotated in order to have a sense of what the genes are responsible for. Annotation is a comment or an explanation. Gene function annotations are associations between a gene and a term of a controlled vocabulary describing gene functional features. A gene annotation dataset consists of attributes such as

gene-id, symbol, name, chromosome, genomic position. An example attribute set of a dataset may look like the following:

{
"id": "1017"
"taxid": 9606,
"symbol": "CDK2",
"entrezgene": 1017,
"name": "cyclin-dependent kinase 2",
"genome_pos": {
"start": 56360553,
"chr": "12",
"end": 56366568,
"strand": 1
}
}

Most attributes/ fields are excluded from the example above and the full version can be accessed at http://mygene.info/v3/gene/1017. Gene annotation data can be obtained from online databases such as NCBI Entrez, Ensembl, UniProt, NetAffy, and UCSC.

1.6. Gene Expression Data

Microarray technology is widely used to analyze gene expressions. The raw microarray data are converted to gene expression matrices[3]. The rows of these matrices represent genes and the columns of the matrices represent various samples and the numbers in each cell characterize the expression level of the particular gene in the particular sample. This experiment of gene expression is a method to quantitatively measure the transcription phase of protein synthesis[4].

1.7. Gene Function Prediction

Among many other solutions for gene function prediction, GAKNN is a solution which only uses gene annotation data/ gene expression data to predict gene functions whereas many other solutions rely on gene sequence data to predict gene functions. There are data records with missing gene functions in online data repositories where we can use the data records with annotated gene functions to predict the unknown functional classes. As a solution to predict gene functions using the knowledge stored in repositories, GAKNN has been developed to analyze and predict the gene functions using gene annotation data/ gene expression data.

GAKNN is compatible with different file formats including .csv, .txt or .arff and with annotation hierarchies such as Gene Ontology and FunCat. Also, GAKNN follows the data mining process to predict functional classes of genes. Furthermore, GAKNN provides a workspace for data pre-processing, data transforming, data mining, data interpreting, and data evaluating.

Despite the general data mining solutions like WEKA, the software we built is specifically designed for function prediction with the compatibility of many functional schemes available at present. All the steps are implemented following a white box approach where the user can see each step of the prediction process. Starting with data pre-processing, the workspace in GAKNN provides the interface to upload and make required changes to the datasets. The pre-processing steps included are: deletion of data records and data attributes (if the records are sparse or not important in prediction), and missing data imputation. Then the pre-processed data file can be optimized using the genetic algorithm where the developed solution shows a graph with fitness scores for each iteration in the genetic algorithm. The fitness score in the genetic algorithm is a measurement to show how good the solution is. Also, the weights given by the genetic algorithm for each attribute can be seen in another graph where the user can do his or her own analysis determining when to stop optimizing based on the fitness scores and weights displayed in graphs. Finally, once the trained model is obtained, the user can then upload a test dataset where its gene functions are not known and to do the prediction based on the saved trained model. The

solution is developed in a user-friendly way for the biologists in predicting and confirming the experimental analysis of gene functions.

1.8. Research Contributions

- A solution which experimentally proves the mechanism of predicting gene functions using the GAKNN framework.
- A solution which is capable of predicting gene functions using gene annotation datasets in any functional category hierarchy.
- Missing data imputation for gene expression data using Genetic Algorithm Optimized k- Nearest Neighbor Imputation Algorithm.
- Evaluation of the importance of using a machine learning algorithm in imputation over mean/ median imputation.
- Analysis of gene annotation datasets with data pre-processing, data model training, and optimizing.
- Comparative analysis of the accuracy of datasets between WEKA and GAKNN.

1.9. Organization

The rest of the chapters are organized as follow. Chapter 2 is on the literature review of the research describing the structure of the cell, gene, gene functions to gene annotation, gene expression data, the importance of gene function prediction, existing methods of function prediction, introduction to genetic algorithm, and machine learning approaches of function prediction. Chapter 3 covers the methodology by introducing the proposed framework namely GAKNN. Chapter 4 illustrates the results obtained by the analysis of gene annotation data by GAKNN. Chapter 5 is the discussion of the analysis and experiments followed by the conclusion at the end of Chapter 6.

CHAPTER 2

2. Literature Review

2.1. Cell, gene, and gene functions

A cell is the fundamental unit of all organisms. The main unit is the cell nucleus which controls the functionality of the cell. The cell nucleus consists of many organelles. Among many organelles of the nucleus, there are some codes that carry out genetic information. These codes are called chromosomes which are made out of DeoxyriboNucleic Acid (DNA). Genes are located inside DNA. The gene is the functioning unit of heredity. Each chromosome contains many genes as illustrated in figure 1 below. A gene is a segment of DeoxyriboNucleic Acid (DNA) that specifies the sequence of nitrogen-based nucleobases. Usually, genes related to a trait such as an eye color, hair color, enzymes, hormones and other genetic information. An entire set of DNA of an organism is called a genome.

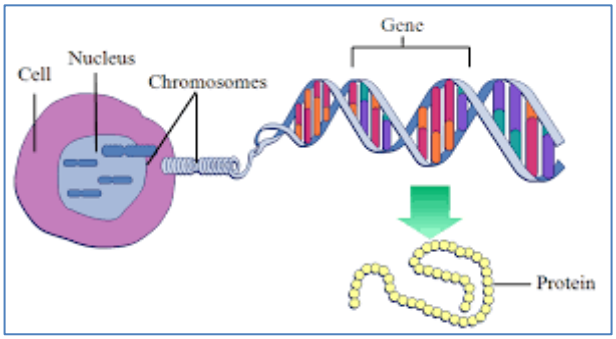

Figure 1: Illustration of a gene inside a chromosome

The fundamental building blocks of DNA are nucleotides which are made out of a sugar base, a phosphate group, and nitrogen-based nucleobases. There are four main nucleobases found in DNA such as Adenine, Guanine, Thymine, and Cytosine. As the goal of the pre-genomic era is to identify the order/ sequence of nucleobases in a particular gene, many projects initiated the process of gene sequencing.

As a gene is a sequence of nucleotides, there can be thousands of genes in one DNA molecule. A nucleotide sequence of a gene is determined by the start codon and end codon. A codon is three nucleotides in a row. The three nucleotides which specify the start is the start codon and the three nucleotides which specify the end of a gene is called the end codon.

DNA is mainly involved in protein synthesis. These proteins are the building blocks of structural elements in the cell. The protein synthesis has two main steps called transcription and translation. Transcription is the process of copying information resides in DNA to an RNA molecule. Translation is the process where the RNA with the copied information plugs with a ribosome and produce proteins from the codes in RNA. Also, a gene consists of two regions called intron and exon as shown in figure 2. During the transcript phase of protein synthesis, the removal of intron happens by a process called RNA splicing. The nucleotide sequences that are joined together after RNA splicing are called extrons[1]. The protein synthesis is explained below for the ease of understanding of gene functions.

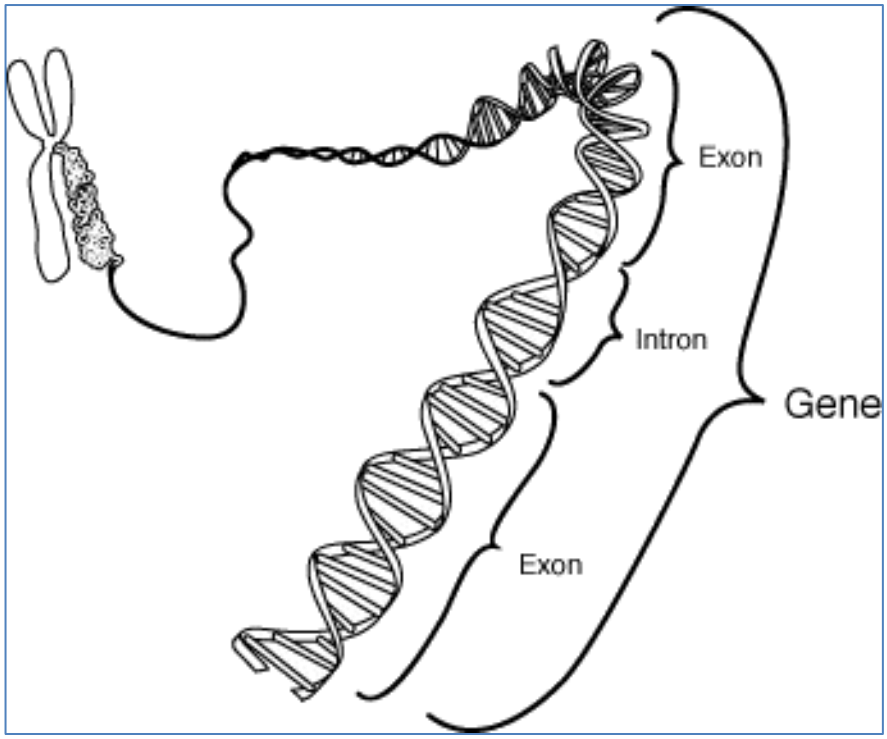

Figure 2: Illustration of Exon and intron of a gene

2.2. Protein Synthesis

The process of protein synthesis happens inside a cell nucleus which has two steps: transcription and translation. Transcription takes the information encoded in DNA and encodes it into mRNA (messenger RNA), which heads out of the cell into the cytoplasm of the cell. After that, translation begins where mRNA binds to a ribosome and tRNA (transfer RNA) to synthesize proteins.

2.2.1. Transcription

The first step in proteins synthesis is transcription. Transcription begins with the unwinding of two strands of DNA. DNA has two strands which give the double- helix structure for a DNA. Each strand consists of nucleotides such as Adenine, Guanine, Thymine, and Cytosine. Once the strand got unwounded, an enzyme called RNA polymerase helps line up nucleotides to create a complementary strand of mRNA. This mRNA is a single strand of the molecule which always combines with a single strand of DNA. This mRNA has the complementary nucleotides of DNA according to the rules of base pairing. Rules of base pairing are:

DNA Adenine pairs with RNA Uracil
DNA Guanine pairs with RNA Cytosine
DNA Thymine pairs with RNA Adenine
DNA Cytosine pairs with RNA Guanine

As an example, the complement of DNA strand “AATCG” in mRNA is “UUGGC”. After transcription, the mRNA will be released and the unwounded DNA will combine otherwise another mRNA might connect to the same portion of the unwounded DNA strand.

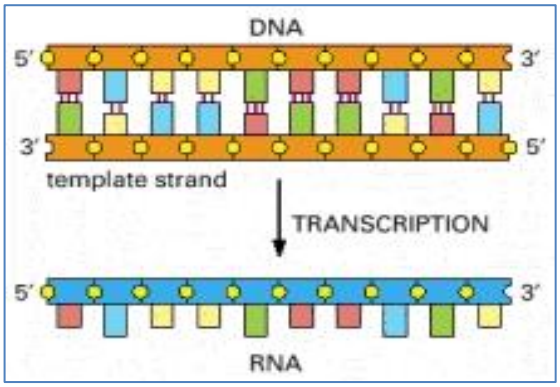

Figure 3: DNA Transcription

2.2.2. Translation

In translation, the mRNA is sent out to the cytoplasm from the nucleus. There are two main components need to complete the translation process. The first component is the ribosome. In the cytoplasm of the cell, the mRNA combines with a ribosome. The nucleotide sequence stored in mRNA is read back as three letter words (triplets) called codons. Each codon is an amino acid. Inside the ribosome, each amino acid is then combined and forms into a polypeptide chain. The second component is tRNA (transfer RNA). tRNA is a specialized RNA which carries an amino acid at one end and has a triplet of nucleotides and an anticodon at one end. The anticodons of tRNA will base-pair with one of the mRNA's codon. Hence the tRNA acts as the translator between the mRNA and protein by bringing the specific amino acid coded for by the mRNA codon.

2.3. Gene Annotation

There are many functional catalog hierarchies associated with gene function annotation. KEGG [5], Multifun [6], FunCat [7], and Gene Ontology[8], [9] are the most popular schemes in gene functional annotation at present. These catalogs have gene annotation data stored in a way where computerized methods can be used to annotate the unknown functions. Description of each scheme on gene functions is discussed in detail below. Gene functions have many categorizations which evolved many functional hierarchies available at present. Many functional classification schemes have adopted the idea of functional categories and built up the hierarchies for function annotation based on a categorical approach.

One such categorization suggested by Bork et al. [10] in 1998 is to gene function being categorized into molecular function, cellular function and phenotypic function. Soon after, many functional schemes were proposed including EcoSyc [11] which is very specific to *Escherichia Coli* gene functions. TIGRFAM [12] is another functional scheme developed using hidden markov models for completed function annotations on genomes. Also, many functional hierarchies used for general gene function annotation was also developed such as MIPS [13], FunCat [7] and Gene Ontology [8] where Gene Ontology has become the widely used functional hierarchy at present.

2.3.1. Gene Ontology

Gene Ontology has become the de-facto standard in function annotation [9]. Gene Ontology has classes which describe gene functions. These classes have relationships between them such as: *is a, has part, part of, regulates, negatively regulates, positively regulates* [14]. The annotation terms are modeled as a Directed Acyclic Graph (DAG) in Gene Ontology.

The DAG structure hierarchy is similar to where each child annotation term inherits general/ parent annotations. There are three categories describing each gene product; Molecular Function (MF), Biological

Processes (BP) and Cellular Component (CC). One example from each category is shown in figure 3, figure 4, and figure 5. The descriptions of the three aspects are as follows:

1. Molecular function- Molecular activities of gene products that occur at the molecular level are described in molecular function level. (e.g- catalytic activity, binding activity)
2. Cellular component- The location where gene products are active. These terms describe a component of a cell that is a part of an anatomical structure (e.g- rough endoplasmic reticulum) or a gene product group (e.g- ribosome).
3. Biological process- Pathways and larger processes made up of the activities of multiple gene products.

According to Gene Ontology Consortium, the annotation is the modeling of biology. Annotations are statements or comments describing the functions of specific genes, using concepts in the Gene Ontology [14]. These GO annotations consist of evidence codes which indicate how the annotation to a particular term is supported and a reference that describes the work or analysis upon which the association between a specific GO term and the gene product is based. The evidence codes are fallen into three categories: (1) Experimental evidence codes, (2) Computational analysis evidence codes, and (3) Author statement evidence codes.

(1) Experimental evidence codes

The experimental evidence code in a GO annotation indicates the results from a physical characterization of a gene or gene product that has supported the association of a GO term [14]. Following are the types of experimental evidence codes:

- Inferred from Experiment (EXP)
- Inferred from Direct Assay (IDA)
- Inferred from Physical Interaction (IPI)

- Inferred from Mutant Phenotype (IMP)
- Inferred from Genetic Interaction (IGI)
- Inferred from Expression Pattern (IEP)

(2) Computational analysis evidence codes

The computational analysis evidence codes indicate that the annotation is based on an in-silico analysis of the gene sequence. Following are the types of computational analysis evidence codes:

- Inferred from Sequence or Structural Similarity (ISS)
- Inferred from Sequence Orthology (ISO)
- Inferred from Sequence Alignment (ISA)
- Inferred from Sequence Model (ISM)
- Inferred from Genomic Context (IGC)
- Inferred from Biological aspect of Ancestor (IBA)
- Inferred from Biological aspect of Descendant (IBD)
- Inferred from Key Residues (IKR)
- Inferred from Rapid Divergence(IRD)
- Inferred from Reviewed Computational Analysis (RCA)

(3) Author statement evidence codes

In author statement evidence code, the annotation was made on the basis of a statement made by the author(s) in the reference cited. There are two types of author statement evidence codes namely:

- Traceable Author Statement (TAS)
- Non-traceable Author Statement (NAS)

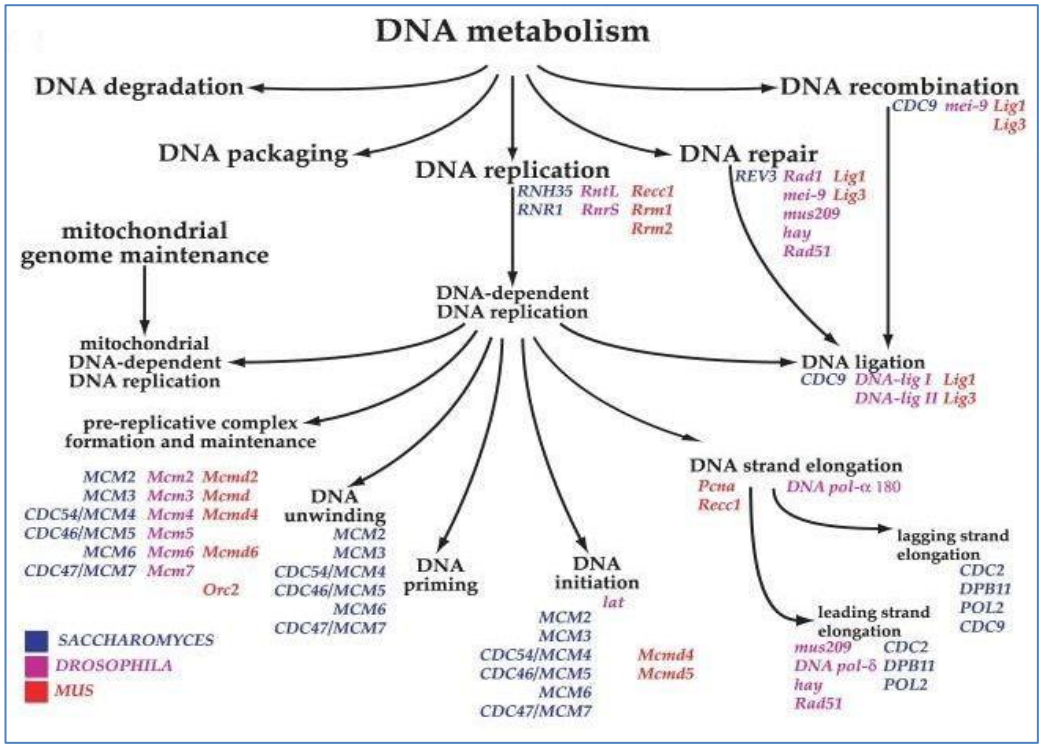

Figure 4: Biological Process- DNA Metabolism hierarchy breakdown

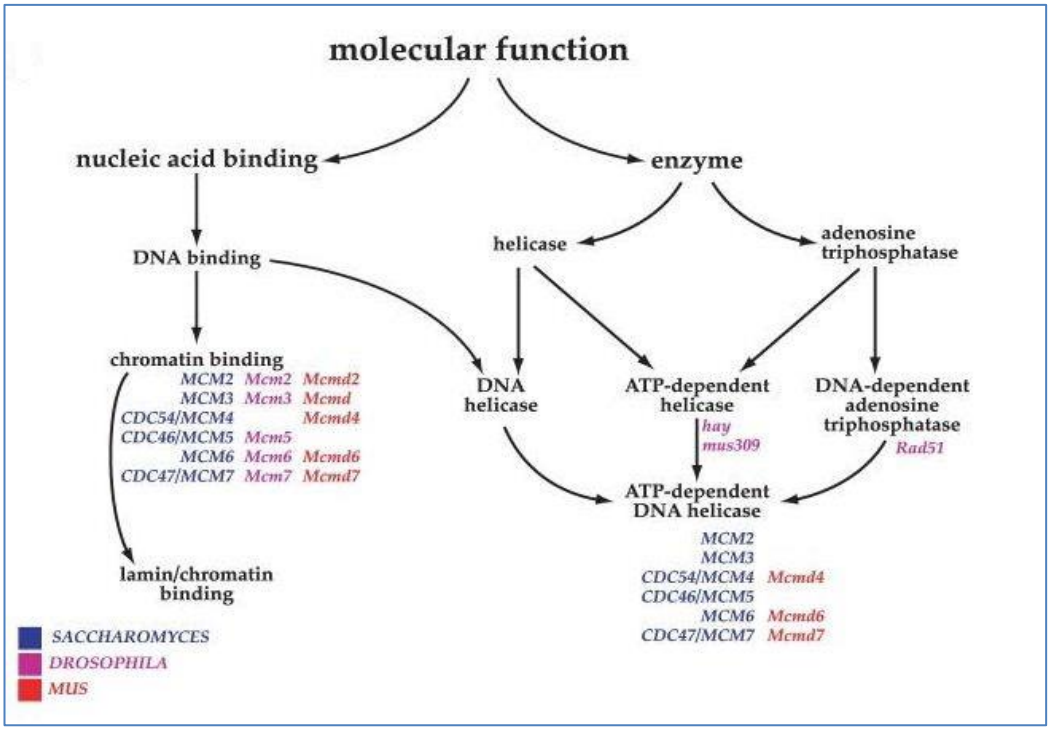

Figure 5: Molecular Function breakdown taken from (Ashburner et al. 2000)

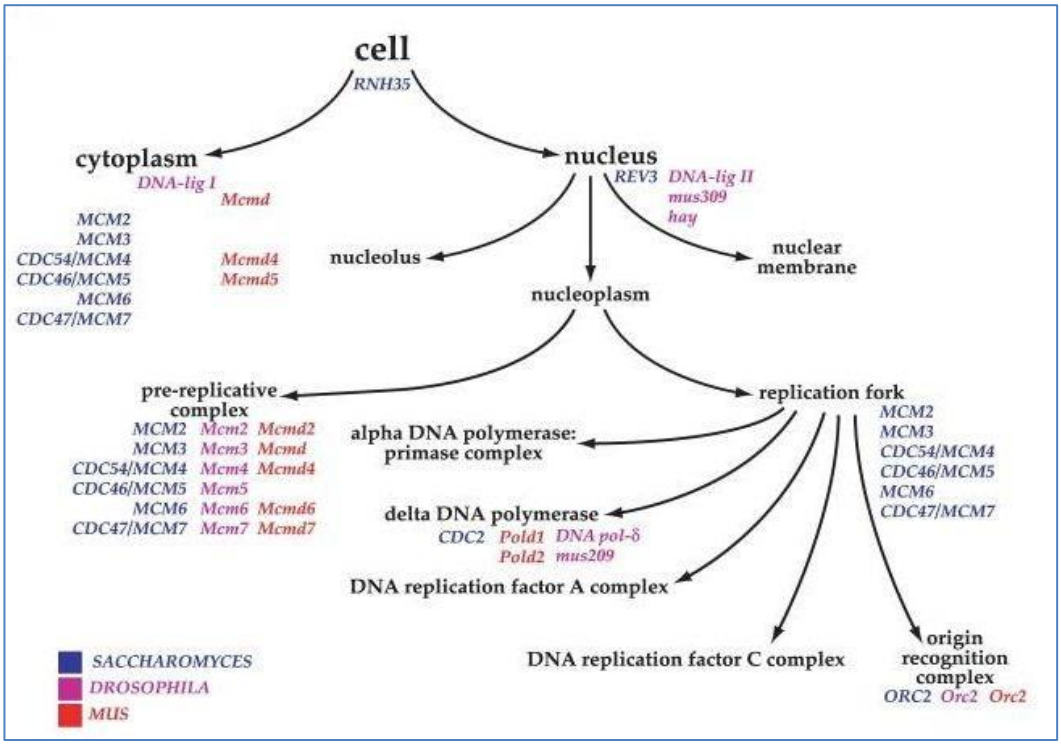

Figure 6: Cellular Component: Cell breakdown taken from (Ashburner et al. 2000)

2.3.2. FunCat (Functional Catalogue)

The functional catalog is an organism independent functional classification scheme which enables the manual and automatic annotation of gene functions. The main functional categories (version 2.0) of FunCat is spread over 7 categories as shown in table 1. They are metabolism, information pathways, transport, perception and response to stimuli, developmental process, localization, and experimentally uncategorized proteins [7].

Table 1: Main functional categories of the FunCat- version 2.0 taken from (a. ruepp et al. 2004)

Category ID	Category Name
Metabolism	
01	Metabolism
02	Energy
04	Storage Protein
Information Pathways	
10	Cell cycle and DNA processing
11	Transcription

12	Protein synthesis
14	Protein fate (folding, modification and destination)
16	Protein with binding function or cofactor requirement (structural or catalytic)
18	Protein activity regulation
Transport	
20	Cellular transport, transport facilitation and transport routes
Perception and response to stimuli	
30	Cellular communication/signal transduction mechanism
32	Cell rescue, defense and virulence
34	Interaction with the cellular environment
36	Interaction with the environment (systemic)
38	Transposable elements, viral and plasmid proteins
Developmental processes	
40	Cell fate
41	Development (systemic)
42	Biogenesis of cellular components
43	Cell type differentiation
45	Tissue differentiation
47	Organ differentiation
Localization	
70	Subcellular localization
73	Cell type localization
75	Tissue localization
77	Organ localization
78	Ubiquitous expression
Experimentally uncharacterized proteins	
98	Classification not yet clear-cut
99	Unclassified proteins

2.4. Importance of gene function prediction

In-silico gene function prediction helps biologists to identify the risks of developing a particular disease, track the inheritance of disease genes within families, locate genes that are associated with disease, the reaction of cells on drugs and drug discovery in an efficient manner over time-consuming, extensive and expensive laboratory experiments. The analysis, comparison, and predictions become easier with computational gene function prediction but the challenge is the accuracy level of the predictions made. A key problem of the pharmaceutical industry is to understand variations in drug treatment responses among individuals at the molecular level where gene function prediction plays a major role to identify new drugs accordingly and develop reliable diagnostics. The Human Genome Project is an international research effort to determine the nucleotide sequence in the human genome which identified the genes in a human body. This project has helped the researchers to understand the blueprint of a human. There are several input data which are helpful in identifying risks of health by function prediction such as gene sequence information, protein structure information and even single nucleotide polymorphisms (SNPs). Also, single nucleotide polymorphisms have gained much attention because they usually associate with diseases and these are helpful in identifying and tracking the inheritance of diseases. To make a new cell, an existing cell divided into two. It first copies the existing DNA so the new cells also have the same set of genetic instructions. During this process of copying, cells sometimes make mistakes. As an example, the nucleotide cytosine (C) gets replaced with the nucleotide thymine (T) in a certain stretch of DNA. This leads to a variation in the DNA sequence at a particular location, called single nucleotide polymorphisms or SNPs which create biological variation between people. Therefore, many research projects were begun to identify SNPs that will affect human health. When SNPs occur within a gene or in a regulatory region near a gene, they may play a vital role in diseases by affecting the gene's function [15]. Many machine learning algorithms are currently in use to predict functions of genes based on sequence data, protein structure data, and SNPs.

In this paper, we have proposed an optimized classification method called K- Nearest Neighbor to do the function prediction. It is an unsupervised machine learning algorithm. These algorithms take labeled data and create a model prior to the prediction. After that, the algorithms will use the generated model during the training phase to predict functions of test data. Accuracies are calculated by several validation methods. There are many research activities discussed more on prediction accuracy which is one aspect we are going to improve by optimizing the general K- Nearest Neighbor Algorithm. Based on various cross-validation methods such as homogeneous, heterogeneous and mixed methods tested with support vector machines and decision trees on data sets and also by analyzing the accuracy on predictions, researchers have identified more accurate functional class prediction [15].

Current research on gene function prediction, aimed at maximizing predictive accuracy but not in validation and interpretation of discovered knowledge[16]. Most of them are "black-box" methods which provide very less biologically meaningful explanation and give little new insight into data and application domain to biologists. Therefore, our idea extended into providing a white-box approach to gene function prediction using GAKNN where each knowledge discovery step is interpreted clearly with user-friendly interfaces.

2.5. Methods to predict the functions of genes

Labeled sequence data is available in many public databases such as NCBI, SWISS-PROT, PROSITE, MIPS etc. These databases store sequence data, microarray data, function data mostly in the form of Gene Ontology terms. Research on gene function prediction uses many bioinformatic approaches such as sequence-based approach, structure-based approach, and machine learning approach. Sequence-based methods for gene prediction has been dominating the field for several years with algorithms such as BLAST[17], PSI_BLAST and FASTA[18]. There are several approaches to sequence-based methods. Methods based on global and local sequence

alignments[19] where the method will look for similar sequences in different organisms. For an example, a user with an unknown mice gene sequence can perform a BLAST search in the human genome in order to know whether humans carry a similar sequence of mice. In local alignment, the query sequence will be test against a subpart of the reference whereas in global alignment the query sequence will be tested against the whole reference. Another method based on sequences is sequence motifs [20] [21]. Sequence motifs are short, recurring patterns in DNA that are likely to have biological functions. Sequence motifs correspond to functional regions of protein-catalytic sites, binding sites, structural motifs. The presence of these motifs often reveals important clues to a protein's role even if it is not globally similar to any known protein [22]. The most popular approach between the two strategies mentioned above is the use of alignment–based sequence similarity search as BLAST [1] or FASTA [18] to find similar proteins in public databases. The methods like PSI-BLAST [23] which are based on sequence profiles are developed to provide high sensitivity for detecting remote homologs.The sequence similarity-based approaches may not be always adequate for function identification of novel proteins because of the genes that have low or no sequence similarity where functional classes not known.

As sequence alignment methods suffer from insufficient homologous sequence data, computational methods utilizing three-dimensional structures are employed to overcome the issues in sequence alignment. There are several approaches to structure-based function prediction. The methods that utilize fold information depend on global and local structural alignments algorithms [24]. There are many methods of structural alignment such as: DALI(Distance alignment matrix method), Combinatorial extension, and SSAP(Sequential Structure Alignment Program). In DALI, the input structures are breaks into hexapeptide fragments and calculate the distance by evaluating the patterns between successive fragments. The combinatorial extension is also quite the same as DALI where the method breaks each query structure into a series of fragments and the reassemble for alignment. A series

of pairwise combinations of fragments are taken as aligned fragment pairs are used to define a similarity matrix where an optimal path is generated to identify the final alignment. The sequential structure alignment method is using the atom-to-atom vectors where double dynamic programming is used to determine a series of optimal local alignments first and then summed into a summary matrix to determine the overall structure. Although sequence and structure-based methods dominated the field of gene prediction, they suffer from limitations of availability of adequate data of homologous proteins and fail when homology relationships cannot be established for target proteins [25]. On the other hand, the structure-based approaches are again limited to structures and folds in databases. Because of these challenges, machine learning approaches were introduced to predict various functional aspects of genes. These machine learning based approaches utilize sequence or protein structure data to build classification models and do prediction without homology information [1]. Furthermore, machine learning algorithms use existing data and build models and classify or predict novel data. Supervised and Unsupervised learning are two approaches to machine learning where supervised learning has its input data called as training data with a known label. A model is built through the training process where that model is required to make predictions. In unsupervised learning, there are no labeled data and a model is prepared by deducing structures present in input data. These machine learning algorithms have become very important in many bioinformatics prediction methods such as gene prediction, localization prediction, and protein function prediction. There are several algorithms in machine learning which can be applied in function prediction such as Artificial Neural Networks, Hidden-Markov Models, K-Nearest Neighbor, Decision Trees, and Support Vector Machines.

Artificial Neural Networks are algorithms based on the concept of neurons in human brain. Typically, a neural network consists of 3 layers. Input layer, hidden layer, and an output layer. These neural networks train a hidden-layer-containing a network and use its connected structures for pattern recognition and classification [26]. Advantages of such artificial neural

networks are the ability to process and analyze large complex datasets, containing non-linear relationships and the ability to handle noisy data. One disadvantage is time taken in processing complex datasets [27]. Some applications of neural networks in the field of bioinformatics are protein secondary structure prediction [28] and sequence feature analysis. The downsides of neural networks are that it requires a lot of data and when the dataset gets complex, neural networks also get difficult to interpret[29]. The black box approach is a disadvantage when implementing a software to analyze and determine the functions of genes as we intend to develop a solution where biologists can easily understand and describe the final outcome of prediction using gene annotation data.

Not only neural networks but also the Hidden Markov model is a very popular machine learning approach in bioinformatics. These are probabilistic models which are generally applicable to time series or linear sequences. These models are easy to use, need smaller datasets but require a lot of computational cost. Also, the hidden markov model is unable to express the dependencies between the hidden layers [30]. Hidden Markov Models have been used in protein secondary structure and function prediction in recent years [31] [32].

On the other hand, Support Vector Machines provide the best generalization ability with less over-fitting of data and robustness to noisy data. This is the most commonly used machine learning algorithm in computational biology. Support Vector Machines have been used for site prediction, fold recognition and gene finding [33].

Another machine learning algorithm with better interpretability is Decision Trees. This is a simple and easy to learn classifier which constructs decision tress by analyzing a set of training examples for which the class labels are known. This information is then used to classify naïve examples. In reference [34], they presented the use of a decision tree-based ensemble learner called CLUS-HMC-ENS for multi-label functional prediction. CLUS-

HMC outperforms the decision tree learner (C4.5H/M) predictive performance. This extends to hierarchical multi-label classification context where the gene function prediction task is set. Another approach in multi-label classification is to compare three decision tree algorithms on the task of hierarchical multi-label classification [35]. First, a single tree will predict all the classes. Then, a separate decision tree for each class will be generated. Finally, an algorithm learns and applies single-label decision trees in a hierarchical way.

Decision trees are widely used in prediction [36] [37] although they are sensitive to small variations in data and error-prone in case of data with a large number of classes where K-Nearest Neighbor classifier comes with the greater ability to perform efficiently with large training data. KNN is a simple algorithm which stores available instances and classifies new instances based on a similarity measure. KNN is also widely used in functional genomics [38] occupying its ability in high efficiency on larger datasets and robustness with noisy data.

As k-Nearest Neighbor Classification gained much attention because of high efficiency on larger data sets and robustness when processing noisy data, it is widely used now in prediction. There are many applications which use gene sequencing data of animals, plants, and fungi to predict the protein localization sites [39]. As a for heterogeneity of gene functions data, a Multi-Source K-Nearest Neighbor algorithm has been proposed by integrating the KNN algorithm with multiple data sources such as; (1) protein sequence data; (2) microarray expression data; (3) protein-protein interaction data[40]. Also by using gene annotation data exist in comprehensive genome databases such as, phenotype, localization, protein class, complexes, enzyme catalogs, pathway information, and protein-protein interactions, the functional classes are predicted for proteins [4]. In GAKNN framework, a white-box approach has been followed where data interpretation can be viewed clearly by the user. Like in many machine learning algorithms, k-NN requires a high computational cost for larger data sets and also k-NN shows inconsistent

performance when the number of attributes increases. Therefore, k-NN needs to be combined with an optimization process. The genetic algorithm is used to serve this purpose of optimization and reduce the search space. We have used a Genetic Algorithm optimized k-Nearest Neighbor Classification technique to predict the functional classes of genes given the gene annotation data, gene expression data, and gene ontology data. Also, the GAKNN framework supports all types of data whether that is numeric data, nominal data or binary data.

2.6. Genetic Algorithm

Genetic algorithms are widely used for optimization needs. A genetic algorithm searches for the optimal solution. Figure 7 illustrates the steps of the genetic algorithm. First, it creates a population of strings. These strings are named as chromosomes. In general, these chromosomes are represented by a bit string (a binary string with 1 s and 0s). The population is a collection of candidate s for a defined problem. A single solution in the population is referred to as an individual. A fitness score which represents a solution is calculated for each individual to measure how "good" the individual is. The highest the fitness value is, better the solution. Two individuals which are more fit are selected out of this population. This selection process is based on the concept of "Survival of the fittest" in the natural world. This fitness function is used to measure the quality of an individual in order to increase its probability of survival throughout the evolutionary process. After that, these individuals are selected for "breeding" where they reproduce another two new individuals (offspring). This is done by the crossover operator in genetic algorithms. Crossover creates two offspring strings which are copied from the parent strings with highest fitness value. During each new generation of individuals, there is a chance for each individual to mutate. There is a random chance for individuals to get a change in a small way than their parents by changing the value of a single bit in the string. This process will continue until the algorithm meets its termination conditions. Each iteration of selecting the fittest individuals, cross-over, and

mutation is called a generation. There is no one way in the termination condition mentioned above because there are many ways to end the algorithm. One way is to run the search for a number of generations. The longer the algorithm runs the better. Another approach is to end the algorithm after a certain number of generations pass with no improvement in the fitness of the best individual in the population [41]. The simplest genetic algorithm uses fitness-proportionate selection, single-point crossover, and single-point mutation.

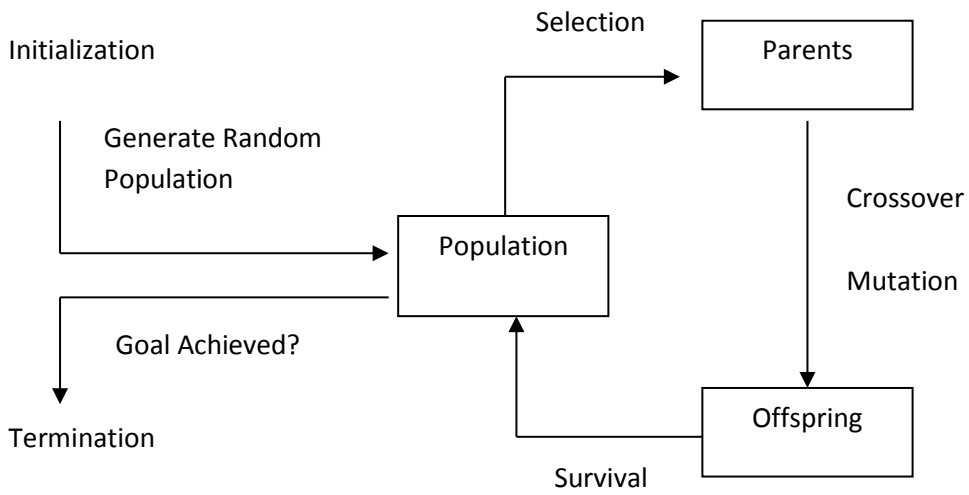

Figure 7: Illustration of how genetic algorithm works

In this paper, we use the genetic algorithm to optimize the K- Nearest Neighbor Algorithm to find out the best number of neighbors to consider when predicting unlabeled data and also to assign weights to each attribute in the data set. There are so many applications under genetic algorithms optimizing K-Nearest Neighbor Algorithm. Genetic algorithms have been widely used for many optimization problems. Genetic algorithms are used for feature selection and optimization of many algorithms[42], [43]. Also, the genetic algorithms are commonly used with k- nearest neighbor algorithm as k-NN is mainly used with large datasets. The k-NN algorithm with genetic algorithms can be used to get weighted vectors for attributes in a dataset [44], [45]. The genetic algorithm assigns weights to each attribute and finds weight vector for attributes. Also, the

genetic algorithm is used to find the optimum value for the parameter k which indicates the number of neighbors to look up in a majority vote.

2.7. Machine Learning Algorithms

Machine learning algorithms are mainly divided into three categories: Supervised Learning Algorithms, Semi-Supervised Learning Algorithms, and Unsupervised Learning Algorithms.

2.7.1. Supervised Learning Algorithms

A supervised learning algorithm is where there is an input (x) and output (y).

$$y = f(x)$$

In supervised learning, a learning algorithm is used to learn the mapping function from the input to output [46]. Once the mapping function is learned, the algorithm can take new input data and predict the output. Similarly, in data mining, the input is the training data. The mapping function is creating a model out of input data. In order to supervise the algorithm to build a model, a training dataset is essential. A training dataset has all classes labeled in it. The classes are the predictive values in a dataset. The model is built through the training process in which it is required to make predictions and is corrected when those predictions are wrong. This training process continues until the model achieves the desired level of accuracy on the training data. Finally, a test dataset or an actual dataset which does not have the classes labeled in it can be the input once the model is built by the algorithm.

These supervised learning algorithms are further can be grouped as Classification and Regression. Classification is when the prediction falls into a category such as “blue”, “affected”, “not affected”, “pass” or “fail”. Regression is when the prediction is a real value such as: “weight”, “height”, “distance”.

2.7.2. Unsupervised Algorithms

In unsupervised learning, there is no corresponding output for the input(x). The algorithm has to model the structure or distribution of data in order to learn about data [41].

There is no training associated with unsupervised algorithms as algorithm will learn from the data structure. Unsupervised learning further divided into two categories: clustering and association.

Clustering is grouping similar type of data into groups such as similar type of flower species, purchasing behaviors.

Association is rules that imply relationships among data such as people who buy X also buys Y.

2.7.3. Semi-Supervised Learning Algorithms

Semi-supervised is some input data are consist of class labels and some are not. Usually, the majority of data are unlabeled and the data with labeled classes can be used as training data to create models. Also, the unsupervised learning techniques can be applied to unlabeled data and do the predictions. Most of the real world machine-learning problems are fallen into this category.

As we are using a classification technique to predict gene functions, the k-Nearest Neighbor Classification Algorithm adheres a supervised learning technique where the need for a training data comes first.

2.7.4. K- Nearest Neighbor algorithm

The k- nearest neighbor algorithm is a supervised learning algorithm. It is an instance based learner where a training data set is required. The parameter k implies the number of nearest neighbors to look upon when predicting a value. When an unknown class is given, the algorithm will calculate the distance between the unknown instance and other instances where the class label is

known. From that, the algorithm identifies the closest instances which give the lowest values for the distance measurement. If k is 3, the algorithm will look upon the 3 nearest neighbors and predict the unknown class label by the majority vote out of the three neighbors. Figure 8 shows an example of finding the closest neighbors when k= 3.

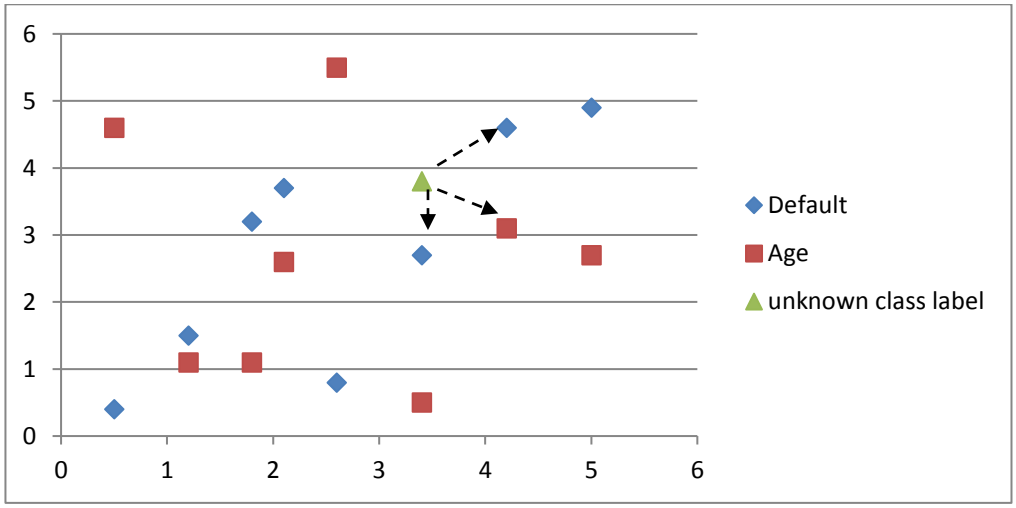

Figure 8: k- Nearest Neighbor Algorithm distance measurement when k= 3

As shown in the above example k- NN classifier will take the nearest neighbors and take the majority vote. In here, by majority vote, the unknown class label belongs to "Default" class. Despite the simplicity of the k- NN classifier the advantages of k- NN classifier in gene function prediction is the efficiency shown by k- NN for larger datasets and the robustness when processing noisy data. K- NN also shows better results where the class boundaries are inherently vague, and many classes cannot be categorized by a simple model [47].

CHAPTER 3

3. Methodology

The k- Nearest Neighbor Algorithm and the Genetic Algorithm are the core of the prediction approach used in gene function prediction of GAKNN using gene annotation data. This chapter describes the methodology used in GAKNN in detail.

The disadvantages of standard kNN are the high computational cost while calculating distance measurements and the inconsistence performance when the number of attributes increased. In order to overcome the disadvantages of standard k-NN; we have used the genetic algorithm optimization for k- NN to make the classification method more efficient and accurate in results.

The optimization will reduce the search space for a certain dataset by assigning weights to most prominent attributes towards the prediction. Also, the genetic algorithm will find the optimum k (the optimum number of nearest neighbors) for a certain application. The selection of a similarity metric to identify the neighbors and the selection of the optimum k can be considered as the optimization problem that we use the genetic algorithm for.

3.1. Genetic Algorithm Optimized k- Nearest Neighbor Algorithm

In the context of data mining, classification is done using a model that is built on historical data. The goal of predictive classification is to accurately predict the target class for each record in new data which could not in the historical data. A classification algorithm like k-NN needs training data for which the target values (or class assignments) are known. A training dataset is used to find relations between the predictor attributes' values and the target attribute's values in the trained data. These relations are summarized in a model and the model can then be applied to new cases with unknown target values to predict target values. Then the built model can be applied to data that was held aside from the training data to compare the predictions to the known target values; such data is also known as test data or evaluation data. By testing, we can get the model's predictive accuracy [48].

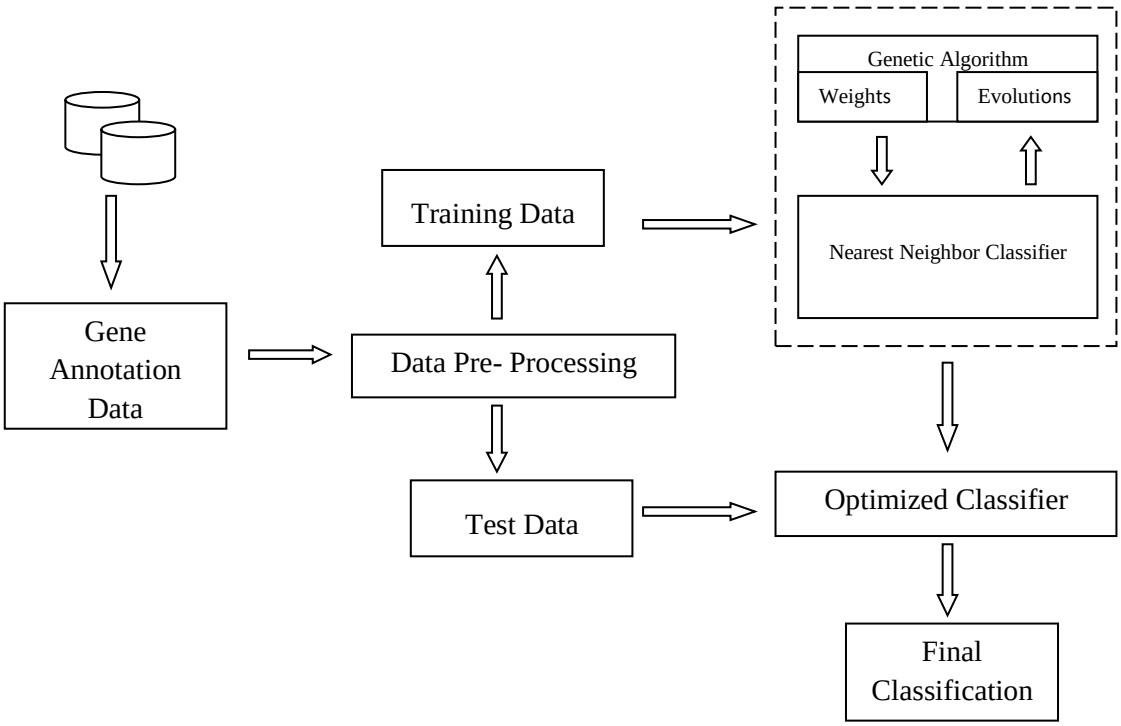

Figure 9: System architecture of GAKNN

Gene annotation data related to gene functions are downloaded from repositories and put into data pre-processing. After the data pre-processing, the dataset can be divided into two sets where one set is used for training. The training dataset is passed to the genetic algorithm optimized k- nearest neighbor algorithm where the algorithm assigns weights to each attribute. Based on the importance of the attribute in predicting the unknown class, the genetic algorithm assigns weights to each attribute. For an example, when identifying a person as an Asian or not based on a dataset with features associated with height, weight, hair color, eye color, the features such as hair color and eye color plays a vital role in predicting a person as an Asian rather than the features like height and weight. This learning and assignment of weights are done by the genetic algorithm resulting a trained model with attribute weights and optimum k to be used with the test dataset while predicting unknown classes.

As the genetic algorithm runs several iterations/ evolutions, a set of weights for all attributes will be assigned by the algorithm at each iteration. A fitness score will also

be calculated for each iteration using the genetic algorithm and that is to indicate how good the solution is. Higher the fitness score, better the solution. The fitness function is defined for this application is as below:

$$Fit = \left[1/\sqrt{\sum_{0}^{m-1} (1 - Conf)^2} \right] \text{x } 100$$

The above fitness score is computed assuming an appropriate classification confidence measure for the respective application. Usually, a genetic algorithm runs until it meets a threshold or until there is no improvement in the fitness score [44] [49]. In this application, we have used the latter where several iterations of the genetic algorithm run until there is no improvement in the fitness score.

The weights set with the highest fitness score is saved as the model to be used during the imputation process where missing values get estimated and in the prediction process. The selection of a similarity metric to identify the neighbors and the selection of the optimum k can be considered as the optimization problem where we use the genetic algorithm for.

The similarity functions are defined as follows:

Nominal variables, di = $\begin{cases} W_i & X_i = Y_i \\ 0 & X_i \neq Y_i \end{cases}$

Numeric attributes, $d_i = W_i (X_i - Y_i)$

Similarity, Sim = $\frac{1}{\sum_{i=0}^{n-1} d_i}$

The test dataset with unknown classes is given to the optimized classifier with the trained model. The results will be predicted for a given test dataset using the optimized classifier. We have developed GAKNN as a standalone application with a graphical user interface to be more approachable and user-friendly for the use of bio-informaticians and biologists.

3.2. GAKNN as a software

GAKNN is developed using Java while the genetic algorithm is built upon JGAP (Java Genetic Algorithms Package). As shown in Figure 9, the high-level architecture of this software can be illustrated using 4 main steps:

i) Data selection and pre-processing

ii) Training a dataset which predictive class labels are known.

iii) Save the model and the use of genetic algorithm optimization

iv) Input test data to the optimized classifier

v) The output of prediction results based on the trained model from step ii).

GAKNN follows the steps of knowledge discovery iteratively until the step of data mining. As GAKNN follows the steps of knowledge discovery process, each step is elaborated below:

3.2.1. Knowledge Discovery process

Knowledge discovery process can be defined as extracting knowledge from data. The data mining process is also known as knowledge discovery process while some people think of data mining as an essential step in the process of knowledge discovery [50].

3.2.1.1. Data Cleaning

Data cleaning is the first step of knowledge discovery process. Before each dataset goes through the knowledge discovery process, noise and inconsistent data should be removed otherwise the final results will create bias and give a wrong interpretation of data.

3.2.1.2. Data Integration

In this step, data from multiple sources are collected in one place. The process of combining data from various sources is called data integration.

3.2.1.3. Data Selection

Data relevant to the analysis task are retrieved from the database in this step. Feature selection, where relevant features/ columns are selected is also happened in this phase.

3.2.1.4. Data Mining

Data mining is to use machine learning methods in order to identify patterns in the input data. The extracted data patterns are then sent for evaluation.

3.2.1.5. Pattern Evaluation

The identified patterns from data mining are taken for analysis in this step. By pattern evaluation, the truly interesting patterns which represent knowledge will be selected on interestingness measurements. That is, not all identified patterns are useful and interesting. A pattern is interesting if it is i) easily understood by the user ii) valid on test data with some degree of certainty iii) potentially useful iv) and novel.

3.2.1.6. Knowledge Representation

The final step of knowledge discovery process is knowledge representation where data visualization techniques are used to interpret knowledge to the user. Steps from data cleaning to data mining are involved with data pre-processing where data is initially prepared for mining. GAKNN uses the steps up to data mining where gene functions are predicted based on a supervised learning technique.

3.2.2. Data Pre- Processing

Data pre-processing is a vital part of data mining as data found in real life can always be with missing values, noise, and redundant data. Dirty data always result in unreliable output. GAKNN is compatible with .csv, .txt and .arff data file formats. The GAKNN provides a workspace as shown in figure 10 for data pre-processing as gene annotation data/ gene expression data can be often incomplete, noisy and inconsistent. The reasons for noisy data are having faulty instruments when collecting data, human or computer errors happen during data entry, errors occur in data transmission. Incorrect data usually occur from inconsistencies in naming conventions or data codes used, or inconsistent formats for input fields, such as *date*. Redundant data columns or rows should be deleted before using a dataset for data mining. There are various techniques for data

cleaning. The process of data cleaning associates with filling in missing values, smoothing noisy data, identifying or removing outliers, and resolving inconsistencies. When data comes from multiple sources, data integration can be used to combine all data from different sources into one file and make a common format. GAKNN provides several pre-processing techniques. Feature selection is one technique available to delete redundant, noisy or inconsistent data. Figure 11 shows a screenshot taken when a user does feature selection in the GAKNN workspace. GAKNN handles incomplete data by missing data imputation as shown in figure 12. Our research work on building a gene function prediction tool is also propose kNNImputation algorithm which is used for estimating the missing values as gene annotation data/ gene expression data highly consist of missing values.

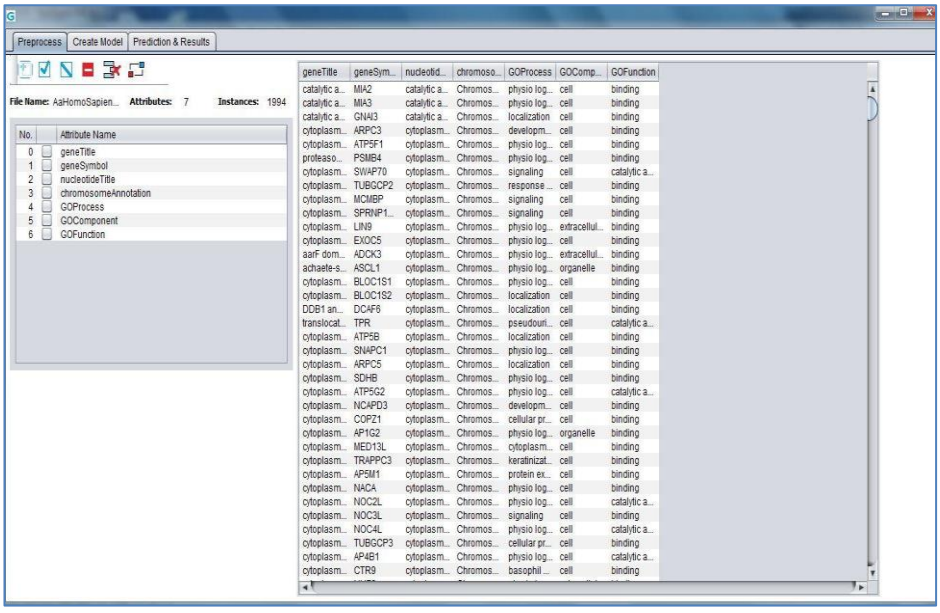

Figure 10: GAKNN workspace for data pre-processing

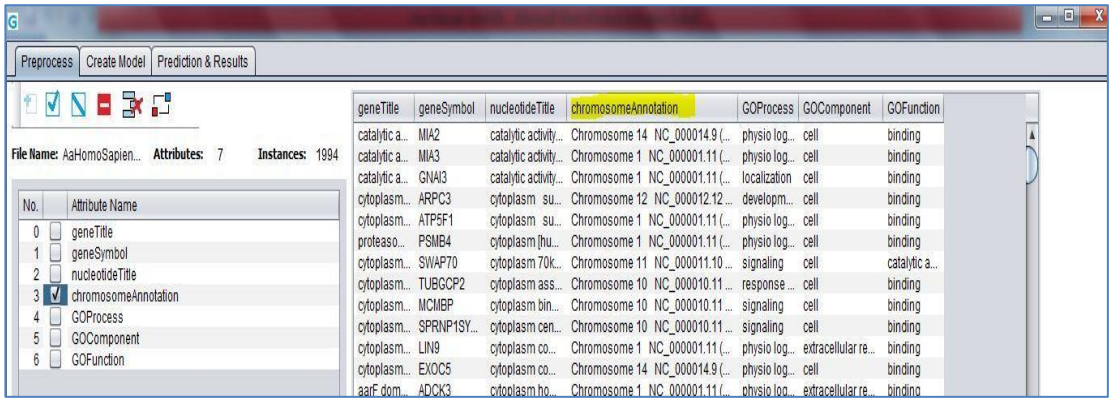

Figure 11: Feature selection in GAKNN by user

3.2.2.1. Missing Data Imputation

GAKNN's missing data imputation only supports for numerical data. The workspace provided for pre-processing has the option to impute numerical data in training data and save the changes for further optimization.

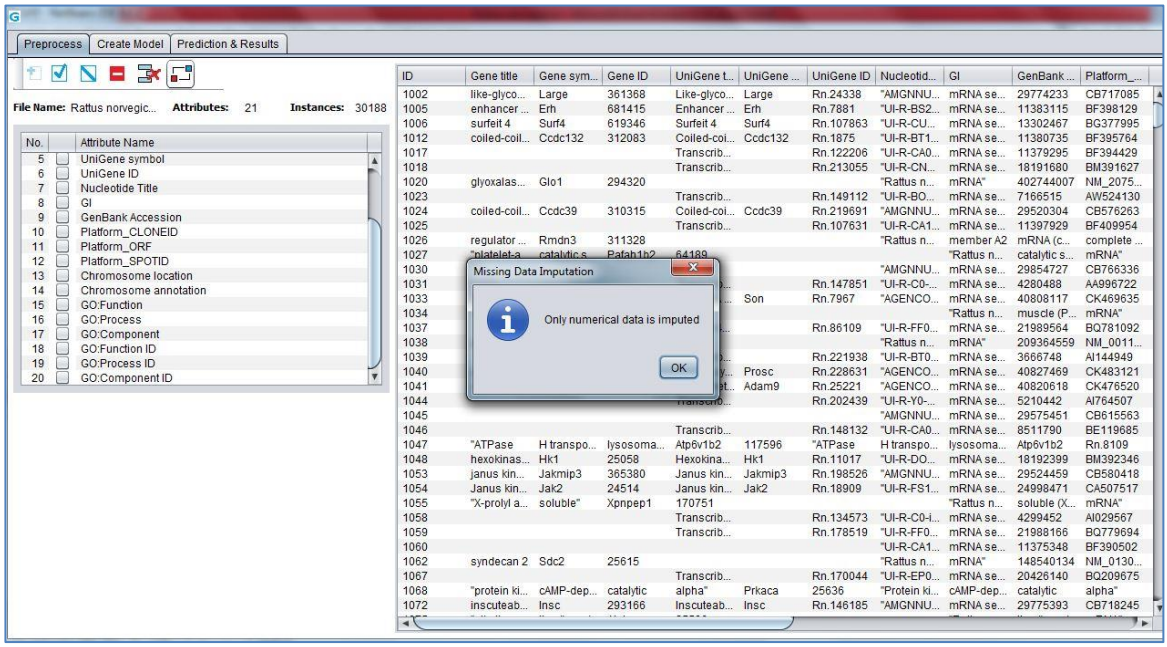

Figure 12: Missing data imputation of GAKNN

The DNA microarray technology is used to monitor expression data under a variety of conditions [51]. Scientists are using microarray technology to study

biological processes of gene expression data in human tumors to yeast sporulation [52], [53]. Also, with the microarray technology to generate gene expression data some spots on the array may be missing due to various factors (for example, machine error) [54].

Generally, most methods handle missing data simply by discarding the missing data. The discard of missing data can lead to estimates with larger standard errors due to reduced sample size [55]. Dropping such cases with missing data has yield biased or inconclusive results even though such techniques are still widely used in software engineering [56]. This method is known as "complete case analysis". Another method of ignoring missing data from datasets is the "available case analysis" where different subsets of data are taken to different aspects of the same study due to inability in taking the full dataset because some values of variables have incomplete data. This approach excludes some variables which are needed to satisfy the assumptions necessary for desired interpretations [55]. Complete case analysis and available case analysis both are reducing the sample sizes of the datasets. Missing value imputation of numerical data is mostly handled in general by mean substitution in several works [56]. The main disadvantage is that this method can distort the distribution for the variable which is used for imputation by underestimating the standard deviation [55]. Median imputation is also used to assure robustness since mean is affected by outliers of a dataset. For the categorical attributes, the mode imputation is used instead of mean and median of a dataset [57].

The disadvantage of this method is that it does not consider dependencies among attribute values [58]. Another widely used method is multiple imputations which missing values are predicted using existing values from other variables. This process is performed multiple times, producing multiple imputed data sets[59].

Missing value estimation approaches can be categorized into four main categories such as Global approach, Local approach, Hybrid approach, and Knowledge-based approach.

3.2.2.1.1. Global Approach

In global approach, the algorithm does the missing value estimation by looking into the entire data matrix with global correlation information. According to [58] if the algorithms assume that there exists a global covariance structure in all genes samples and that the genes exhibit dominant local similarity structures, then the imputation will become less accurate. Examples of algorithms which use the global approach are SVDImputation [59] and Bayesian Principal Component Analysis (BPCA) [60].

3.2.2.1.2. Local Approach

In local approach, algorithms take the local similarity structures of the data matrix in order to do the missing value estimation. The subsets of genes that show high correlation with the genes that contain the missing values are used to compute the missing values. The KNN imputation (KNNimpute) and local least square imputation (LLSimpute) are some of the common and efficient algorithms for the local approach. The KNNImpute [59] takes the pairwise information between the target gene with missing values and the reference genes to do the imputation of missing data. The LLSImpute [61] uses a multiple regression model to impute missing values. Sequential LLSimpute (SLLSimpute) [62] is an extension of LLSimpute algorithm which performs imputation sequentially by starting from the gene with least missing rates. The imputed genes are then reused for imputation of other genes. It has been proven that SLLSimpute performs better than LLSimpute because the genes with missing values are reusable in this algorithm.

3.2.2.1.3. Hybrid Approach

Heterogeneous datasets require a local approach as the local correlation between genes are used to do the missing value estimation. There are some data sets which require global approaches because of the global correlation structure of data. There are some hybrid methods like LinCmb [63] that can capture both local and global correlation information in the data sets. Using this method, the missing values are estimated by a convex combination of five different imputation methods such as row average, KNNimpute, SVDimpute, BPCA, and GMCimpute. The LinCmb generates fake missing entries where the

true values are known and use the constituent methods to estimate the missing entries. LinCmb is also adaptive to the correlation structure of data matrix where more missing entries are present, global methods will become the focus to determine the missing values.

3.2.2.1.4. Knowledge Assisted Approach

This approach integrates the domain knowledge or external information into the missing values imputation process. This is a powerful approach as it significantly improves the accuracy of imputation using domain knowledge. Also, this approach performs well on datasets with a small number of samples which are noisy or have a high missing rate. An example of this approach is the Projection Onto Convex Set (POCS), which exploits the biological occurrence of synchronization loss and correlation information between genes. Histone Acetylation Information Aided Imputation (HAIimpute) is another example which combines histone acetylation information into KNNimpute and LLSimpute to improve the accuracy of missing value estimation [64].

Many machine learning algorithms solve missing data problem in an efficient way. One advantage of using a machine learning approach is that the missing data treatment is independent of the learning algorithm used [65]. Most common algorithms used in imputation are EM algorithm [66], SVDImpute [59], CN2 Induction Algorithm [67, p. 2], C4.5 Algorithm, the local least squares imputation method (LLS) [61], the Bayesian principal component analysis (BPCA) [60] and kNNImpute [65], [55], [68]. Commonly used imputation methods for gene expression data use clustering technique [69]-[70] and techniques based on supervised learning [71], [72]. Also, it has been found that some approaches are not compatible with missing data imputation because missing values are causing a negative effect on support vector machines (SVM), single value decomposition (SVD), and principal component analysis (PCA) as these methods cannot function on data with missing values [58]. As supervised learning algorithms show efficient ways of treating missing values, we have chosen kNNImpute for missing data imputation for following reasons: k-NN Imputation does not require to create a predictive model for each attribute with missing values in the dataset, treat

instances with multiple missing values, considers the correlation structure of data, and predict both qualitative and quantitative attributes. We have used three datasets to evaluate the efficiency of using a supervised algorithm like k-NN. Depending on the size of the datasets we have used several missing rates ranging 5% - 25% in the three datasets.

Table 2: Description of datasets used

Dataset	Description	Features	Instances	Time taken for optimization
gasch2	Microarray data	51	204	28.85 sec
spo	Microarray data, sporulation in budding yeast	76	1597	19.8 min
seq	Attributes calculated from sequence alone	14	500	44.12 sec

In gasch2 dataset[60] missing rates of 0.05%- 25% are used. The spo dataset [53] consisted of 0.02% - 0.16% of missing rates. The seq dataset[60] had missing rates of 0.06% - 0.6%. Figure 13, figure 14, and figure 15 illustrate the mean errors calculated for each imputation algorithm: meanImpute, kNNImpute, and EvlkNNImpute. The *k* value used in kNNImpute algorithm was 10.

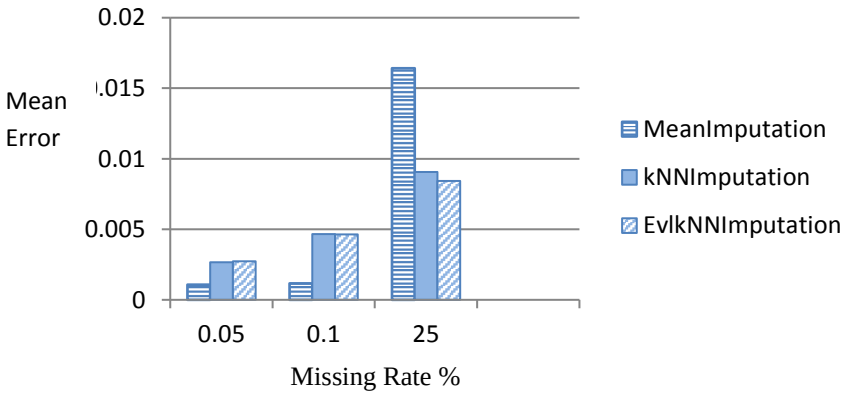

Figure 13: Mean errors of Mean Imputation, kNNImputation and EvlkNNImputation for gasch2 dataset

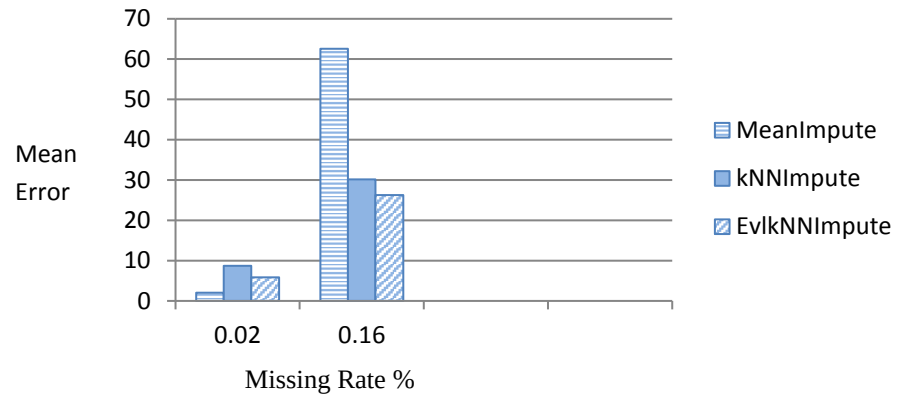

Figure 14: **Mean errors of Mean Imputation, kNNImputation, and EvlkNNImputation for spo dataset.**

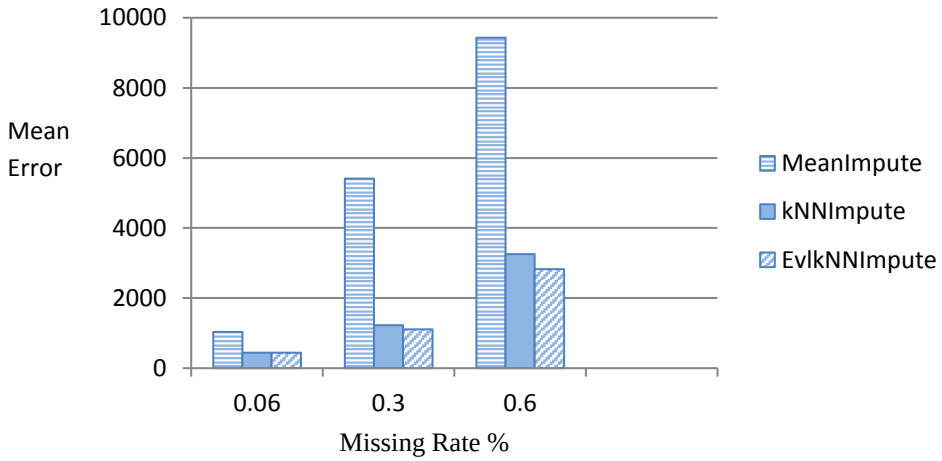

Figure 15: **Mean errors of MeanImputation, kNNImputation, and EvlkNNImputation for seq dataset.**

By looking at figure 13, figure 14, and figure 15 we can conclude that mean imputation is better when there are very few missing values in a dataset but often this is not the case when it comes to gene expression data as those datasets contain a considerable amount of missing data. Also, by mean imputation, the correlation of attributes in the data set is not taken into consideration by distorting the distribution and underestimating the standard deviation. Therefore, having a machine learning algorithm to estimate missing data values is more efficient and accurate hence GAKNN has kNNImpute for missing data imputation.

3.2.3. Data Transformation and Data Mining

3.2.3.1. GAKNN Optimization

GAKNN Optimization is done to find the optimum k for a dataset and mainly to find the weight vector given to attributes of the dataset. The weights assigned to each attribute make the machine learning algorithm aware which are the prominent attributes in the prediction of gene functional classes. Therefore, the optimization problem for the k-NN by the genetic algorithm is to find the vector of weights and optimum k.

After data pre-processing, the data is taken for training. First, the genetic algorithm will process the dataset and assign weights to each attribute. GAKNN provides another interface for data optimization as the next step in predicting the gene functional class. The user can set the number of iterations to run the genetic algorithm as the user is deciding which model is best to use in the process of prediction. By the time when certain iterations are passed with no improvement in the fitness score, that is an indication where the user has to stop running the optimization. Therefore, the user can keep on increasing the fitness score while there is no improvement in the fitness score and save the best model.

Figure 16 below shows the fitness scores for the *gasch2* dataset where the optimum k is 3.

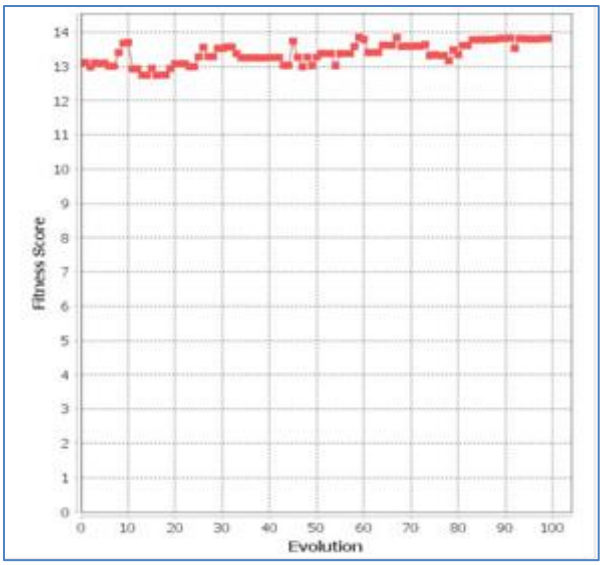

Figure 16: **Fitness score over iteration/ evolution for the gasch2 dataset**

Figure 17 below shows the fitness scores for the *seq* dataset when optimum k value is 10.

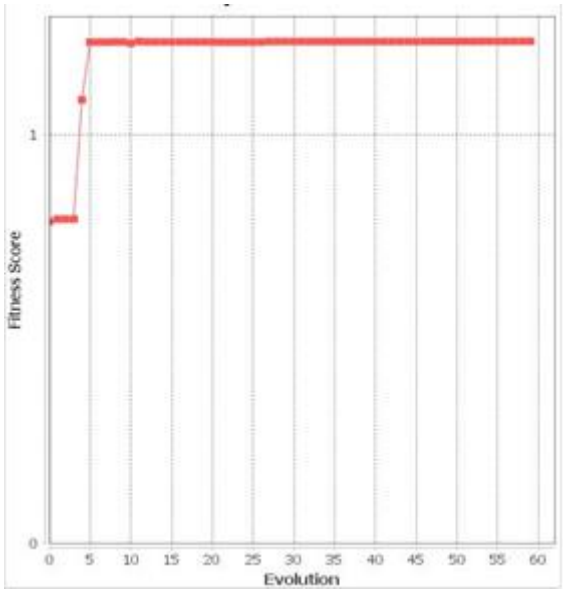

Figure 17: Fitness score over iteration/ evolution for the seq dataset

The weights assigned by the genetic algorithm also illustrated on a line chart by GAKNN. The fitness scores assigned by the genetic algorithm for each iteration/ evolution are also given using a line chart by GAKNN as shown in figure 15, figure 16, and figure 17.

3.2.3.2. Gene Functions Prediction

After the optimization process, the next step is to use the test dataset which we do not know the gene functional class labels. The algorithm will take the trained model and predict the gene functional class for each instance in the test dataset using the optimized k-NN classifier.

CHAPTER 4

4. Results

Our research study is on in-silico gene function prediction using gene annotation data available in repositories in order to overcome the difficulties and inefficiency of traditional curation process done in laboratories to find gene functions. Here, we have tested the solution we developed using two different datasets in two functional annotation schemas. The gene functions are mainly categorized into three aspects: biological_process, molecular_function, and cellular_component in Gene Ontology functional annotation scheme. The FUNCAT annotation scheme has more categories to classify functions which is shown in table 1 of chapter 2. The GAKNN was developed to be compatible with many functional annotation schemes when predicting the function classes. The core of the system is the genetic algorithm optimized kNN classification algorithm which was built upon JAVA and JGAP framework. The step by step methodology explained in the previous chapter with the knowledge discovery process is expected to give a white-box approach towards the prediction of gene functions as the solution we developed is mainly used by biologists. GAKNN can take any dataset from any functional annotation schemas which are saved in .txt, .arff, and .csv file formats.

Data are downloaded from various sources for testing purposes. We have chosen one dataset from Aberystwyth University Bioinformatics and Computational Biology Unit which belongs to FUNCAT hierarchy. Access to datasets: http://www.aber.ac.uk/en/cs/research/cb/dss/yeastdata/. Also, another Gene Ontology dataset was taken from National Center for Biotechnology Information (NCBI). NCBI is a repository storing bioinformatics and biomedical data. The data set from NCBI was categorized under “GEO Datasets” in NCBI page. GEO- Gene Expression Omnibus is a database in NCBI which stores curated gene expression data sets. Access to GEO datasets can be found at: https://www.ncbi.nlm.nih.gov/gds.

The gene expression datasets are multi-class classification domains. The GAKNN provides 10-fold cross validation for accuracy testing. In order to provide an extensive evaluation of datasets, we have used two functional schemes: Gene Ontology and FunCat from two different sources.

Table 3 below is a binary classification done by us for a FunCat dataset before moving towards to multi-class classification for the purpose of providing an extensive analysis. As an example, the data instances with the FunCat function 2_0_0_0 "Energy" were labeled with 1s and other functions were labeled with 0s. All functions labeled in the dataset were taken for binary classification and GAKNN reported high accuracy levels in each function. We were able to achieve more than 80% accuracy for many FunCat classes as illustrated below in table 3. The optimization time and the number of iterations to achieve a consistent fitness score are given in table 3 below.

Table 3: Binary classification results from GAKNN

Function	Term	Time	Accuracy
2_0_0_0	Energy	1632226 ms	96.64
3_0_0_0	Cell cycle and DNA processing	4384 ms	73.09
4_0_0_0	Transcription	1629642 ms	85.87
5_0_0_0	Protein synthesis	5413 ms	88.79
6_0_0_0	Protein fate (folding, modification, destination)	1926545 ms	74.80
8_0_0_0	Cellular transport and transport mechanisms	1861736 ms	96.113
11_0_0_0	Cell rescue, defense and virulence	3885 ms	87.03
13_0_0_0	Regulation of / interaction with cellular environment	1754680 ms	96.52

The cross-validation method is used to find the accuracy of multi-class classification of gene expression datasets. GAKNN provides a separate tab in its interface to load the test data where classes are not labeled and to test the cross-validation process. The accuracy of cross-validation, true positive rate, false positive rate, false negative rate, true negative rate, precision, and recall are given for each gene functional class by GAKNN.

The first dataset tested is from NCBI where the gene functional scheme is Gene Ontology. Reference to the published paper which the dataset is used can be found at https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3846434/?report=classic.

The dataset was pre-processed to get a comma separated file. The three aspects of Gene Ontology functions are included in the dataset: Biological Process, Cellular Component, and Molecular Function. The training dataset consists of 1994 instances and 6 attributes.

The attributes are listed below:

Gene title

Gene symbol

Nucleotide title

GO Process

GO Component

GO Function

The missing molecular_functions (GO Functions) were predicted using the GAKNN and details are given below. The training dataset went through many iterations at first in the genetic algorithm in order to find the optimum k and weight values for each attribute. Figure 18 below shows the weight values assigned by genetic algorithm to each attribute.

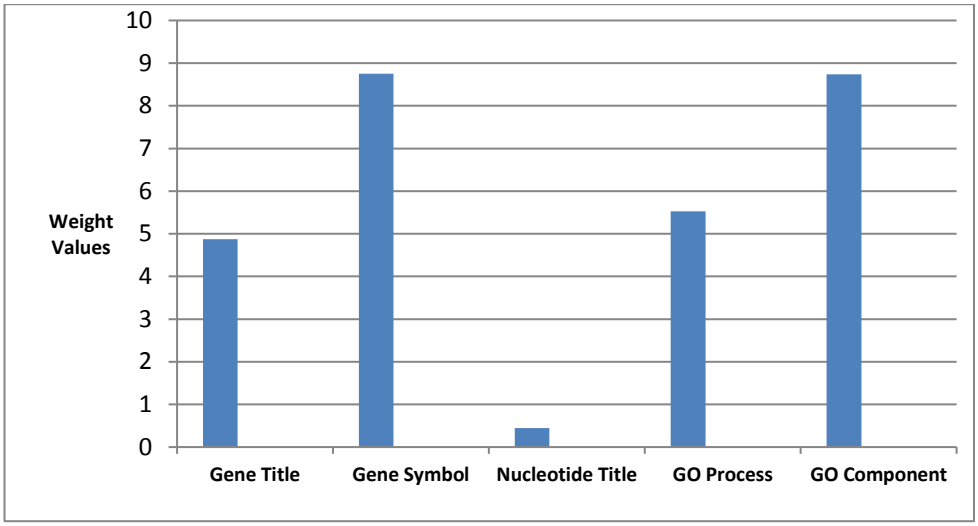

Figure 18: Weight values given by GAKNN for each attribute

We have chosen data instances where values of some "GO Function" attribute are missing and built a test dataset. The test dataset consists of 190 instances. The two *GO:molecular_function* classes predicted are binding and catalytic activity. The accuracy recorded for 50 and 65 iterations are shown in table 5 and table 6.

The accuracy recorded for 50 iterations are shown in table 4.

Accuracy: 72.8

Fitness score: 0.01752

k value: 9

Table 4: Accuracy recorded for 50 iterations in dataset 1

Class	True positive rate %	False positive rate %	Precision	Recall
Binding	0.149	0.016	0.9707	0.8846
Catalytic activity	0.005	0.019	0.2920	0.2054

After the training dataset went through 65 iterations in the genetic algorithm, there was no improvement in the fitness score. Therefore, the execution of the genetic algorithm stopped at a fitness score of 0.0175.

The accuracy recorded for 65 iterations are shown in table 5.

Accuracy: 74.72

Fitness score: 0.01754

k value: 8

Table 5: Accuracy recorded for 65 iterations in dataset 1

Class	True positive rate %	False positive rate %	Precision	Recall
Binding	0.966	0.827	0.539	0.999
Catalytic activity	0.981	0.898	0.522	0.999

The optimum value for k is 8 with an accuracy of 74.72. We tested the dataset from 5 to 65 iterations and following are the accuracy measures as seen in Figure 19 and 20. The accuracies got from GAKNN were compared with three other machine learning algorithms: Naïve Bayes Classification, standard k- Nearest Neighbor Classification, and Decision Trees. The figures below show the ROC (Receiver Operating Characteristic) curves generated by each algorithm and GAKNN outperforming other algorithms.

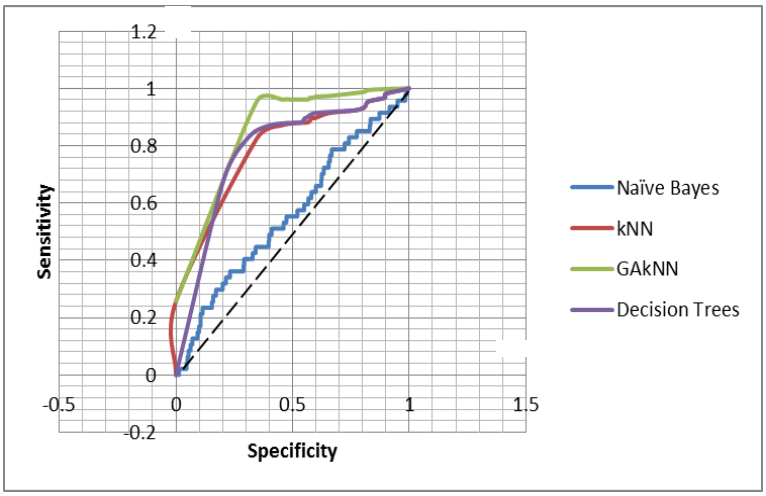

Figure 19: Accuracy measures given for *binding* in Gene Ontology dataset

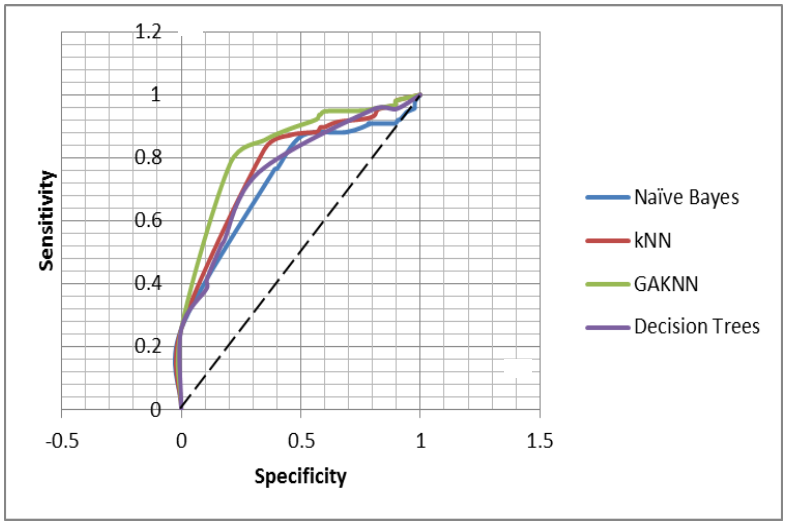

Figure 20: **Accuracy measures given for *catalytic activity* in Gene Ontology dataset**

Precision gives how many instances of binding function labels are correctly identified by the classifier. The recall is the number of binding function labels identified by the classifier. As for *catalytic activity*, GAKNN gives high precision-recall measurements relative to the number of instances falls into *catalytic activity*. Having high precision and recall is important to denote the high accuracy of the prediction done by GAKNN.

The second dataset was taken from MIPS and the published paper which the dataset is used can be found at http://www.aber.ac.uk/en/cs/research/cb/dss/yeastdata/, the Clare, A. and King R.D‘s publication on “Predicting gene functions in Saccharomyces cerevisiae.” The dataset contains 1340 instances and 16 attributes. The attributes are as follows:

1. mol_wt: Molecular weight
2. theo_pl: Theoretical pl (isoelectric point)
3. atomic_comp_c: Atomic composition of carbon
4. atomic_comp_o: Atomic composition of oxygen
5. atomic_comp_n: Atomic composition of nitrogen
6. atomic_comp_s: Atomic composition of sulfur

7. atomic_comp_h: Atomic composition of hydrogen
8. aliphatic_index: Aliphatic index
9. hydro: Grand average of hydropathicity
10. strand: The DNA strand which the ORF (open reading frame) lies
11. position: Number of exons (how many start positions are there in its coordinates list)
12. Number of motifs: according to PROSITE dictionary release 1 of Nov. 1995

Figure 21 below shows the weight values assigned to each attribute by the genetic algorithm. Figure 21 shows the precision and recall of each functional category in the dataset.

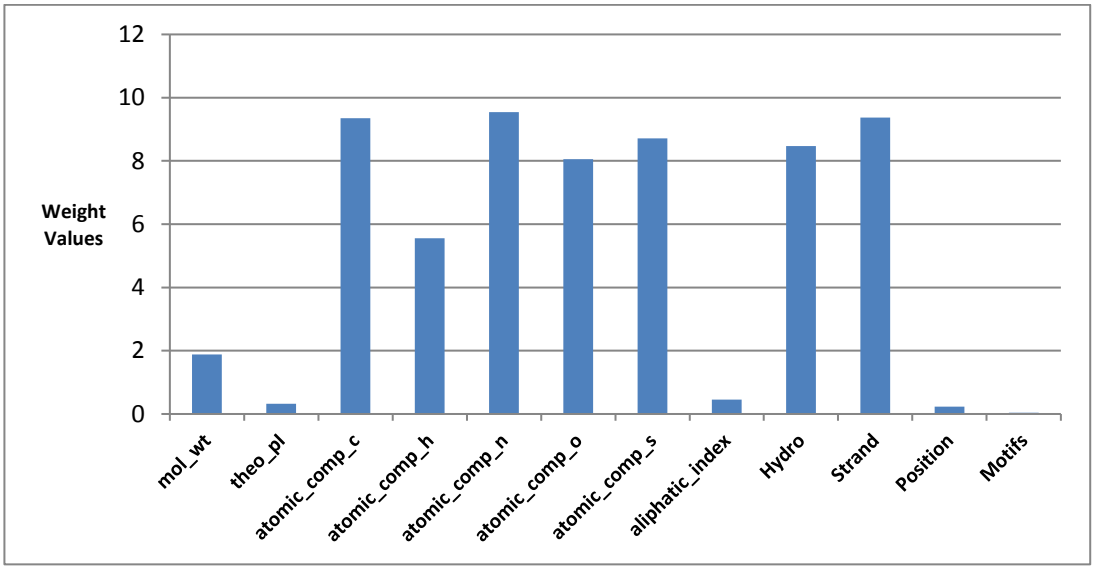

Figure 21: Weight values given to each attribute in dataset 2

The accuracy recorded for 25 iterations are given in table 6 below.

Accuracy: 63.77

Fitness score: 1.094

k value: 6

Table 6: Accuracy given for 25 iterations in dataset 2

Class	FunCat Term	True positive rate %	False positive rate %	Precision	Recall
40_0_0_0	Subcellular Localization	0.88	0.14	0.902	0.796
29_0_0_0	Transposable Elements, Viral and Plasmid Proteins	0.4	0.26	0.004	0.001
30_0_0_0	Control of Cellular Organization	0.014	0.061	0.014	0.044
3_0_0_0	Cell Cycle and DNA Processing	0.082	0.16	0.083	0.286

The accuracy recorded for 30 iterations are given below in table 7.

Accuracy: 65.268

Fitness score: 0.045

k value: 6

Table 7: Accuracy given for 30 iterations in dataset 2

Class	FunCat Term	True positive rate %	False positive rate %	Precision	Recall
40_0_0_0	Subcellular Localization	0.898	0.127	0.908	0.801
29_0_0_0	Transposable Elements, Viral and Plasmid Proteins	0.1	0.9	1.010	0.286
30_0_0_0	Control of Cellular Organization	0.015	0.061	0.014	0.048
3_0_0_0	Cell Cycle and DNA Processing	0.076	0.160	0.076	0.285

The accuracy recorded for 35 iterations are given below in table 8.

Accuracy: 83.420

Fitness score: 1.664

k value: 6

Table 8: **Accuracy given for 35 iterations in dataset 2**

Class	FunCat Term	True positive rate %	False positive rate %	Precision	Recall
40_0_0_0	Subcellular Localization	0.758	0.037	0.968	0.926
29_0_0_0	Transposable Elements, Viral and Plasmid Proteins	0.001	0.009	0.278	0.001
30_0_0_0	Control of Cellular Organization	0.001	0.006	0.002	0.013
3_0_0_0	Cell Cycle and DNA Processing	0.024	0.052	0.030	0.343

The accuracy recorded for 40 iterations are given below in table 9.

Accuracy: 88.67

Fitness score: 2.01

k value: 6

Table 9: Accuracy given for 40 iterations in dataset 2

Class	FunCat Term	True positive rate %	False positive rate %	Precision	Recall
40_0_0_0	Subcellular Localization	0.999	0.898	0.526	0.998
29_0_0_0	Transposable Elements, Viral and Plasmid Proteins	0.999	0.751	0.571	0.988
30_0_0_0	Control of Cellular Organization	0.981	0.897	0.522	0.988
3_0_0_0	Cell Cycle and DNA Processing	0.965	0.827	0.538	0.998

Figures 22, 23, 24, and 25 show accuracy given for each functional class in Functional Catalogue dataset by GAKNN, standard k-NN classification, Naïve Byes Classification, and Decision Trees. As illustrated by figures, GAKNN outperforms other machine learning algorithms in predicting the functional classes.

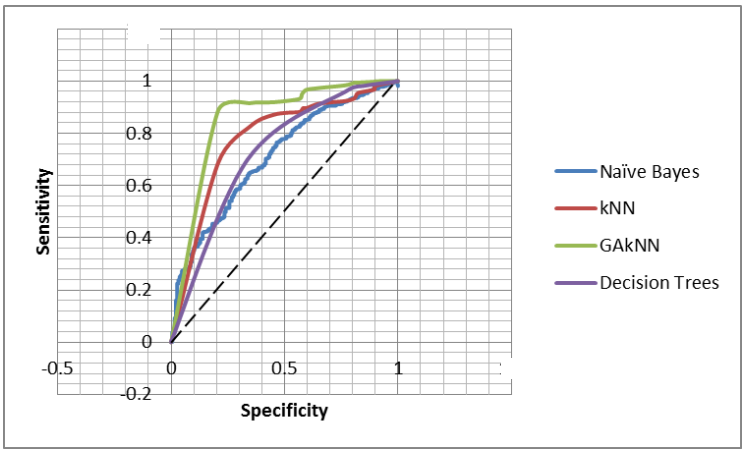

Figure 22: **Accuracy measures given for *subcellular localization* in FunCat dataset**

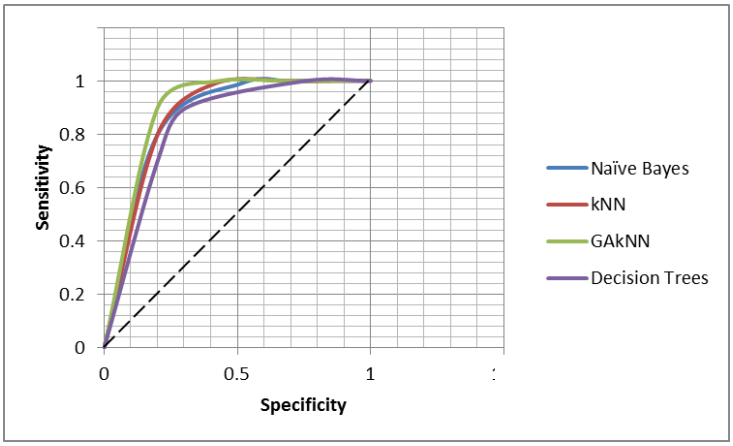

Figure 23: **Accuracy measures given for *transposable elements, viral, and plasmid proteins* in FunCat dataset**

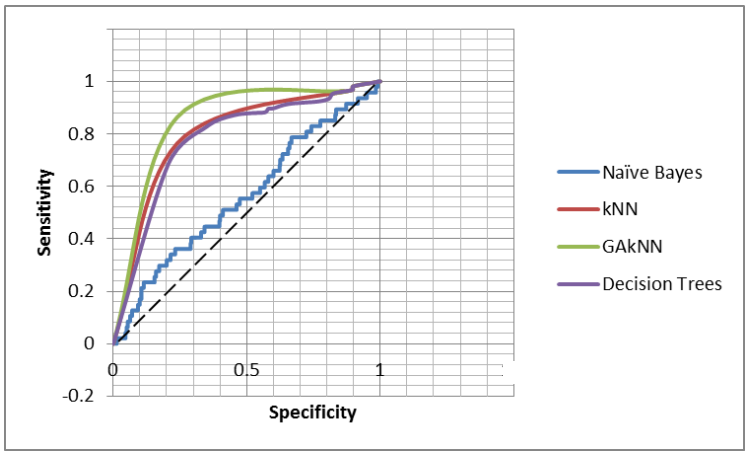

Figure 24: **Accuracy measures given for *cellular organization* in FunCat dataset**

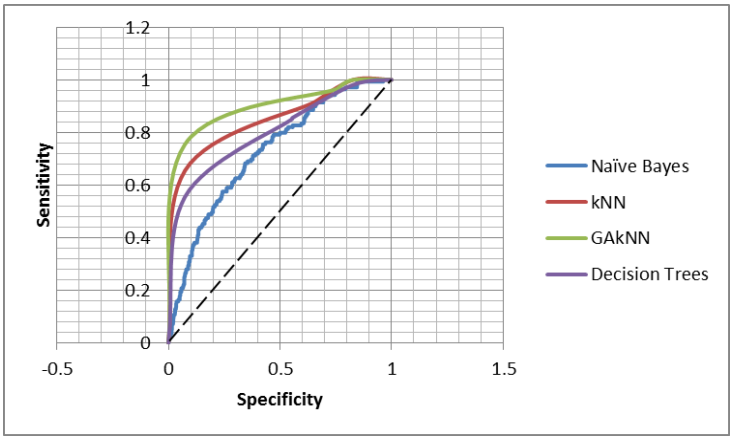

Figure 25: **Accuracy measures given for *cell cycle and DNA processing* in FunCat dataset**

The figure 26 below shows how the overall accuracy changed over the genetic algorithm iterations. The GO dataset achieved an accuracy over 70 with the genetic algorithm optimization and the FunCat dataset achieved an accuracy over 85 with the genetic algorithm optimization. This clearly shows how optimization works in GAKNN and how it has enabled to reach effective results.

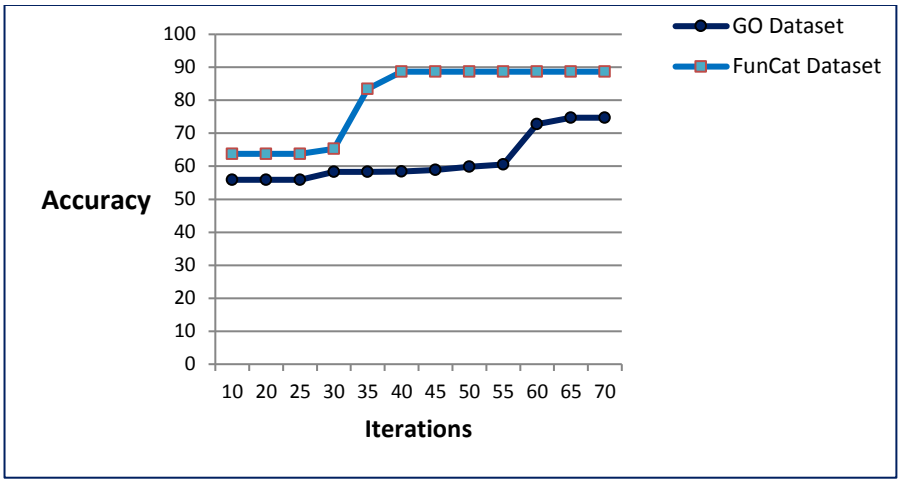

Figure 26: Accuracy over genetic algorithm iterations for Gene Ontology and FunCat datasets

Finally, the results from GAKNN were tested against WEKA under same conditions of prediction environments. We have set the parameter k similar for both WEKA's kNN (IBk) and GAkNN. The fitness score, number of iterations, and weights are only given when the datasets are run in GAkNN as the parameters mentioned before are specific to the genetic algorithm. Therefore, we tested the same

datasets with WEKA's kNN (IBk) with the default value of k and the optimum value of k assigned by the genetic algorithm. Table 10 below shows the results took from WEKA and GAKNN for the same datasets.

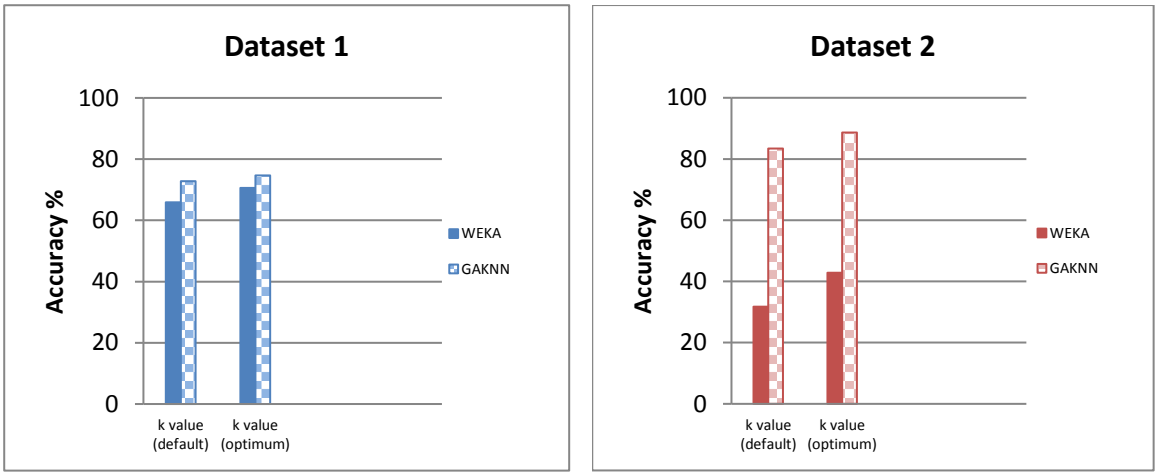

Figure 27: Comparison between WEKA and GAKNN accuracy measures

Table 10: Comparison between the results given by WEKA and GAKNN for last two iterations

Dataset 1	
WEKA	GAKNN
k value: 1 (default) Accuracy: 65.85%	k value: 9 Iterations: 50 Fitness score: 0.01752 Accuracy: 72.8%
k value: 8(optimum k value) Accuracy: 70.36%	k value: 8(optimum k value) Iterations: 65 Fitness score: 0.01754 Accuracy: 74.72%
Dataset 2	
WEKA	GAKNN
k value: 1 (default) Accuracy: 31.77%	k value: 6 Iterations: 35 Fitness score: 1.664 Accuracy: 83.42%

k value: 6(optimum k value) Accuracy: 42.79%	k value: 6(optimum k value) Iterations: 40 Fitness score:2.01 Accuracy: 88.67%

WEKA is a general data mining tool with kNN classification algorithm built in as IBk with standard parameters. BY using the optimum k value both WEKA and GAKNN achieved higher accuracy than with the default k value. This indicates the importance of finding the optimum k value for a dataset.

Our approach is to classify the gene function class labels using the genetic algorithm optimized kNN classification in order to overcome the inefficient predictive results and disadvantages of the standard kNN algorithm. As mentioned in the methodology, when the number of attributes is increasing the kNN classification algorithm tends to perform inefficiently due to distance measurements and inability to find the optimum k for a dataset [38]. The genetic algorithm is combined with k-NN to optimize the parameters, especially by assigning a weight vector to the attributes and by finding the optimum k. Once the weight vector and optimum k are determined, the performance is fast, efficient, and accurate by giving the complete solution to traditional manual curation process. Also, GAKNN is a computational solution which is employed to overcome the issues of sequence-based approach where there is insufficient homologous sequence data and of structure-based approach where there is a limited number of structure information in databases.

The software developed has been uploaded to SourceForge.net in beta version as a desktop environment solution at https://sourceforge.net/projects/gfp-gaknn/?source=navbar. The GAKNN setup file with its user manual can be found in the above link and screenshots of the uploaded setup file (22.5 MB) and download statistics are given in figure 23 and figure 24 respectively.

GAKNN is developed as a stand-alone application which can be installed on a general computer with 2GHz Processor, 2GB RAM to perform analysis of quite larger gene annotation datasets.

Figure 28: Home page and Description of software in SourceForge.net

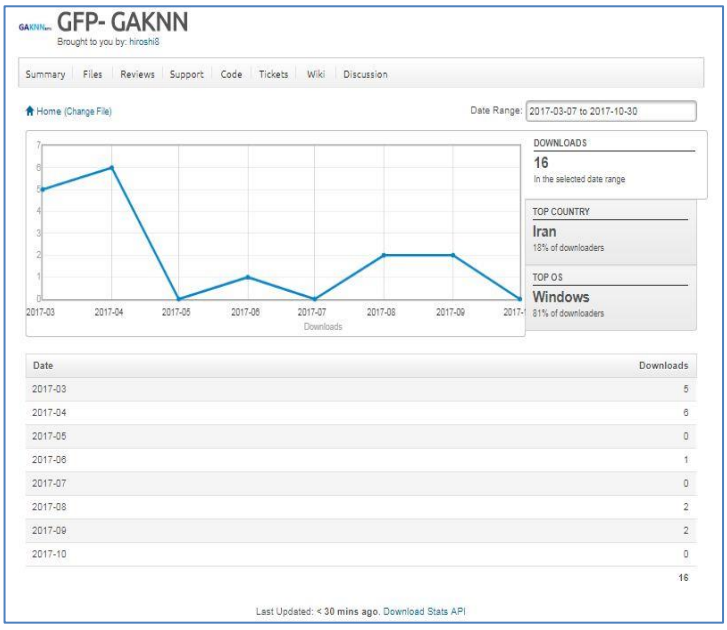

Figure 29: Downloads made by users in SourceForge.net

CHAPTER 5

5. Discussion

Gene function prediction has been the post-genomic era of bioinformatics where many types of research work are carried out to experiment and find out the functions related to genes. The experiments and annotations of functions have become tedious, extensive, and expensive in a laboratory environment as one small experiment to find one gene function takes few weeks to complete and annotate the findings. This happens due to heavy processes of methodologies in laboratories with various equipment to compile one small task of analyzing experimental cells, genes, and proteins. The most common approach to function recognition has been the sequence similarity methods like BLAST and PSI-BLAST. The sequence similarity alone has failed to determine the functions of newly sequenced genomes. Therefore, additional information was combined with computational approaches to provide accurate function prediction. As a computational approach, machine learning algorithms have come a long way in providing fast and accurate function predictions cutting down the inefficiency of manual curation processes done in laboratories. Neural networks, decision tress, kNN, and support vector machine approaches were used tremendously in function classification over the past years but these algorithms tend to give inefficient results due to a large number of attributes. Out of the algorithms mentioned above, kNN performed well to the datasets with a large number of attributes. The gene annotation datasets we use for function prediction always have many attributes which make the dimensional complex processing. As an efficient machine learning approach to function prediction, we have proposed a solution based on genetic algorithm optimized k- Nearest Neighbor to find out gene functions given the annotation data named GAKNN. The genetic algorithm is used for optimization by assigning weights to each attribute in the dataset. Also, by finding the optimum k for a dataset, the algorithm is capable of reducing the search space for a dataset with a large number of attributes. In order to provide a more user-friendly approach, we have developed our solution with a user interface where one can upload gene annotation data, pre-process, optimize,

and predict the unknown functions of the genes. The developed solution is available on SourceForge at https://sourceforge.net/projects/gfp-gaknn/?source=navbar.

GAKNN is compatible with many file formats and functional categories. We have tested .arff, .txt, and .csv data with GAKNN. The results given are from two different functional categories: Gene Ontology and MIPS Functional Catalogue. Before optimizing, data can be pre-processed by feature selection and missing data imputation. Feature selection is based on the user where the user can select the features to be in the dataset. Missing data imputation is done by the proposed Evolutionary k- Nearest Neighbor Imputation algorithm. Imputation process has been tested with three datasets namely *gasch2*, *spo*, and *seq*. In the *gasch2* dataset, the content of missing rates ranged between 0.05%- 25%. The *spo* dataset consisted of 0.02% - 0.16% of missing rates. The *seq* dataset had missing rates between 0.06% - 0.6%. We also proposed an imputation algorithm named EvlkNNImputation for missing data imputation purpose. EvlkNNImputation algorithm was compared and tested against kNNImpuation and Mean Imputation algorithms. EvlkNNImputation gave the least mean error rates among the other two imputation methods because of the evolutionary kNN imputation based on the genetic algorithm optimization. After the pre-processing steps, the optimization begins with the execution of the genetic algorithm. The user has to state the number of iterations to run in the genetic algorithm. The number of iterations in the genetic algorithm can be determined by the fitness score increment shown in the graph of GAKNN. After the optimization, the trained data model should be saved and proceed to function prediction for an unlabeled dataset.

The final gene function prediction results depend on the optimized k value and the fitness score of the genetic algorithm. Optimization can be carried out for several iterations until there is no improvement in the fitness score. We used two datasets from two different sources for testing. The first dataset is a human dataset which has gene ontology functions in it. The dataset was pre-processed to remove missing values and to have gene functions classified at an upper level such as binding or catalytic activity. Out of the three GO functional

categories in the first dataset we took, Biological Process and Cellular Component have the highest weight values when predicting the missing Molecular Functions. The prediction of Molecular Functions very much depends on Biological Process and Cellular Component as Biological Process and Cellular Component have a major impact towards the Molecular Function. Also, the Gene Symbol attribute holds a high weight value which means Gene Symbol is quite an important annotation attribute when it comes to predicting missing gene functions. The optimization stopped at 65^{th} iteration as there was no improvement in the fitness score and the accuracy recorded is 74.72% for the GeneOntology dataset.

The second dataset we took has more attributes which increase the horizontal dimension complexity. The dataset is the yeast dataset in level 1of MIPS functional category. In this dataset, we got 4 functional classes in level 1: subcellular localization, transposable elements viral and plasmid proteins, control of the cellular organization, cell cycle and DNA processing. By observing the weights assigned by the genetic algorithm to attributes, it can be clearly seen that the most important attributes in predicting the functional class labels are getting the higher values. The attributes such as the atomic composition of Carbon, Hydrogen, Nitrogen, Oxygen, Sulphur play a vital role when predicting the functional classes. After the 40^{th} iteration, the accuracy is not increasing as there is no improvement in the fitness score. The accuracies recorded are 83.42 for 35 iterations and 88.67 for 40 iterations respectively.

CHAPTER 6

6. Conclusion

GAKNN is a platform specially designed to work on gene annotation data by following the data mining approach to predict the functions of genes which is a major need in the field of functional genomics. Upon completing the genome sequencing, the functions of the genes are yet to be determined. As genome sequence projects continue to produce more sequence data, many approaches are used to discover the unknown functions of the genes. Among the function prediction approaches, manual curation process has become inefficient with the time it takes and the fact that the sequence alone could not be taken for function prediction in many proteins. At present, many machine learning approaches are tested to predict the gene functions but the accuracy and the prediction of unknown functions have become main challenges to scientists. The solution we developed support the prediction of unknown gene functions with an optimized algorithm to provide more accurate results in prediction. GAKNN follows the steps of knowledge discovery process including data pre-processing, data optimization, and functional class prediction.

There are several pre-processing steps included in GAKNN specially developed for gene annotation data. Gene annotation data are generated from microarray analysis and there is the tendency of having missing values in a dataset due to machine errors and insufficient resolution, corrupted images, and dust or scratches on the slide. As a missing data imputation method, we also have developed Evolutionary k-Nearest Neighbor Imputation algorithm apart from the function prediction algorithm in order to provide a more robust solution in data pre-processing. Most datasets having missing values can be recovered by using machine learning imputation methods. Furthermore, the results obtained from the study has proved the efficiency of using a machine

learning approach for imputation and also the advantages of using GAKNN imputation because the mean imputation method gave the highest error rates compared to kNNImputation and EvlkNNImputation when the size of data got increased. Even though both kNNImputation and EvlkNNImputation are machine learning algorithms, EvlkNNImputation shows the best results due to its genetic algorithm optimization. The other data pre-processing steps include feature selection and deletion of records by the user. The workspace provided by GAKNN has a white-box approach in showing all the mechanisms of data refinement.

We have developed GAKNN to be more scalable and interoperable of functional schemes and data file formats. Despite the data pre-processing methods, GAKNN can be used to run different formats of gene annotation data files including .txt, .csv, and .arff. GAKNN supports Gene Ontology, FunCat, and many other functional category schemes. The learning capability of the genetic algorithm is important in predicting the functional classes. That way, the prominent attributes towards predicting the gene function gets higher weight values. The genetic algorithm also finds out the optimum k for a dataset. Combining the genetic algorithm with k- Nearest Neighbor algorithm helped to overcome the disadvantages faced by the standard kNN algorithm which are the high computational cost and the inconsistent performance when the number of attributes gets increased. Furthermore, by finding the weight vector and the optimum k value have reduced the performance issues when it comes to large datasets with kNN. The gene function test datasets are taken from the upper levels of functional categories. The *Homo sapiens* dataset in Gene Ontology gave an accuracy of 75% in its 65^{th} iteration. The yeast dataset in FunCat gave an accuracy of 88% in its 40^{th} iteration. The termination of genetic algorithm execution happened when the algorithm gives a consistent value for fitness score after certain iteration. The fitness score increment can be monitored with the fitness score graph generated in GAKNN. Also, the weight chart shows the

weight values given for each attribute over the iterations. The assignment of weight values and fitness score optimization is provided separately in GAKNN which will be enabled after the data pre-processing.

After the optimization, the trained model is used for function prediction in datasets with unknown functional classes. Though many approaches available at present are capable in the prediction of known gene functional classes, these approaches seem to be failed in the prediction of unknown functional classes[2]. GAKNN has performed well in predicting the unknown functional classes using gene annotation data rather than relying on sequence data and has shown effective performance of predicting functional classes of genes. We have compared GAKNN algorithm with standard kNN, Decision Trees, and Naïve Bayes Classification algorithms and GAKNN algorithm outperform the machine learning algorithms mentioned above. GAKNN gave the closer ROC curve to the upper left corner of the graph depicting the higher accuracy it performed.

The results were compared with WEKA, a generic data mining solution under the default k value and optimum k value. By applying the k value generated by the genetic algorithm to both WEKA and GAKNN, accuracies got higher than with the default k value. Therefore, we can conclude that finding the optimum k value for a dataset has improved the accuracy of function prediction. GAKNN performed well in predicting functional classes by giving far more accurate results than WEKA for a given dataset. As many function prediction solutions rely on sequence alignment, GAKNN developed as a solution working on gene expression data. GAKNN helps to analyze datasets and predict unknown functional classes. The use of gene expression data is a knowledge-based approach towards predicting gene functions which is valuable in the annotation of functional genomics at present where sequence alignment alone has failed to predict more accurate results in function prediction.

References

[1] L. Han *et al.*, "Recent progress in the application of machine learning approach for predicting protein functional class independent of sequence similarity.," *Proteomics*, vol. 6, pp. 4023–4037, Mar. 2006.

[2] A. Al-Shahib, R. Breitling, and D. R. Gilbert, "Predicting protein function by machine learning on amino acid sequences – a critical evaluation," *BMC Genomics*, vol. 8, p. 78, Mar. 2007.

[3] A. Brazma and J. Vilo, "Gene expression data analysis," *FEBS Lett.*, vol. 480, no. 1, pp. 17–24, Aug. 2000.

[4] D. V. Nguyen, A. B. Arpat, N. Wang, and R. J. Carroll, "DNA microarray experiments: biological and technological aspects," *Biometrics*, vol. 58, no. 4, pp. 701–717, Dec. 2002.

[5] M. Kanehisa and S. Goto, "KEGG: Kyoto Encyclopedia of Genes and Genomes," *Nucleic Acids Res.*, vol. 28, no. 1, pp. 27–30, Jan. 2000.

[6] M. H. Serres and M. Riley, "MultiFun, a multifunctional classification scheme for Escherichia coli K-12 gene products," *Microb. Comp. Genomics*, vol. 5, no. 4, pp. 205–222, 2000.

[7] A. Ruepp *et al.*, "The FunCat, a functional annotation scheme for systematic classification of proteins from whole genomes," *Nucleic Acids Res.*, vol. 32, no. 18, pp. 5539–5545, 2004.

[8] "The Gene Ontology project in 2008," *Nucleic Acids Res.*, vol. 36, no. Database issue, pp. D440–D444, Jan. 2008.

[9] M. Ashburner *et al.*, "Gene ontology: tool for the unification of biology. The Gene Ontology Consortium," *Nat. Genet.*, vol. 25, no. 1, pp. 25–29, May 2000.

[10] P. Bork, T. Dandekar, Y. Diaz-Lazcoz, F. Eisenhaber, M. Huynen, and Y. Yuan, "Predicting function: from genes to genomes and back," *J. Mol. Biol.*, vol. 283, no. 4, pp. 707–725, Nov. 1998.

[11] I. M. Keseler *et al.*, "EcoCyc: a comprehensive database resource for Escherichia coli," *Nucleic Acids Res.*, vol. 33, no. Database issue, pp. D334-337, Jan. 2005.

[12] D. H. Haft, J. D. Selengut, and O. White, "The TIGRFAMs database of protein families," *Nucleic Acids Res.*, vol. 31, no. 1, pp. 371–373, Jan. 2003.

[13] H. W. Mewes *et al.*, "MIPS: a database for genomes and protein sequences," *Nucleic Acids Res.*, vol. 30, no. 1, pp. 31–34, Jan. 2002.

[14] "Gene Ontology Consortium." [Online]. Available: http://geneontology.org/. [Accessed: 27-Mar-2016].

[15] V. G. Krishnan and D. R. Westhead, "A comparative study of machine-learning methods to predict the effects of single nucleotide polymorphisms on protein function," *Bioinformatics*, vol. 19, pp. 2199–2209, May 2003.

[16] A. A. Freitas, D. C. Wieser, and R. Apweiler, "On the importance of comprehensible classification models for protein function prediction," *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, vol. 7, no. 1, pp. 172–182, Jan. 2010.

[17] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, "Basic Local Alignment Search Tool," *J.Mol.Biol*, pp. 403–410, Oct. 1990.

[18] W. R. Pearson and D. J. Lipman, "Improved tools for biological sequence comparison," *Proc. Natl. Acad. Sci. USA*, vol. 85, pp. 2444–2448, Apr. 1988.

[19] T. Smith and M. Wayerman, "Identification of common molecular subsequences," *J. Mol. Biol.*, pp. 195–197, 1981.

[20] A. Bairoch, P. Bucher, and K. Hofmann, "The PROSITE database, its status," *Nucleic Acids Research*, vol. 25, pp. 217–221, Oct. 1996.

[21] T. . Attwood, M. . Beck, A. . Bleasby, and D. . Parry-Smith, “PRINTS- a database of protein motif fingerprints,” *Nucleic Acids Research*, vol. 22, pp. 3590–3596, 1994.
[22] “Sequence Motifs: Highly Predictive Features of Protein Function,” in *Feature Extraction*, vol. 207, Springer Berlin Heidelberg, 2006, pp. 625–645.
[23] S. F. Altschul *et al.*, “Grapped BLAST and PSI-BLAST: a new generation of protein database search programs,” *Nucleic Acids Research*, vol. 25, pp. 3389–3402, Jul. 1997.
[24] L. Holm and C. Sander, “Protein Structure Comparison by Alignment of Distance Matrices,” *J. Mol. Biol.*, pp. 123–138, 1993.
[25] J. C. Whisstock and A. M. Lesk, “Prediction of protein function from protein sequence and structure,” *Quarterly Reviews of Biophysics*, vol. 36, no. 3, pp. 307–340, Aug. 2003.
[26] D. Wang and B. Larder, “Enhanced Prediction of Lopinavir Resistance from Genotype by Use of Artificial Neural Networks,” *The Journal of Infectious Diseases*, Mar. 2003.
[27] L. J. Lancashire, C. Lemetre, and G. R. Ball, “An introduction to artificial neural networks in bioinformatics-application to complex microarray and mass spectrometry datasets in cancer studies.,” *Briefings in Bioinformatics*, Feb. 2009.
[28] M.-H. Chae, F. Krull, S. Korenzen, and E.-W. Knapp, “Predicting protein complex geometrics with a neural network,” *Proteins*, Oct. 2009.
[29] N. Donges, “Pros and Cons of Neural Networks,” *Towards Data Science*, 17-Apr-2018. [Online]. Available: https://towardsdatascience.com/hype-disadvantages-of-neural-networks-6af04904ba5b. [Accessed: 10-Jun-2018].
[30] joshiblog, “Advantages and disadvantages of hidden markov model,” 10:38:33 UTC.
[31] V. De Fonzo, F. Aluffi-Pentini, and V. Parisi, “Hidden Markov Models in Bioinformatics,” *Current Bioinformatics*, pp. 49–61, 2007.
[32] W. Majoros, M. Pertea, and S. Salzberg, “Efficient implementation of a generalized pair hidden Markov model for comparatuve gene finding.,” *Bioinformatics*, pp. 1782–1788, Fenruary 2005.
[33] Z. Yang, “Biological applications of support vector machines,” *Breif Bioinform*, pp. 328–338, Dec. 2004.
[34] L. Schietgat, C. Vens, J. Struyf, H. Blockeel, D. Kocev, and S. Dzeroski, “Predicting gene function using hierarchical multi-label decision tree ensembles,” *BMC Bioinformatics*, vol. 11, Jan. 2010.
[35] C. Vens, J. Struyf, L. Schietgat, S. Dzeroski, and H. Blockeel, “Decision trees for hierarchical multi-label classification,” vol. 73, no. 2, pp. 185–214, Nov. 2008.
[36] S. L. Wong *et al.*, “Combining biolgical networks to predict genetic interactions,” *PNAS*, Sep. 2004.
[37] D. Che, Q. Liu, K. Rasheed, and X. Tao, “Decision tree and ensemble learning algorithms with their applications in bioinformatics,” *Adv Exp Med Biol.*, 2011.
[38] Y. Huang and Y. Li, “Prediction of protein subcellular locations using fuzzy k-NN method,” *Bioinformatics*, vol. 20, no. 1, Apr. 2003.
[39] P. Horton *et al.*, “WoLF PSORT: protein localization predictor,” *Nucl. Acids Res.*, vol. 35, Apr. 2007.
[40] L. Lan, N. Djuric, Y. Guo, and S. Vucetic, “MS-kNN: protein function prediction by integrating multiple data sources,” *BMC Bioinformatics*, vol. 14, Jul. 2010.
[41] S. M. Thede, “An Introduction to Genetic Algorithms,” *Journal of Computing Sciences in Colleges*, vol. 20 Issue 1, pp. 115–123, Oct. 2004.
[42] D. J. Sobajic, *Neural Network Computing for the Electric Power Industry: Proceedings of the 1992 Inns Summer Workshop*. Psychology Press, 2013.
[43] C.-F. Tsai, W. Eberle, and C.-Y. Chu, “Genetic algorithms in feature and instance selection,” *Knowl.-Based Syst.*, vol. 39, pp. 240–247, Feb. 2013.
[44] A. S. Perera and W. Perrizo, “Gene Function Prediction.,” in *ResearchGate*, 2009, pp. 26–31.

[45] M. Middlemiss and G. Dick, “Design and Application of Hybrid Intelligent Systems,” A. Abraham, M. Köppen, and K. Franke, Eds. Amsterdam, The Netherlands, The Netherlands: IOS Press, 2003, pp. 519–527.
[46] J. Brownlee, “Supervised and Unsupervised Machine Learning Algorithms,” *Machine Learning Mastery*, 16-Mar-2016. .
[47] Z. Yao and W. L. Ruzzo, “A Regression-based K nearest neighbor algorithm for gene function prediction from heterogeneous data,” *BMC Bioinformatics*, vol. 7, no. Suppl 1, p. S11, Mar. 2006.
[48] “Data Mining Concepts.” [Online]. Available: https://docs.oracle.com/cd/B19306_01/datamine.102/b14339/3predictive.htm. [Accessed: 07-May-2016].
[49] S. M. Thede, “An Introduction to Genetic Algorithms,” *J Comput Sci Coll*, vol. 20, no. 1, pp. 115–123, Oct. 2004.
[50] “Han and Kamber: Data Mining---Concepts and Techniques, 2nd ed., Morgan Kaufmann, 2006.” [Online]. Available: http://web.engr.illinois.edu/~hanj/bk2/slidesindex.htm. [Accessed: 11-Mar-2017].
[51] J. L. DeRisi, V. R. Iyer, and P. O. Brown, “Exploring the metabolic and genetic control of gene expression on a genomic scale,” *Science*, vol. 278, no. 5338, pp. 680–686, Oct. 1997.
[52] C. M. Perou *et al.*, “Molecular portraits of human breast tumours,” *Nature*, vol. 406, no. 6797, pp. 747–752, Aug. 2000.
[53] S. Chu *et al.*, “The transcriptional program of sporulation in budding yeast,” *Science*, vol. 282, no. 5389, pp. 699–705, Oct. 1998.
[54] S. Friedland, A. Niknejad, M. Kaveh, and H. Zare, “An Algorithm for Missing Value Estimation for DNA Microarray Data,” in *2006 IEEE International Conference on Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings*, 2006, vol. 2, pp. II–II.
[55] A. Gelman and J. Hill, “Missing-data imputation,” in *Data Analysis Using Regression and Multilevel%2FHierarchical Models*, Cambridge University Press, 2006.
[56] A. Mockus, “Missing Data in Software Engineering,” in *Guide to Advanced Empirical Software Engineering*, F. Shull, J. Singer, and D. I. K. Sjøberg, Eds. Springer London, 2008, pp. 185–200.
[57] E. Acuña and C. Rodriguez, “The Treatment of Missing Values and its Effect on Classifier Accuracy,” in *Classification, Clustering, and Data Mining Applications*, D. D. Banks, D. F. R. McMorris, D. P. Arabie, and P. D. W. Gaul, Eds. Springer Berlin Heidelberg, 2004, pp. 639–647.
[58] J. Kaiser, “Algorithm for Missing Values Imputation in Categorical Data with Use of Association Rules,” *ArXiv12111799 Cs*, Nov. 2012.
[59] J. C. Wayman and P. D, *Multiple Imputation for Missing Data: What is It and How Can I Use It*. 2003.
[60] A. P. Gasch, M. Huang, S. Metzner, D. Botstein, S. J. Elledge, and P. O. Brown, “Genomic expression responses to DNA-damaging agents and the regulatory role of the yeast ATR homolog Mec1p,” *Mol. Biol. Cell*, vol. 12, no. 10, pp. 2987–3003, Oct. 2001.