IDENTIFICATION OF PROTEIN COMPLEXES USING MACHINE LEARNING (PyBrain and Scikit-Learn) BASED ON DNA SEQUENCE DATA

A Thesis

by

WUTHIWAT RUANGCHAI

Submitted to the Office of Graduate Studies
Texas A&M University-Commerce
In partial fulfillment of the requirements
For the degree of
MASTER OF SCIENCE
Graduation December 2014

IDENTIFICATION OF PROTEIN COMPLEXES USING MACHINE LEARNING (PyBrain and Scikit-Learn) BASED ON DNA SEQUENCE DATA

A Thesis

by

WUTHIWAT RUANGCHAI

Approved by:	
Advisor:	Sang C. Suh
Committee:	Abdullah Arslan Jinoh Kim
Head of Department:	Sang C. Suh
Dean of the College:	Brent Donham
Dean of Graduate Studies:	Arlene Horne

ABSTRACT

IDENTIFICATION OF PROTEIN COMPLEXES USING MACHING LEARNING (PyBrain and Scikit-Learn) BASED ON DNA SEQUNCE DATA

Wuthiwat Ruangchai, MS
Texas A&M University-Commerce, 2014

Advisor: Sang C. Suh Ph.D.

The National Center for Biotechnology Information (NCBI) provides various information that relate to science and health such as program downloads, databases, submissions, tools, and protocols. The database section is separated into many subsections such as Bioproject (formerly Genome Project), BioSample, Bookshelf, GenBank, and Nucleotide Database. This study especially focused on Nucleotide or DNA sequence database. With such large size data sets, researchers are able to use various theories and algorithms to extract and mine the knowledge. This study applied machine learning approaches for classification and identification of protein complexes by using DNA sequence inputs.

The construction of artificially intelligent systems that take in and analyze data in order to improve themselves and create better interaction with the data is called machine learning. The system effectively learns how to do their job better. Machine learning is the most useful tool for research and an exquisite sample of learning from examples (Love, 2014). This study used Python-Based Reinforcement Learning Artificial Intelligence and Neural Network Library (PyBrain) as a modular machine learning library and supervised learning algorithms as structure algorithms inside the machine.

In this study, the data were separated into two sets. The first set of the data was used to train the machines that had different algorithm structures. The second set of the data was used to test the accuracy of the machines. The machine models were built with specific parameters. They were trained and tested by the datasets. The results from the models were visualized by using line charts and clustered column charts.

From the result of six types of protein complex datasets, the machine that had the best accuracy and learning rate was the Resilient propagation machine model, with 99.98% accuracy and a fast learning rate, compared with others. The accuracy of Back propagation machine model was 97.76%. The accuracy of support vector machine models was 93.60%. The accuracy of stochastic gradient descent machine model was 98.38%.

ACKNOWLEDGEMENTS

I have a great appreciation for everyone who contributed time and assistance to my research work and thesis. First and foremost, I would like to thank my advisor, Dr. Sang Suh. There are not enough words to express my gratitude for the guidance, encouragement and support he has given me. He has been an inspiration to me. I would like to sincerely thank Dr. Jinoh Kim for his tutelage, motivation, and direction, which have greatly helped me achieve my goals. I would like to express thanks to Dr. Abdullah Arslan for serving on my committee. Lastly, I give many thanks to God for the blessings and abilities He has given me.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

Singer and Magazine (2013) stated, “The life sciences are becoming a big data enterprise” (para. 6). Biology is stepping into a big data era, and biologists find themselves unable to use and extract full information from the vast amounts of data that are becoming available. Around five years ago, DNA sequencing costs decreased faster than the cost of a computer chip. Since then, many animals, plants, and other organisms’ genomes have been interpreted. The very well-known public genome repositories, National Center for Biotechnology Information (NCBI) already houses 10^{15} bytes of data (Singer, & Magazine, 2013).

Statement of the Problem

In the past, an area of protein study lacked sufficient data to analyze, but with the evolution of genetic instruments and the internet, small biology labs or individual researchers can generate and access many data sources. For this reason, protein researchers now have enough data with which to work. For a protein complex analysis, the researchers need the internet for accessing information from a database. With the machine models in this study, the researchers did not need to access any databases because the machine already had the information for conducting a classified report.

From the machine learning theory, the performance of the machine depends on the amount and accuracy the data. This study used data that already existed to train the machine. The machine could identify and predict a protein complex from a DNA sequence input, and it also calculated the percentage of similarity for proteins.

Purpose of the Study

The broad goal of this research was to build a learning machine that could identify and predict protein complexes, with a DNA sequence input and to calculate the similarity values of the input sequence. The machines were trained by data from a DNA sequence database, by using supervised learning algorithms.

The performance and accuracy of the machine depends on many factors. This study explored which was the best combination of the machines and supervised learning algorithms for protein identifications. The performances of several machine models during the training section and end product machine models were compared for accuracy of classification. With the online databases, this study could access to the large datasets. One of the factors that affects the performance and accuracy of the models is the size of datasets that is used in training the models. This study also tested how the models perform when using large data sets as training data sets.

Research Questions

1. How is the machines performance with different structures of supervised learning algorithms that is implemented to the machine?
2. How is the accuracy of the machine for identification and prediction in protein complexes with a DNA sequence input?
3. How are the model performances with the large training data sets?

Significance of the Study

A DNA sequence is a product of DNA sequencing, which is a method to determine the precise order of four base nucleotides (adenine, guanine, cytosine, and thymine), within a DNA molecule. Knowledge of DNA sequences have become essential for basic biological research, biotechnology, forensic biology, and biological systematics (Pettersson, 2009). The relationship between DNA and a protein complex is a result of products from biological processes that start with DNA. These biological processes have three major steps. In the first step, the information in DNA is transferred to a messenger RNA molecule by a process called transcription. During the second step, a messenger RNA molecule is transferred to a protein molecule by a process called translation. Throughout the last step, multiple protein molecules are formed into a protein complex (Brown, & Brown, 2008).

A protein complex is a group of two or more polypeptide chains and forming a quaternary structure. The protein complex can have multiple functions if polypeptide chains contain different protein domains. Proteins in protein complexes are linked by non-covalent protein-protein interactions, and different protein complexes have different degrees of stability over time. The protein complexes are a cornerstone of many biological processes and perform many biological functions (Hartwell, & Hopfield, 1999).

Since protein complexes play an important role in biological cells, predicting an unknown protein is an essential goal in bioinformatics. Computational methods for predicting and classifying protein complexes are necessary in order to de determine unknown protein complexes in a fast and more cost-effective manner. Approaches based on sequence and structure comparisons play an important role in predicting and classifying   the unknown protein complexes. Generally, an unknown protein complex is identified by using sequence similarity search methods, such as Basic Local Alignment Search Tool (BLAST) and FASTA (Lee, Leopold, & Frank, 2009). Rather than making predictions based on direct sequences, this study used a machine learning approach to identify protein complexes.

"A computer program is said to learn from experience E with respect to some class of task T and performance measure P if its performance at tasks in T as measured by P, improves with experience E" (Mitchell, 1997, p. 2). Machine learning is a system of artificial intelligence (AI) that can learn from data. The main focuses of machine learning are representation and generalization. Representation of the data is the methods that are used to represent information from the data in a computer. Generalization of the data is the property that the system performs on unknown data based on the experience learned from data sets. Taxonomy can separate machine learning algorithms based on the outcome of the algorithm and the type of information while training the machine. This research focused on supervised learning algorithms. Supervised learning algorithms are trained on input where an output is known. These algorithms try to generalize mapping from the input to output that can be used to predict the output from unseen input data.

A learning tool which was built as part of this thesis work was useful for scientific research of big data (large size of data) in the field of medicine, biology, chemistry, physics, and others where a large-scale of data were analyzed. There are several ways to apply a learning machine. In this research, machine learning was implemented to identify and predict protein complexes by teaching the machine with a large scale data set. With the machine, users did not need to be able to access a whole database because the machine already contained information on which users needed to be able to perform analysis. For this reason, machine learning gives the user easier access to perform analysis and also the freedom to create their machine by training it with the users' data.

Method of Procedure

The proposed approaches were to build the learning machine that could identify protein complexes and to study combinations of algorithms' structures, which gave the best predicted accuracy. The experiments performed on Python shell and code on Python language as well. Modular machine learning libraries for Python that were used in this thesis were Python-Based Reinforcement Learning Artificial Intelligence and Neural Network (PyBrain) and Scilkit–learn: Machine learning in Python. Two supervised learning algorithms that were implemented from Scilkit- learn were Support Vector Machines (SVM) and Stochastic Gradient Descent (SGD). Another group of supervised learning algorithms that were implemented from PyBrain were Back Propagation (Backprop) and Resilient Propagation (Rprop). Final analyzed results were visualized for easier presentation and understanding. This thesis was separated into six steps. The important steps involved in the process of this thesis are as below.

Collection of Data

The six protein complexes were selected from three biological functions. Two protein complexes from a chemical digestion system were Amylase and Dipeptidase. Two protein complexes from an energy and oxygen cell transportation system were F1-ATP and Hemoglobin. Two protein complexes from a generated energy system in the cell are Photosynthetic reaction centre protein and Plastocyanin protein. DNA sequences of the samples were collected from NCBI Nucleotide Database based on the protein complexes that were selected.

Preparation of Data

From the NCBI Nucleotide Database, the protein complex sample data sets came as raw data sets that include ID numbers, names, and DNA sequences. In the data sets, only the DNA sequences needed to be extracted. Each protein complex data sets were separated into two groups. The First group was a learning group. The machine was trained by this group. The second group was a testing group. This group was used to test the accuracy of the machine.

Creation of Machine Structures

The machine models were created in different combinations of the structures by using PyBrain and Scilkit–learn libraries on Python shell. Two machine models were implemented with two supervised learning algorithms from Scilkit-learn library that were SVM algorithm and SGD algorithm. Another two machine models were implemented with two supervised learning algorithms from PyBrain library that were Backprop algorithm and Rprop algorithm. Then, each machine was trained by all the learning data sets from six difference types of protein complexes.

Machine Testing

After the machines were trained, each machine was tested by using testing data groups from six types of protein complexes. In the testing process, the inputs for the machines only had DNA sequence sample. The machines should have predicted and identified protein complexes and also given a familiarity percentage of each protein complexes. The results (output) were labeled and categorized based on the machines and the types of protein complexes.

Analysis and Visualization of the Result

The results were analyzed by checking predicted protein complexes with the recorded data, in the testing data sets. From the analysis, the accuracy of each machine was calculated. The accuracies of the machine and the familiarity percentages were visualized for each machine.

The Resulting Conclusion

This study looked for the combination of structures machine models and algorithms that gave the best accuracy result and familiarity percentage from the analyzed and visualized results.

Definition of Terms

Machine learning. A group of artificial intelligence that can learn from data (Alpaydin, 2004).

DNA. Deoxyribonucleic acid (Watson, & Crick, 1953).

Protein. Biological molecules consisting of one or more chains of amino acidresidues (Kauzma, 1956).

PyBrain. Python-Based Reinforcement Learning Artificial Intelligence (Schaul et al., 2010).

Scilkit–learn. Scilkit–learn: Machine learning in Python (Pedregosa et al., 2011).

SVM. Support Vector Machines (Hearst et al., 1998).

SGD. Stochastic Gradient Descent (Bottou, 2010).

Backprop. Back Propagation (Rumelhart Hinton, & Williams, 1988).

Rprop. Resilient Propagation (Riedmiller, & Braun, 1992).

Limitations and Delimitations

This study collected data from public genome repositories. The machines were able to identify and predict protein complexes that only existed in the databases. The size of the data which were used in this study depended on available equipment because the run time of training the machine depended on the size of data and capability of equipment. The result of the thesis focused on the accuracy of the machines after performing testing with testing data sets.

Assumptions

The fundamental assumption in the use of the machine was to classify the protein complexes from a DNA sequence input to the protein complexes that already existed in the trained database.

Chapter 2

REVIEW OF LITERATURE

There are several algorithms that use training methods for machine learning. The first algorithm is Support Vector Machines that analyzes data and recognizes patterns, and it is used for classification and regression analysis (Cortes, & Vapnik, 1995). The second algorithm is Stochastic Gradient Descent that is a gradient descent optimization method for minimizing an objective function. (Bottou, 2010). The third algorithm is Backpropagation or Backward Propagation of Errors. This algorithm is a method for training neural networks used with an optimization method such as gradient descent (Rumelhart Hinton, & Williams, 1988). The fourth algorithm is Resilient Propagation or Rprop. This algorithm is a first-order optimization algorithm and one of the supervised methods for learning in neural networks (Riedmiller, & Braun, 1992).

This chapter will discuss research activities in each of the above four algorithms and also provide a summary of the results from related researches.

Support Vector Machines

From the study “Identification of DNA-Binding Protein Using Support Vector Machine with Sequence Information,” the identification of DNA-binding has important practical application in various fields. The researchers have proposed an approach method for predicting DNA-binding protein, using sequence information, by using support vector machines in conjunction with a hybrid feature. There are two attributes used in DNA-binding residues and nonbinding in a query protein to obtain DNA-binding propensity and nonbinding propensity. The results show that the SVM method has 0.67

Matthew's Correlation Coefficient (MCC) and 89.6% overall accuracy with 88.4% sensitivity and 90.8% specificity. The SVM method also has the best performance on the independent test dataset (Ma, Wu, & Xue, 2013).

Stochastic Gradient Descent

From the study "Digital Image Correlation Using Stochastic Parallel-Gradient-Descent Algorithm," the stochastic perturbations are imposed on deformation parameters to create the correlation converge to a global extremum. This method allows the final measured values of the deformation parameters to be obtained and the DIC measurement to be created. Simulated and real data processing show that this method can achieve the same accuracy as the Newton Raphson (NR) method in most cases. It has good noise robustness. A series of experiments are conducted to evaluate the convergence characteristics of this method, and it shows that this method is able to process large displacement and has a stable convergence process, good robustness, and a high convergence speed. (Long et al., 2012)

Backward Propagation of Errors

From the study "HYBP_PSSP: a hybrid back propagation method for predicting protein secondary structure," the secondary protein structure prediction is one of the important topics in the field of bioinformatics. Accurate prediction is an essential component of tertiary structure prediction. Sequence comparison methods have incorporated local structure tracks. Using evolutionary information contained in amino acid's physicochemical properties is inputted into the hybrid back propagation system for analysis. The secondary can be predicted at significantly increased accuracy. The compound pyramid model is composed of four layers of the intelligent interface and integrated in several ways. Experiments on three standard datasets present that this model is capable of producing higher scores than existing widely used schemes, and it also shows better results than the traditional methods when tested on target proteins of critical assessment of protein structure prediction experiment (Qu, et al, 2012).

Resilient Propagation

The study "Learning of geometric mean neuron model using resilient propagation algorithm," introduces a new geometric mean neuron model with a collection function based on the geometric mean of the inputs. The models are trained by various learning algorithms like the back-propagation and resilient propagation. Then, their performance is evaluated by the geometric mean neuron model. This study compares the performance of the geometric mean neuron model and the performance of multilayer perceptron. The result of the study shows that the geometric mean based aggregation function has the approximation capability. Comparative performance evaluation on real data sets pertaining to prediction and classification problems indicate the GMM model using resilient propagation performs better than the multilayer perceptron (MLP) (Shiblee, Chandra, & Kalra, 2010)

Chapter 3

METHOD OF PROCEDURE

The overarching goal of this project was to build the machine learning models that could aid in the classification of protein complexes, from DNA sequence information using supervised learning algorithms, from PyBrain and Scikit-learn libraries, on Python sell. To accomplish this goal, the machine models were created with two different structures. From Scikit-learn library, the machine models were built with typical machine learning structures and trained with Support Vector Machines and Stochastic Gradient Descent algorithms. From PyBrain library, the neural network structures were used as machine learning models. The models were trained with Back propagation and Resilient Propagation algorithms. All of the machines were trained by the same training data sets, from the six types of protein complex data pool, and they were also tested by the same testing data set, from the same protein complex data pool. The result from the machine model showed six predicted possibilities of the sample DNA sequence data, based on six types of protein complexes. The model results were calculated and visualized to show the performance and accuracy, based on the protein complex types and supervised learning algorithms.

Scikit-learn: Machine Learning in Python

This machine learning Python library was started in 2007 as a Google summer of Code project, by David Cournapeau. In 2010, the French Institute for Research in Computer Science and Automation (INRIA) took leadership of the project and made the first public release, in February, 2010. Several updates have updated following a 3-month cycle, and a thriving international community has been leading the development. Scikit-learn library has simple and efficient tools for data mining and data analysis. Everybody can access the library (open source – Berkeley Software Distribution license) and reuse in various contexts. The library is built on several Python libraries that are Numpy, Scipy, and matplotlib (Pedregosa et al., 2011). From Scikit-learn library, the machine learning model structure and two supervised learning algorithms were used in this study. The two algorithms involved in the process of this thesis. They are shown below.

Support Vector Machines (SVM)

Support Vector Machine (SVM) comes into play when it becomes necessary to have a classification process, which involved multi-class. SVM became more popular because capability of pattern recognition properties and improved computing power that becomes powerful enough to train the machines with SVM adequately. SVM is used for classification, regression, and outlier detection. The advantages of SVM are the algorithms, which are useful in high dimensional spaces and are still accurate in cases when the number of dimensions is greater than the number of samples. SVM is memory efficient because it uses a subset of training points, in the decision function. The disadvantages of SVM, the algorithm, will give poor performances when the number of features are much greater than the number of samples. SVM, from this library, does not directly provide probability estimates. The probability estimates are calculated using an expensive five-fold cross-validation. The Support Vector Machine in Scikit-learn supports dense and sparse sample input (Pedregosa et al., 2011).

A Support Vector Classification (SVC) builds a hyper-plane in a high-dimensional space, which can be used for classification. A good separation is achieved by the hyper-plane that has the largest distance to nearest training data points of any class.

The larger of the margin will make the generalization error of the classification lower. The SVC implements the "One-against-One" approach for multi-classification (Pedregosa et al., 2011). The definition of this scheme is as below.

"One-against-One strategy is also known as pairwise coupling. If N is the number of classes, then "N * (N – 1) / 2" classifiers are constructed and each one trains data from two types." (Knerr et al., 1990, p.44).

Stochastic Gradient Descent (SGD)

The Stochastic Gradient Descent (SGD) is an efficient algorithm to learn the linear classifiers, under convex loss functions. This algorithm has been in the machine learning commodity for a long time, but it has received attention in the context of large-scale learning. SGD is successfully implemented to large-scale and sparse machine learning problems, such as text classification and natural language processing. The classifiers that are trained by SGD can scale to the problem with more than 10^5 training examples and more than 10^5 features. The advantage of SGD is an algorithm, which is very efficient and easy for implementation. The disadvantage of SGD is the algorithm, which requires a number of hyperparameters and a number of iterations, and it is very sensitive to feature scaling (Pedregosa et al, 2011).

Stochastic Gradient Descent Classification is the SGD that implements a plain SGD learning routine that supports different loss functions and penalties for classification. The SGD Classification supports multi-class classification, by combining multiple binary classifiers in a "one-against-all" scheme (Pedregosa et al, 2011). The definition of this scheme is as below.

"For each of the K classes, a binary classifier is learned that discriminates between that and all other $K-1$ classes. At testing time, we compute the confidence score for each classifier and choose the class with the highest confidence." (Knerr et al., 1990, p. 45).

Python-Based Reinforcement Learning, Artificial Intelligence and Neural Network Library (PyBrain)

PyBrain is a modular machine learning library for Python. This library provides easy to use and robust algorithms. The modular has a variety of predefined environments to test and compare algorithms. PyBrain can be used by entry-level students because it offers flexibility and algorithms for research. PyBrain contains several algorithms for many approaches, such as neural networks, reinforcement learning, unsupervised learning, supervised learning, and evolution. This library is built around neural networks in the kernel, and all of the training algorithms accept a neural network as the trained instance. PyBrain is the open source under the Berkeley Software Distribution license. PyBrain focuses on network architectures, which can be trained and manipulated with the algorithms that offer in this library, such as feed-forward networks, recurrent-networks (RRN), and Multi-Dimensional Recurrent Networks (MDRNN). The next part briefly gives the information on a neural network topic (Schaul et al., 2010).

A neural network is an information-processing system that has performance in common with biological neural networks. The system develops as generalization of math mathematical models of human cognition. The network is classified based on the pattern of connections between the neurons, method of determining the weights in the network, and its activation function. A neural net consists of a number of nodes. Each node is connected to other nodes by means of directed communication links (weight). The weight indicates that the information is being used to solve the problem. Each node had an internal state called activation that is a function of the inputs it has received. A simple neural network consists of X node, Y node, and Z node. X is an input node. Z is an output node. Y is a hidden node. With the presence of the hidden unit, it gives the ability to solve more problems that cannot be solved by the system that has just only input units and output units. On the other hand, it is more difficult to train the network. This study used two types of the neural networks. The first neural network is feed-forward neural network. This network allows signals to travel one way only from input to output. The outputs of each layer do not affect that same layer where they are calculated. The second neural network is a recurrent neural network. This network allows signals traveling in both directions, by implementing loops to the network. Recurrent neural networks are powerful and dynamic, because states of the nodes are changing continuously, until they reach an equilibrium point. The equilibrium points remain the same till the network receives the new input, and new point needs to be found (Fausett, 1994). From PyBrain library, the machine learning model structure (the neural networks) and two supervised learning algorithms were used in this study. The two algorithms involved in the process of this thesis are as below.

Back Propagation (Backprop)

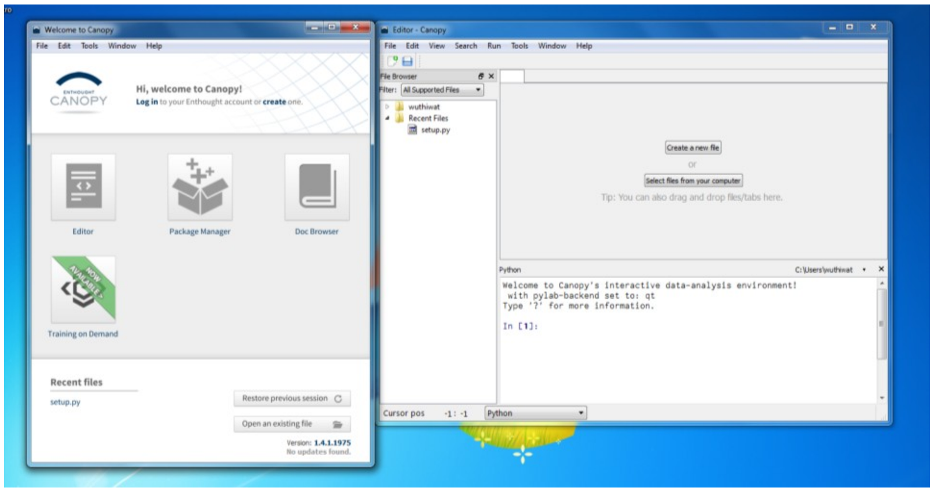
The back propagation training algorithm is a back propagation net that is a feed forward neural network trained by back propagation. This algorithm can be used to solve the problem in many cases. With a neural network, Backprop algorithm aims to train the network to achieve a balance between the ability to respond correctly to the input patterns that are used as a training set, and the ability to give the right reasonable for the input (a testing set). There are three steps that involve while the net is trained by the Backprop algorithm. The first step is the feed-forward of the input training pattern. The second step is the calculation and Back propagation of the associated error. The last step is the adjustment of the weights. In the testing phase, the application of the network involves only the computations of the feed-forward phase. Even if the training process is slow, the network can calculate and predict the output very rapidly. More than one hidden layer may be beneficial, but one hidden layer is sufficient (Fausett, 1994).

Resilient Propagation (Rporp)

Resilient propagation algorithm is a learning heuristic for supervised learning in a neural network that batches out from the Back propagation algorithm. Rporp is based on the Manhattan Learning rule. With this rule, the error function of the weight update function can be more appropriately adjusted. Each weight has its personal update value which evolves during the learning process. The update value is not influenced by the magnitude of the derivatives, but by the behavior of the sign of two succeeding derivatives. The main idea of this algorithm is the size of the step used to update the weight is not determined by the gradient. This algorithm was created by Martin Riedmiller and Heinrich Braun in 1992 (Riedmiller, & Braun, 1992).

Programming Language Selection

The proposed approach to develop learning machines was to use Python as the software language. Python is high-level programming language and is widely used. This language emphasizes a readable code, and its syntax allows the users to write concepts in fewer lines of code. Python provides constructs on both a small and large scale. The language supports multiple programming paradigms, such as object-oriented, imperative, and procedural or functional styles. Python also features an automatic memory management and a dynamic type system. This language has a large and comprehensive standard library. Python can be installed on many operating systems, which means that Python can also be executed on a majority of the systems. Python codes can be packaged into executable programs for almost every modern operating system. Python is a free and open-source software. Python language was created by Guido van Rossum, in 1980. Python is designed to use English keywords where other languages use punctuation. Python uses whitespace indention to delimit blocks. This feature is termed the off-side rule. Python implementations can function as a command line interpreter that means the user can enter statement codes and receive the results immediately, because Python acts as a shell. The Python shell implements many features, such as auto-completion, syntax highlighting, and retention of session state. The latest version of Python is Python 3.4.1 (Van, 1993). This study used Python package from Enthought Scientific Computing Solution. The package name is “Enthought Canopy”. This package is a scientific and analytic Python deployment with integrated analysis environment. The package has easy installation and creates a robust platform that user can explore, develop, and visualize. This Python package has valuable tools for visualization, application development, and iterative data analysis, such as code editor with IPython notebook support and the integrated IPython prompt (Enthought Canopy, 2014). The figure below is Enthought Canopy user interface.



```
Welcome to Canopy's interactive data-analysis environment!
 with pylab-backend set to: qt
Type '?' for more information.

In [1]:
```

Figure 1. Enthought canopy's user interface.

Selection of Data

Protein complexes compose of two or more polypeptide subunits and attach with non-covalent protein-protein interactions. Protein complexes play an important role in biological systems in living things. All biological systems always have protein complexes as parts of the system such as an initial substance, a product substance, an indicator, and buffer. The goal of this study was to build learning machines that could classify and predict protein complexes. For this purpose, the machines needed data sets to train them. Six protein complexes from three biological systems were selected. This study chosen to pick protein complexes from different biological systems in stand of focused on only one system, because the researcher wanted the diversity of protein complexes in learning machines. The three biological systems were chemical digestion systems, cell material transportation systems, and energy production systems. From each system, two protein complexes were picked.

The Chemical Digestion Systems

The digestion systems are where the food is broken down into smaller components that can be absorbed and assimilated by the body. Digestions are divided into two processes based on how the food is broken down; there is mechanical digestion and chemical digestion. In chemical digestion, enzymes digest the food into the small molecules the body can use. The enzymes that work in this biological system are protein complexes. Two proteins were selected from the chemical digestion system list, as below.

a. **Amylase.**

   Amylase is an enzyme that digests the starch into sugars. It presents in human saliva and some other mammals. The salivary gland and pancreas create amylase. Some bacteria and plants also produce amylase (Hill, & Needham, 1970).

b. **Dipeptidase**

   Dipeptidase is an enzyme that is encoded by the DPDEP1 gene (in humans). The kidney membranes produce this enzyme. Dipeptidase hydrolyzes a variety of dipeptides and is implicated in renal metabolism of glutathione. Earlier, this protein is thought to occur only in bacteria, but it can be found in human and some other mammals (Eisenach et al., 2013).

Cell Material Transportation Systems

There are many biological systems that relate to cell material transport. This study focused on two systems. The first system was photophosphorylation. It is in the process of photosynthesis. There are two sources of available energy for living organisms, which are sunlight and reduction-oxidation reactions. Photophosphorylation transforms adenosine diphosphate (ADP) to adenosine triphosphate (ATP), by absorption the energy from sunlight and transports the energy in the form of ATP to other organisms. The second system was the oxygen transport in humans and animals. The blood carries oxygen from lungs or gills, to the rest of the body, where it releases the oxygen to use in cell metabolisms. Two proteins are selected from photophosphorylation and oxygen transportations list as below (Berg, 2007).

a. **Adenosine Triphosphate Synthase Gamma (F1 – ATP)**

   ATP synthesis is the enzyme that carries energy to the cell for using through the cell activities. ATP is commonly used as an energy carrier for the most organisms. Energy is contained in hydrogen ions (H^+) when ADP is transferred to ATP, and the energy is released to the organisms when ATP is changed back to ADP. ATP consists of two subunits. The first subunit is $F_{0,}$ which is in the membrane of the mitochondria. The second subunit is $F_{1,}$ which is above the membrane but still inside of the mitochondria's matrix. The F_1 can be seen in the transmission electron microscope. The F_1 forms the central shaft that connects to the F_0. The human F_1-ATP subunit is encoded by the gene ATP5C1 (Mccarty, 1992).

b. **Hemoglobin**

   The hemoglobin is the oxygen transport metalloproteinase in all vertebrates and some invertebrates. This protein complex can found in the red blood cells. In the mammal red blood cells, they consist of proteins by 96%. Hemoglobin can transfer 1.34 ml of oxygen per a gram. The protein is also involved in other gas transportation, like carbon dioxide and nitric oxide. The hemoglobin can be found outside of red blood cells. Some other cells that have this protein complex are substantial nigra, alveolar cells, mesangial cells in the kidney, and macrophages. This protein consists of many protein subunits and is folded chains with a large number of different amino acids. The hemoglobin exists in two forms that are a tense form and a relaxed form. This protein has a quaternary structure (Perutz, 1965).

Energy Production Systems

One of the critical energy production systems in the plant and other organisms is photosynthesis. This system converts light from the sun into chemical energy. Later, this chemical energy can be released to fuel the organism activities. Photosynthesis is worked differently by different species. In bacteria, this system occurs in cell membranes that tightly fold into cylindrical sheets called thylakoids. In plants, this system happens in organelles called chloroplasts. A plant cell can contain about 10 to 100 chloroplasts. In the photosynthesis, there are two minor systems that are photosystem I and photosystem II. Photosystem I is an integral membrane protein complex. Photosystem II is located in the thylakoid membrane of plants, algae, and cyanobacteria, and it is the first protein complex in the photosynthesis that was found (Raven, & Evert, 1999). Two proteins were selected from the energy production systems list as below.

a. **Photosynthetic Reaction Centre Protein**

The photosynthetic reaction centre protein can be found in bacteria and plants. This protein is a principal component of photosynthetic reaction centre. In bacteria, this protein consists of light-harvesting protein-pigment complexes I (LH1) and II (LH2). This protein complex uses carotenoid and bacterio-chlorophyll as the donors. LH1 is an energy hub that receives electrons from the donors and temporarily stores the electrons, before they are passed, to the next cells. In the plant, this protein is related to photosystem II (Scheuring, 2006).

b. Plastocyanin

The plastocyanin is in electron-transfer systems and a copper-containing protein. This protein functions as an electron transfer in photosystem II. This protein is the first blue copper protein that was identified by X-ray crystallography. The molecular surface of this protein is differed based on organisms such as plants, algae, and cyanobacteria (Gewirth et al., 1988).

Collection and Preparation of Data

Classification of Data

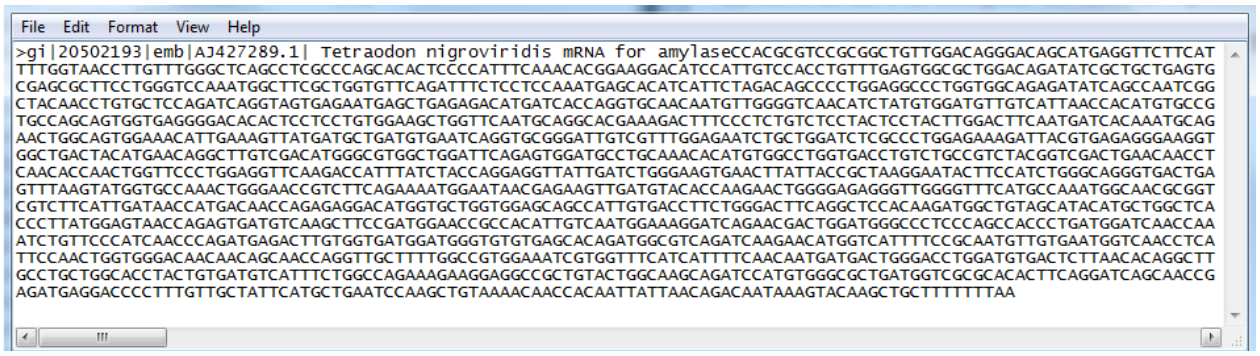
This study selected six protein complexes for training and testing learning machines. The protein complexes were classified into six types of protein complex targets, for an easy data management. The classification of the protein complexes lists as below.

- Type 1: Amylase.
- Type 2: Dipeptidase.
- Type 3: Adenosine triphosphate synthase gamma (F1 – ATP).
- Type 4: Hemoglobin.
- Type 5: Photosynthetic reaction centre protein.
- Type 6: Plastocyanin.

Collection of Data

All of the protein complexes data was downloaded from the National Center for Biotechnology Information (NCBI) database. The protein data that came from NCBI database was in the FASTA file format.

The FASTA format is a text based format in bioinformatics. It presents peptide sequences or nucleotide sequences, in which amino acids or nucleotides are represented as single-letter codes. In this form, sequence names and comments can be included into the data. The FASTA format makes it easy to manage and input the sequences using scripting languages, like Python and Perl. The FASTA format starts with a single-line description, and then follows by lines of sequence data. The description is separated from the sequence data by a greater-than symbol (“>”). The word that follows the greater-than symbol is the identifier of the sequence. The rest is the description. The end of the sequence is identified by appearing of the greater-than symbol that is a starter of the next data sequence. The figure below is the example of the data in the FASTA format.



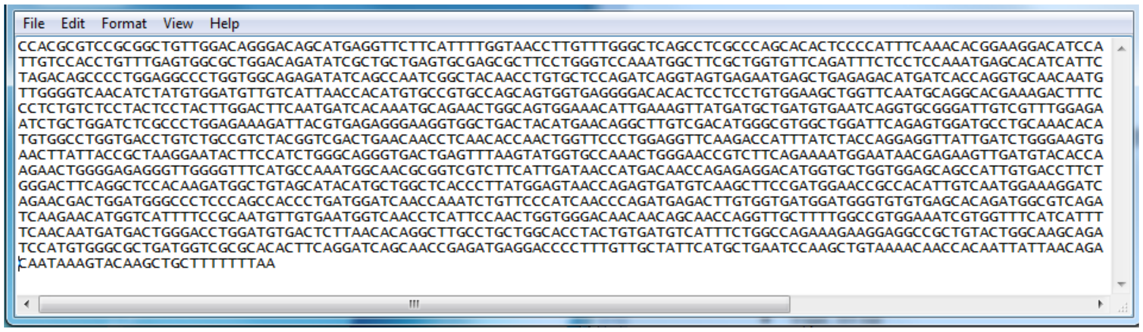
```
>gi|20502193|emb|AJ427289.1| Tetraodon nigroviridis mRNA for amylaseCCACGCGTCCGCGGCTGTTGGACAGGGACAGCATGAGGTTCTTCAT
TTTGGTAACCTTGTTTGGGCTCAGCCTCGCCCAGCACACTCCCCATTTCAAACACGGAAGGACATCCATTGTCCACCTGTTTGAGTGGCGCTGGACAGATATCGCTGCTGAGTG
CGAGCGCTTCCTGGGTCCAAATGGCTTCGCTGGTGTTCAGATTTCTCCTCCAAATGAGCACATCATTCTAGACAGCCCCTGGAGGCCCTGGTGGCAGAGATATCAGCCAATCGG
CTACAACCTGTGCTCCAGATCAGGTAGTGAGAATGAGCTGAGAGACATGATCACCAGGTGCAACAATGTTGGGGTCAACATCTATGTGGATGTTGTCATTAACCACATGTGCCG
TGCCAGCAGTGGTGAGGGGACACACTCCTCCTGTGGAAGCTGGTTCAATGCAGGCACGAAAGACTTTCCCTCTGTCTCCTACTCCTACTTGGACTTCAATGATCACAAATGCAG
AACTGGCAGTGGAAACATTGAAAGTTATGATGCTGATGTGAATCAGGTGCGGGATTGTCGTTTGGAGAATCTGCTGGATCTCGCCCTGGAGAAAGATTACGTGAGAGGGAAGGT
GGCTGACTACATGAACAGGCTTGTCGACATGGGCGTGGCTGGATTCAGAGTGGATGCCTGCAAACACATGTGGCCTGGTGACCTGTCTGCCGTCTACGGTCGACTGAACAACCT
CAACACCAACTGGTTCCCTGGAGGTTCAAGACCATTTATCTACCAGGAGGTTATTGATCTGGGAAGTGAACTTATTACCGCTAAGGAATACTTCCATCTGGGCAGGGTGACTGA
GTTTAAGTATGGTGCCAAACTGGGAACCGTCTTCAGAAAATGGAATAACGAGAAGTTGATGTACACCAAGAACTGGGGAGAGGGTTGGGGTTTCATGCCAAATGGCAACGCGGT
CGTCTTCATTGATAACCATGACAACCAGAGAGGACATGGTGCTGGTGGAGCAGCCATTGTGACCTTCTGGGACTTCAGGCTCCACAAGATGGCTGTAGCATACATGCTGGCTCA
CCCTTATGGAGTAACCAGAGTGATGTCAAGCTTCCGATGGAACCGCCACATTGTCAATGGAAAGGATCAGAACGACTGGATGGGCCCTCCCAGCCACCCTGATGGATCAACCAA
ATCTGTTCCCATCAACCCAGATGAGACTTGTGGTGATGGATGGGTGTGTGAGCACAGATGGCGTCAGATCAAGAACATGGTCATTTTCCGCAATGTTGTGAATGGTCAACCTCA
TTCCAACTGGTGGGACAACAACAGCAACCAGGTTGCTTTTGGCCGTGGAAATCGTGGTTTCATCATTTTCAACAATGATGACTGGGACCTGGATGTGACTCTTAACACAGGCTT
GCCTGCTGGCACCTACTGTGATGTCATTTCTGGCCAGAAAGAAGGAGGCCGCTGTACTGGCAAGCAGATCCATGTGGGCGCTGATGGTCGCGCACACTTCAGGATCAGCAACCG
AGATGAGGACCCCTTTGTTGCTATTCATGCTGAATCCAAGCTGTAAAACAACCACAATTATTAACAGACAATAAAGTACAAGCTGCTTTTTTTAA
```

Figure 2. The raw data in the FASTA format.

Preparation of Data

From the raw FASTA data, this study used pyfasta 0.5.2 library for obtained the DNA sequences, from the FASTA file format. The pyfasts 0.5.2 was created by Brent Pedersen. It is an open source library. After the DNA sequences were extracted from the raw data sets, the DNA sequences were separated into the categorized files based on the types of protein complexes. At the end of this process, six text files were created. The figure 3 is the example of the extracted data. The list of the text files is below.

- Type_01.txt
- Type_02.txt
- Type_03.txt
- Type_04.txt
- Type_05.txt
- Type_06.txt



```
CCACGCGTCCGCGGCTGTTGGACAGGGACAGCATGAGGTTCTTCATTTTGGTAACCTTGTTTGGGCTCAGCCTCGCCCAGCACACTCCCCATTTCAAACACGGAAGGACATCCA
TTGTCCACCTGTTTGAGTGGCGCTGGACAGATATCGCTGCTGAGTGCGAGCGCTTCCTGGGTCCAAATGGCTTCGCTGGTGTTCAGATTTCTCCTCCAAATGAGCACATCATTC
TAGACAGCCCCTGGAGGCCCTGGTGGCAGAGATATCAGCCAATCGGCTACAACCTGTGCTCCAGATCAGGTAGTGAGAATGAGCTGAGAGACATGATCACCAGGTGCAACAATG
TTGGGGTCAACATCTATGTGGATGTTGTCATTAACCACATGTGCCGTGCCAGCAGTGGTGAGGGGACACACTCCTCCTGTGGAAGCTGGTTCAATGCAGGCACGAAAGACTTTC
CCTCTGTCTCCTACTCCTACTTGGACTTCAATGATCACAAATGCAGAACTGGCAGTGGAAACATTGAAAGTTATGATGCTGATGTGAATCAGGTGCGGGATTGTCGTTTGGAGA
ATCTGCTGGATCTCGCCCTGGAGAAAGATTACGTGAGAGGGAAGGTGGCTGACTACATGAACAGGCTTGTCGACATGGGCGTGGCTGGATTCAGAGTGGATGCCTGCAAACACA
TGTGGCCTGGTGACCTGTCTGCCGTCTACGGTCGACTGAACAACCTCAACACCAACTGGTTCCCTGGAGGTTCAAGACCATTTATCTACCAGGAGGTTATTGATCTGGGAAGTG
AACTTATTACCGCTAAGGAATACTTCCATCTGGGCAGGGTGACTGAGTTTAAGTATGGTGCCAAACTGGGAACCGTCTTCAGAAAATGGAATAACGAGAAGTTGATGTACACCA
AGAACTGGGGAGAGGGTTGGGGTTTCATGCCAAATGGCAACGCGGTCGTCTTCATTGATAACCATGACAACCAGAGAGGACATGGTGCTGGTGGAGCAGCCATTGTGACCTTCT
GGGACTTCAGGCTCCACAAGATGGCTGTAGCATACATGCTGGCTCACCCTTATGGAGTAACCAGAGTGATGTCAAGCTTCCGATGGAACCGCCACATTGTCAATGGAAAGGATC
AGAACGACTGGATGGGCCCTCCCAGCCACCCTGATGGATCAACCAAATCTGTTCCCATCAACCCAGATGAGACTTGTGGTGATGGATGGGTGTGTGAGCACAGATGGCGTCAGA
TCAAGAACATGGTCATTTTCCGCAATGTTGTGAATGGTCAACCTCATTCCAACTGGTGGGACAACAACAGCAACCAGGTTGCTTTTGGCCGTGGAAATCGTGGTTTCATCATTT
TCAACAATGATGACTGGGACCTGGATGTGACTCTTAACACAGGCTTGCCTGCTGGCACCTACTGTGATGTCATTTCTGGCCAGAAAGAAGGAGGCCGCTGTACTGGCAAGCAGA
TCCATGTGGGCGCTGATGGTCGCGCACACTTCAGGATCAGCAACCGAGATGAGGACCCCTTTGTTGCTATTCATGCTGAATCCAAGCTGTAAAACAACCACAATTATTAACAGA
CAATAAAGTACAAGCTGCTTTTTTTAA
```

Figure 3. The data after preparation.

Creating of Training and Testing Datasets

The learning machines needed training data sets to train the machines for classification of protein complexes, and by evaluated accuracy of the machines, the machines needed to be tested with testing datasets. Each learning data set was consisted of 1000 DNA sequences that randomly chosen from the protein complex data pools, based on types of the protein. Each testing data set was consisted of 400 DNA sequences that randomly picked from the protein complex data pools, based on types of the protein. The DNA sequences in testing data sets were not the same sequences that already appear in the training data sets. At the end of this process, twelve text files were created. The lists of the files are below.

a. Training data list

- Type_01_train.txt (1000 samples)
- Type_02_train.txt (1000 samples)
- Type_03_train.txt (1000 samples)
- Type_04_train.txt (1000 samples)
- Type_05_train.txt (1000 samples)
- Type_06_train.txt (1000 samples)

b. Testing data list

- Type_01_test.txt (400 samples)
- Type_02_test.txt (400 samples)
- Type_03_test.txt (400 samples)
- Type_04_test.txt (400 samples)
- Type_05_test.txt (400 samples)
- Type_06_test.txt (400 samples)

Creation of Machine Structures

Building of the Learning Machines

Two libraries were used for building learning machines, in this study. On the Scikit-learn library, the machines were made by directly using supervised learning model as the learning machines. On the PyBrain library, the machines were built by using neural network as the cores of the machines, and the machines were trained by supervised learning algorithms.

The Learning Machines from Scikit-learn Library

a. A support vector classification

The machine from a support vector classification was built by using the follow parameters that are listed below.

SVC(C=1.0, kernel='rbf', degree=3, gamma=0.0, coef0=0.0, shrinking=True,probability=True, tol=0.001, cache_size=200, class_weight=None, verbose=False, max_iter=-1, random_state=None)

- C: the parameter of the error term – 1
- Kernel: the kernel types – radial basis function kernel (rbf)
- Degree: the degree of the polynomial kernel function – 3
- Gamma: the kernel coefficient for the kernel – 0.0
- Coef0: the independent term in kernel function – 0.0
- Shrinking: using the shrinking heuristic – True
- Probability: using to enable probability estimates – True
- Tol: the tolerance for stopping criterion – 0.001
- Cache_size: specify the size of the kernel cache – 200
- Class_weight: set the parameter C of Class i to class_weight[i]*C of SVC – None
- Verbose: enable verbose output – False
- Max_iter: Hard limit on iterations within solver – - 1 (no limit)

- Random_state: the seed of the pseudo-random number generator to use when shuffling the data for pro probability – None

b. A stochastic gradient descent

The machine from a stochastic gradient descent classification was built by using the follow parameters that are listed below.

SGDClassifier(loss='log', penalty='l2', alpha=0.0001, l1_ratio=0.15,fit_intercept=True, n_iter=5, shuffle=True, verbose=0, epsilon=0.1, n_jobs=1, random_state=None,learning_rate='optimal', eta0=0.0, power_t=0.5, class_weight=None)

- Loss: set the loss function – the logistic regression (log) with a probabilistic classifier
- Penalty: set the penalty – the standard regularizer (l2)
- Alpha: constant that multiplies the regularization term – 0.0001
- L1_ratio: the elastic net mixing parameter – 0.15
- Fit_intercept: set the estimated intercept – True
- N_iter: the number of passes over the training data (epochs) – 5
- Verbose: enable verbose output – False
- Epsilon: the epsilon in the epsilon-insensitive loss function – 0.1
- Shuffle: set data shuffling – True
- Random_state: the seed of the pseudo-random number generator to use when shuffling the data for pro probability – None
- Learning_rate: the learning rate – optimal
- Eta0: the initial learning rate – 0.0

- Power_t: the exponent for inverse scaling learning rate – 0.5
- Class_weight: pre-set for the class weight fit parameter – None

The Learning Machines from PyBrain Library

a. A neural network for Back propagation

The neural network for back propagation algorithm was built by using the follow parameters that are shown below.

buildNetwork(500, 500, 6, hiddenclass=TanhLayer, outclass=SoftmaxLayer)

- Input nodes – 500
- Hidden node – 500
- Output nodes – 6
- Hiddenclass: the type of hidden layer – TanhLayer: a layer that implementing the tanh squashing function
- Outclass: The type of output later – SoftmaxLayer: a layer implementing a softmax distribution
- The type of the network is a feed-forward network

b. A neural network for Resilient propagation

The neural network for resilient propagation algorithm was built by using the follow parameters that are presented below.

BuildNetwork(500, 500, 6, hiddenclass=LSTMLayer, outclass=SoftmaxLayer, recurrent=True)

- Input nodes – 500
- Hidden node – 500
- Output nodes – 6

- Hiddenclass: the type of hidden layer – LSTMLayer: Long short-term memory cell layer
- Outclass: the type of output later – SoftmaxLayer: a layer implementing a softmax distribution
- Recurrent: setup the recurrent neural network – True

Training of the Learning Machines

All of the learning machines needed to be trained before they could perform protein complex classification. The training datasets for the machines were the same datasets that came from six types of protein complexes. The training methods were different, based on the machine learning library. After the machines finished the training processes, the machines were saved in the XML file format, based on the supervised learning algorithms. At the end of this process, four XML files were created. The list of the files is listed below.

- SVM_network.XML
- SGD_network.XML
- Backprop_network.XML
- Rprop_network.XML

The Machine Training for Scikit-learn Library

In this library, the method that was used to train the machines for support vector classification algorithm and stochastic gradient descent algorithm was the same. The machines were trained by the parameters that were shown below.

Fit[n_samples, n_label]

- N_samples: the sequences of the training samples data – DNA sequences

- N_label: the sequences of the sample labels – the protein complexes types (1, 2, 3, 4, 5, 6)

The Machine Training for Bybrain Library

In the Bybrain library, the trainer for the neural networks were created, based on the supervised learning algorithms, and the parameters for the trainer were deferent, based on the learning algorithms.

a. **A back propagation trainer**

   BankpropTrainer(module, dataset, learningrate=0.01, lrdecay=1.0, momentum=0.1, verbose=True, batchlearning=False, weightdeca*y=0.001)*

   - Module: the neural network
   - Dataset: training dataset
   - Learningrate: the ratio of the changed direction in the gradient – 0.01
   - Lrdecay: the learning rate decreases – 1.0
   - Momentum: the adjusted rate for the learning – 0.1
   - Verbose: set to provide console output while learning – True
   - Batchlearning: set parameters update at the end of each epoch – False
   - Weightdecay: corresponds of the weight decay – 0.001

b. **A Resilient propagation trainer**

   RPropMinusTrainer*(*module,

   dataset, etaminus=0.5, etaplus=1.2,deltamin=9.995e-

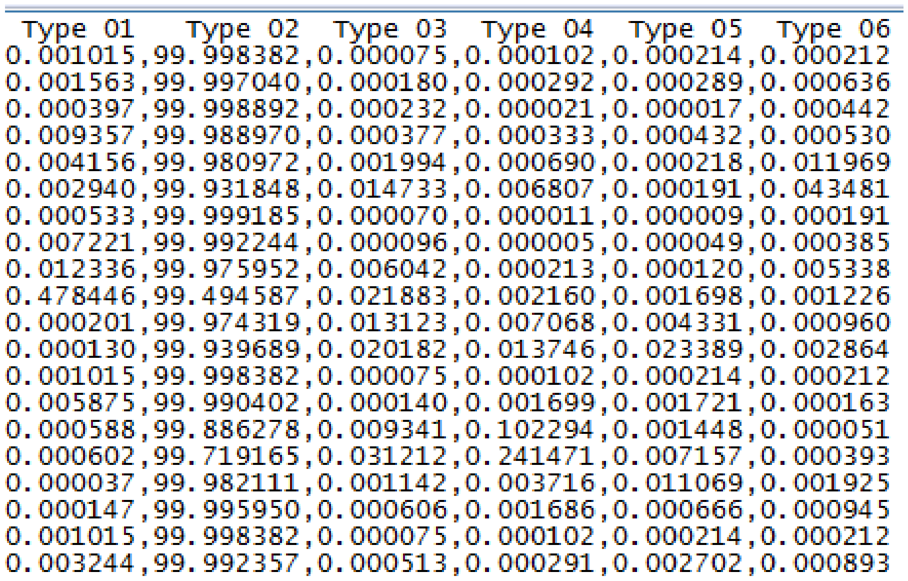
   07, deltamax=5.0, delta0=0.1)

   - Module: the neural network

- Dataset: training dataset
- Etaminus: factor by which step width is decreased when overstepping – 0.5
- Etaplus: factor by which step width is increased when following gradient – 1.2
- Deltamin: minimum step width – 9.995e-07
- Delamax: maximum step width – 5.0
- Delta0: initial step width – 0.1

Testing of the Learning Machines

The learning machines were tested by performing classification of protein complexes by using the testing datasets as the input for the machines. The testing datasets only had the DNA sequence data, but they did not have the types of the protein complexes. The outputs of the machines after preformed classification were the predicted possibilities. The testing datasets were inputted into the machines, based on the types of the proteins. The predicted probability results were recorded on the text file, based on the types of protein and types of supervised learning algorithms. The predicted probability result for each protein classification was in the form of an array that consists of six values. The six values in the array could add up into one hundred percent, which means the predicted possibility result was the percentage of the protein classification. Figure 4 is an example of the output from the learning machines.



```
  Type 01    Type 02   Type 03   Type 04   Type 05   Type 06
0.001015,99.998382,0.000075,0.000102,0.000214,0.000212
0.001563,99.997040,0.000180,0.000292,0.000289,0.000636
0.000397,99.998892,0.000232,0.000021,0.000017,0.000442
0.009357,99.988970,0.000377,0.000333,0.000432,0.000530
0.004156,99.980972,0.001994,0.000690,0.000218,0.011969
0.002940,99.931848,0.014733,0.006807,0.000191,0.043481
0.000533,99.999185,0.000070,0.000011,0.000009,0.000191
0.007221,99.992244,0.000096,0.000005,0.000049,0.000385
0.012336,99.975952,0.006042,0.000213,0.000120,0.005338
0.478446,99.494587,0.021883,0.002160,0.001698,0.001226
0.000201,99.974319,0.013123,0.007068,0.004331,0.000960
0.000130,99.939689,0.020182,0.013746,0.023389,0.002864
0.001015,99.998382,0.000075,0.000102,0.000214,0.000212
0.005875,99.990402,0.000140,0.001699,0.001721,0.000163
0.000588,99.886278,0.009341,0.102294,0.001448,0.000051
0.000602,99.719165,0.031212,0.241471,0.007157,0.000393
0.000037,99.982111,0.001142,0.003716,0.011069,0.001925
0.000147,99.995950,0.000606,0.001686,0.000666,0.000945
0.001015,99.998382,0.000075,0.000102,0.000214,0.000212
0.003244,99.992357,0.000513,0.000291,0.002702,0.000893
```

Figure 4. The machine's outputs.

Visualization of the Results

The predicted results from the machines were visualized by using two types of the visualization methods. The first method was a line chart. The line chart is the chart that displays information as a series of markers that are connected by straight line. This chart shows how a particular data changes at intervals of the time or units (Andreas, 1960). The second method was a clustered column chart. This chart is useful to compare data points in one more data series. The chart illustrates comparisons among items and categories for this chart's organization, along the horizontal axis and values along the vertical axis (Jon Peltier, 2011).

The Line Charts

From the figure 5, the line cheat shows the comparison between the predicted probabilities from machines with a testing dataset, based on types of supervised learning algorithms. The x-axis represents the sample number in the testing dataset. The y-axis represents the predicted values from the machines. The different colors lines represent the types of supervised learning algorithms.

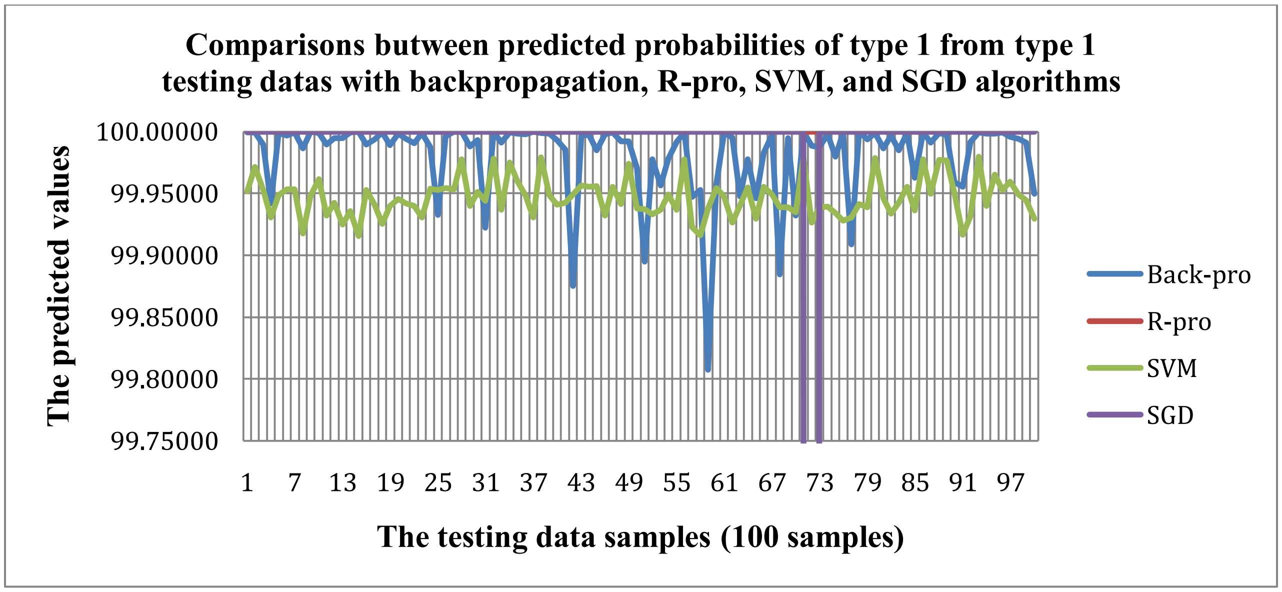


Figure 5. The comparison of the output.

From figure 6, the line chart shows the comparison of total training errors of training datasets between Back propagation algorithm and Resilient propagation algorithm. The x-axis represents the epoch numbers. The y-axis represents the percent error from the training dataset. The different color lines represent the types of supervised learning algorithms.

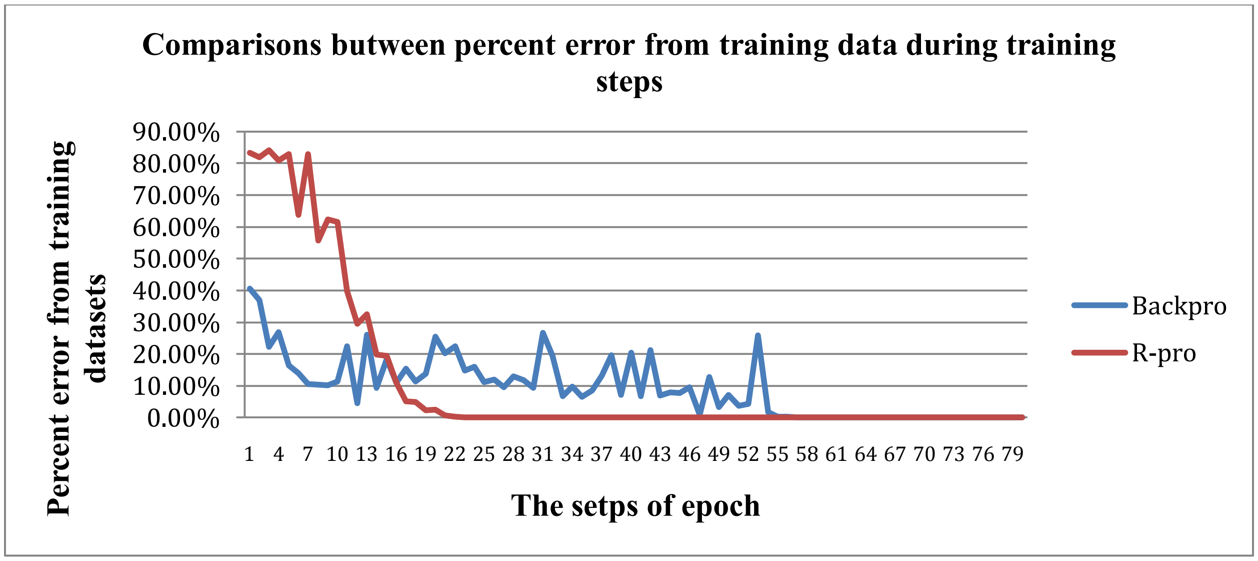


Figure 6. The comparison of the learning rates.

The Clustered Column Charts

From figure 7, the clustered column chart shows the comparisons between the percentages of prediction, from the machines, based on types of protein and supervised learning algorithms. The x-axis represents the percentages of the prediction. The y-axis represents the types of protein complexes. The different colors lines represent the types of supervised learning algorithms.

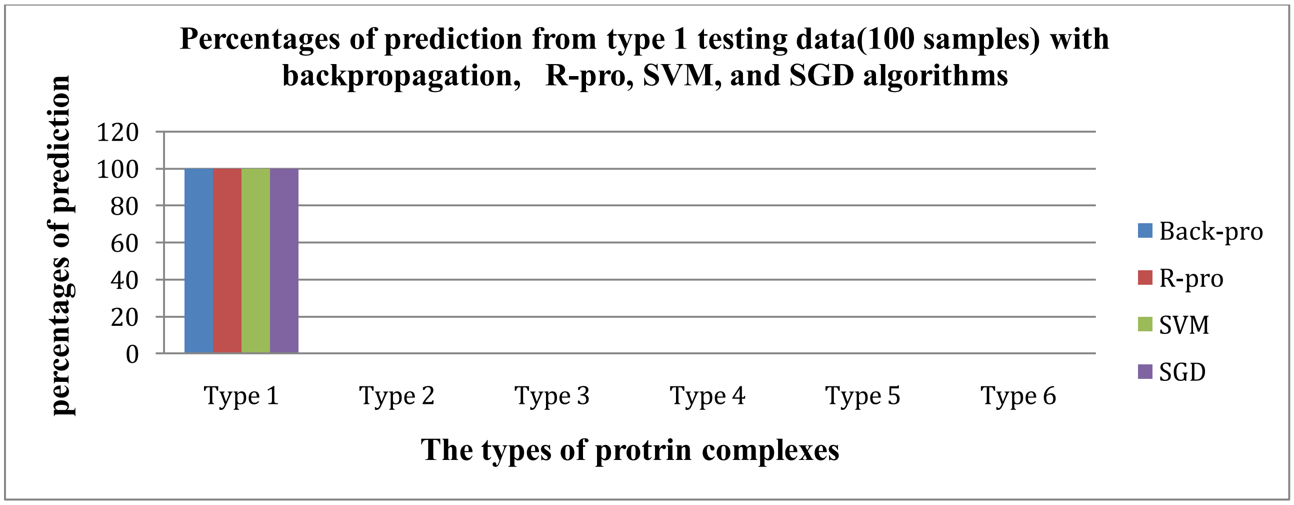


Figure 7. The percentages of the predictions.

From figure 8, the chart shows the comparison between the percentage efficiencies of the classification, from supervised learning algorithms, by using testing datasets as inputs. The x-axis represents the types of supervised learning algorithm. The y-axis represents percentage accuracies from the machines.

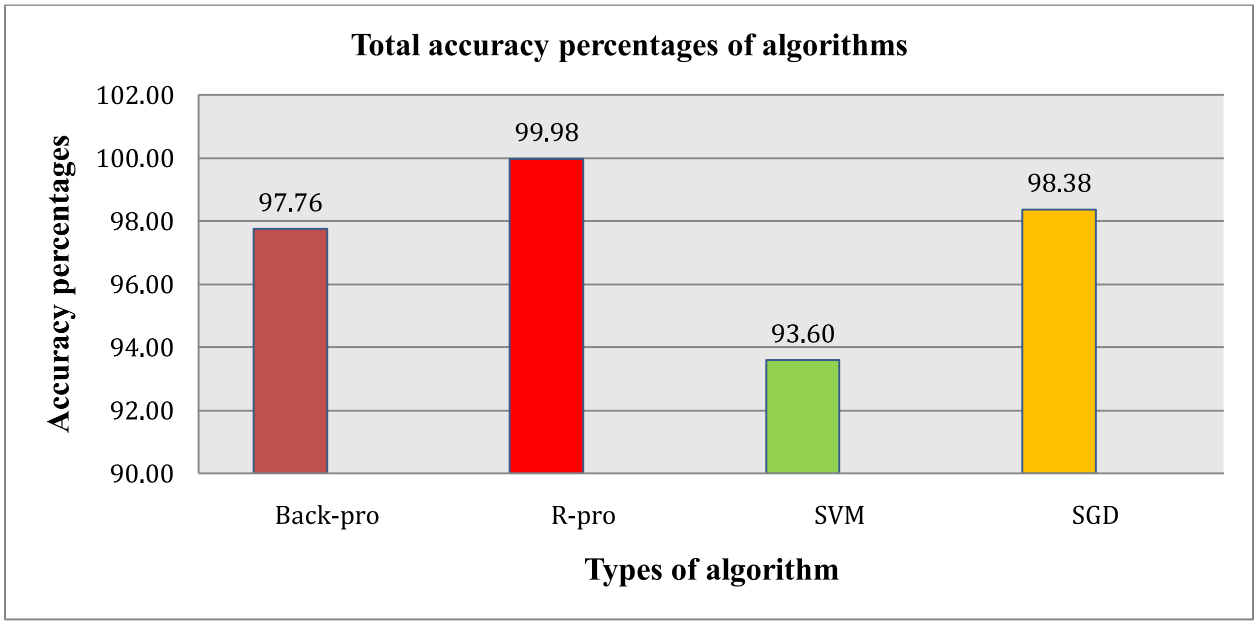


Figure 8. The accuracy of the machine models.

Chapter 4

APPLICATION AND RESULTS

To demonstrate the effectiveness of the machine learning as an aid to protein complexes classification, the results of the protein complex prediction were included. The goal of the study was to distinguish a protein complex target from the candidate protein complexes, by means of analyzing DNA sequences. The DNA sequences were downloaded from NCBI database, and then prepared into the dataset that ready to use. The training datasets were used to train the machines for creating of classification models which were used to predict the protein complex target from the candidate protein complexes. The testing dataset were used to test the machines for checking the accuracy of the machines.

Results of Using the Testing Datasets as the Inputs for Learning Machines

Type - 01 Testing Datasets

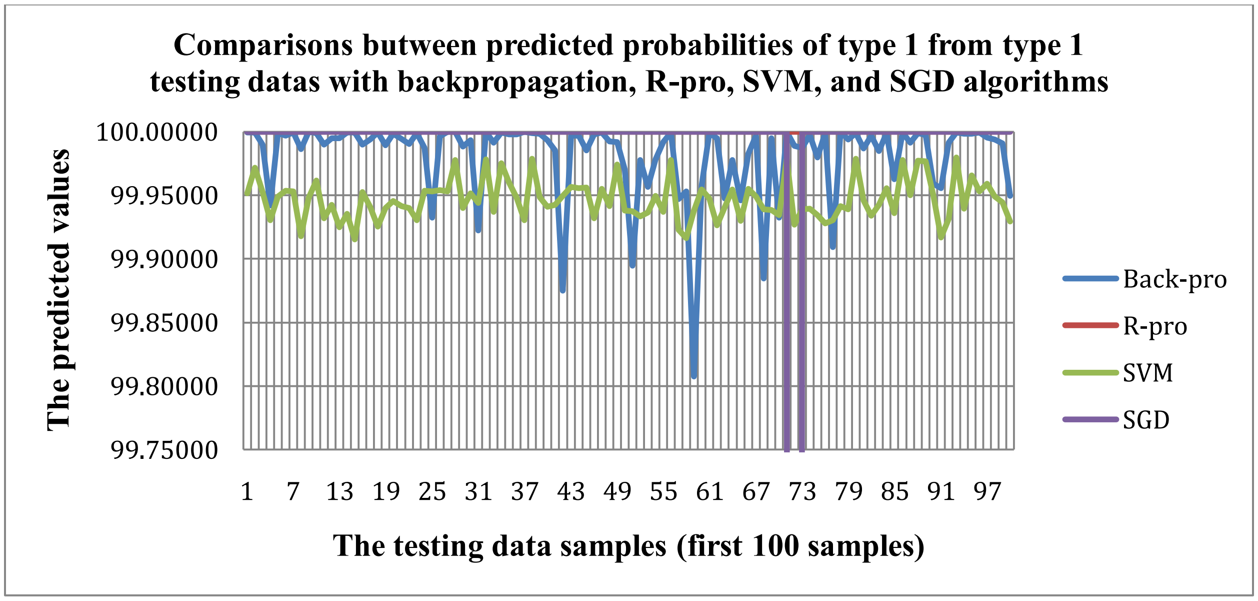


Figure 9. The comparison of type 1 outputs from type 1 testing dataset.

From figure 9, the predicted results for Rporp were all 100% in every sample. The predicted results for Backprop were between 99.80 % and 99.99%. The predicted results for SVM were between 99.91 % and 99.97%. The predicted results for SGD were between 51.00 % and 100.00%. In SGD algorithm, two samples in the model yielded 51.00% as the predicted values.

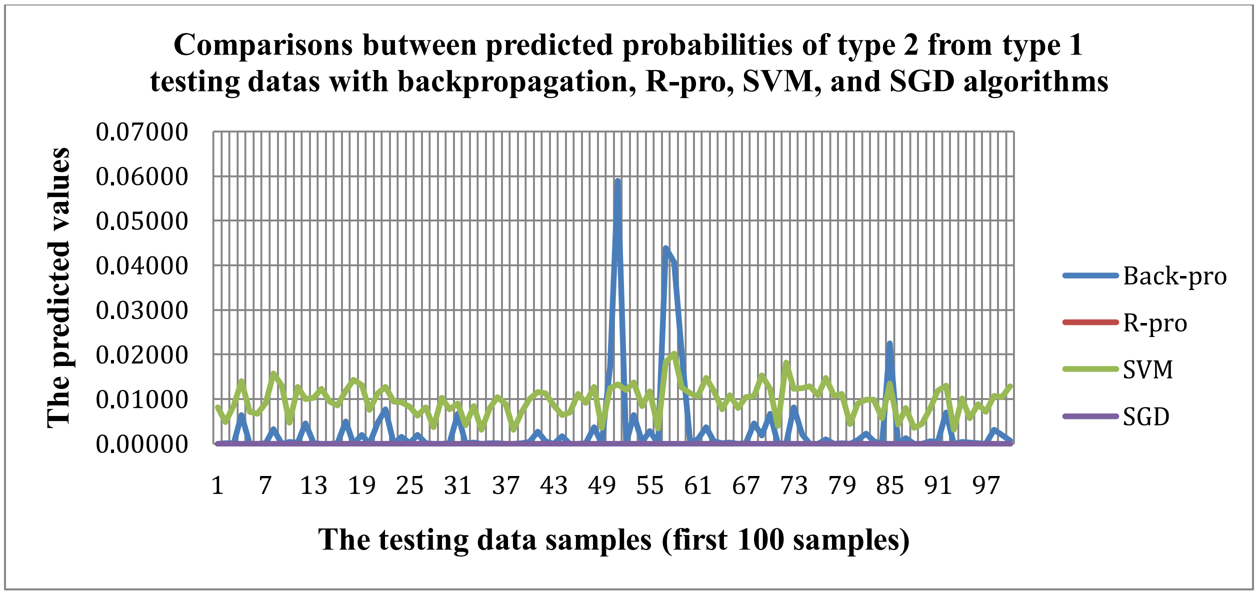


Figure 10. The comparison of type 2 outputs from type 1 testing dataset.

From figure 10, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 0.058%. The predicted results for SVM were between 0.003 % and 0.02%. The predicted results for SGD were all 0.00% in every sample.

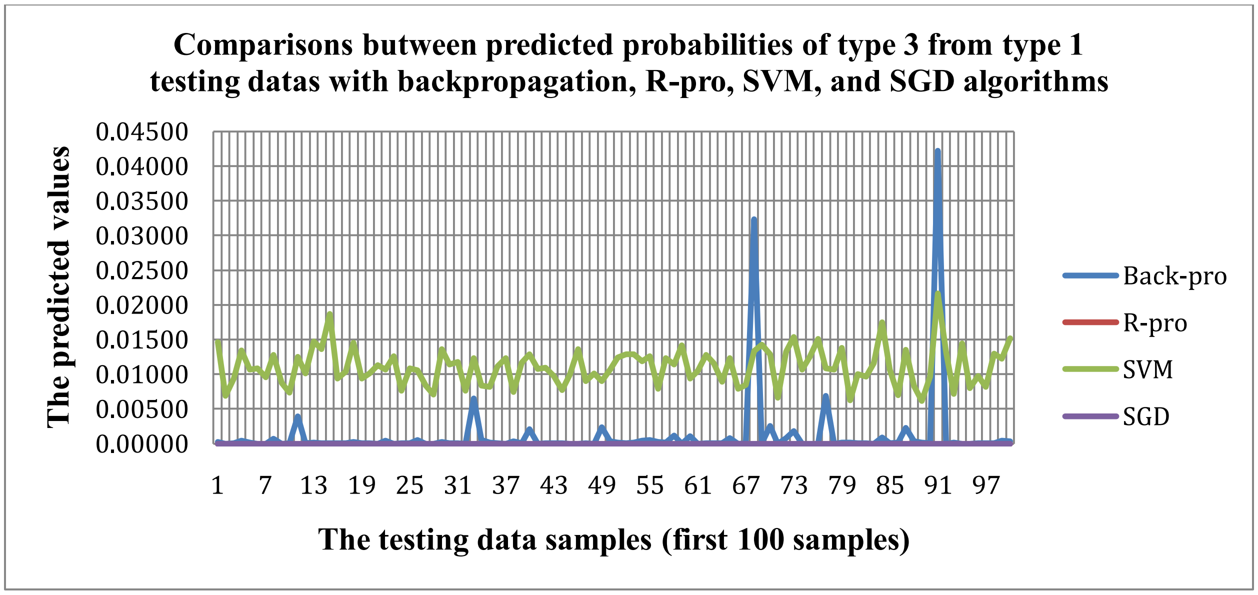


Figure 11. The comparison of type 3 outputs from type 1 testing dataset.

From figure 11, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 0.0422%. The predicted results for SVM were between 0.006 % and 0.021%. The predicted results for SGD were all 0.00% in every sample.

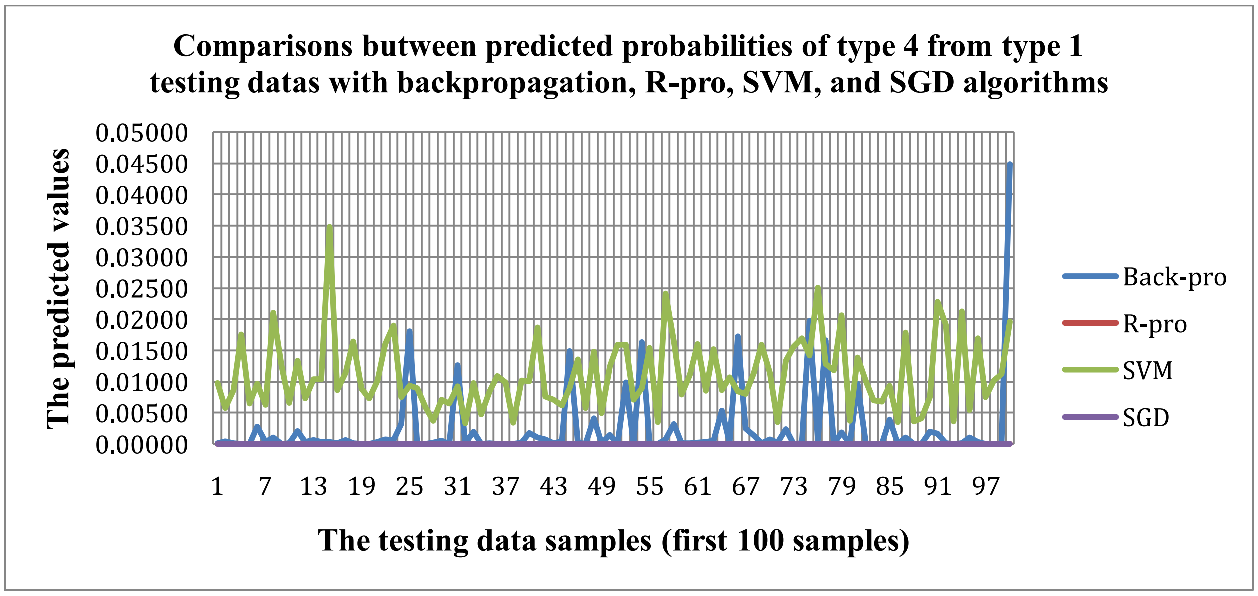


Figure 12. The comparison of type 4 outputs from type 1 testing dataset.

From figure 12, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 0.044%. The predicted results for SVM were between 0.003 % and 0.034%. The predicted results for SGD were all 0.00% in every sample.

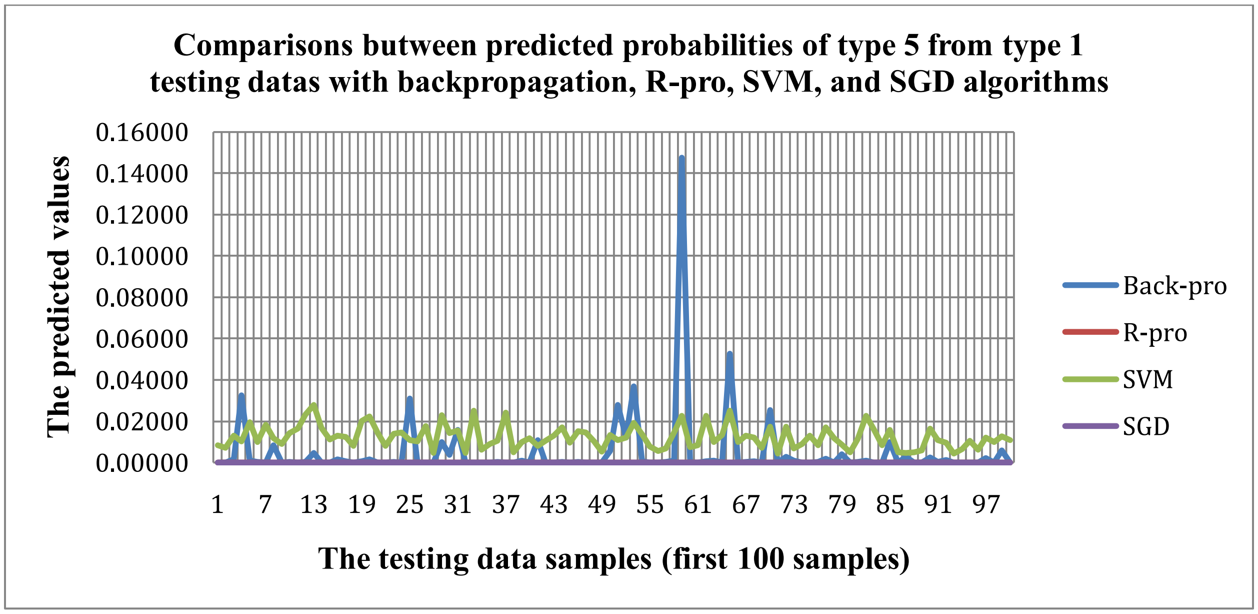


Figure 13. The comparison of type 5 outputs from type 1 testing dataset.

From figure 13, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 0.14%. The predicted results for SVM were between 0.004 % and 0.027%. The predicted results for SGD were all 0.00% in every sample.

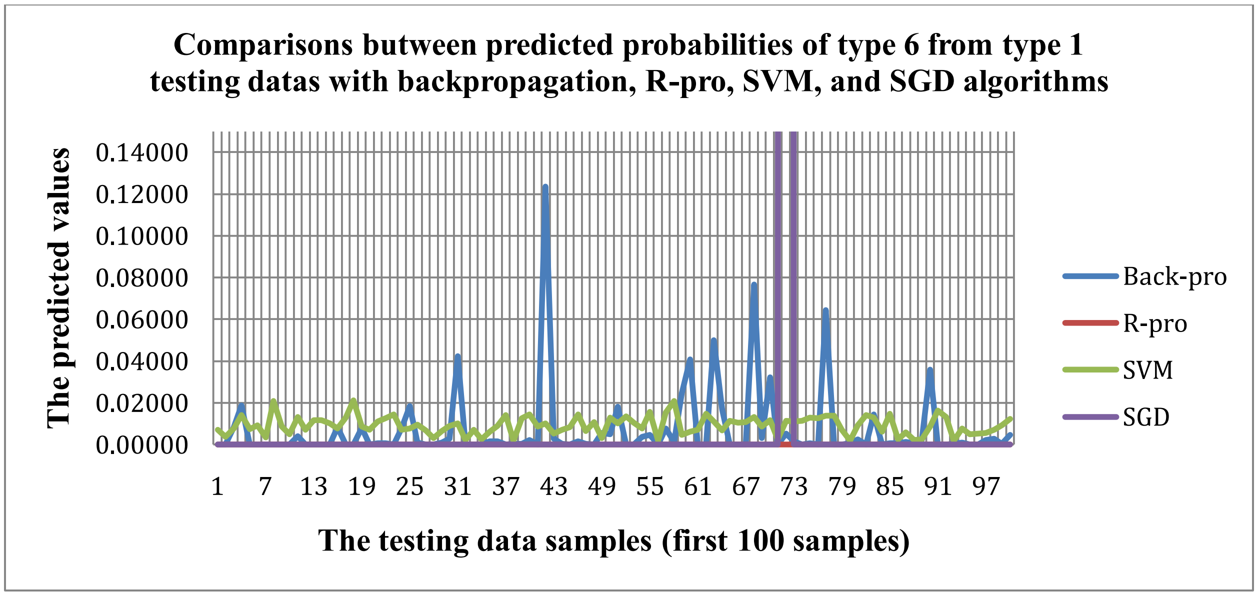


Figure 14. The comparison of type 6 outputs from type 1 testing dataset.

From figure 14, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 0.12%. The predicted results for SVM were between 0.001 % and 0.021%. The predicted results for SGD were between 0.00 % and 49.00%.

Type - 02 Testing Datasets

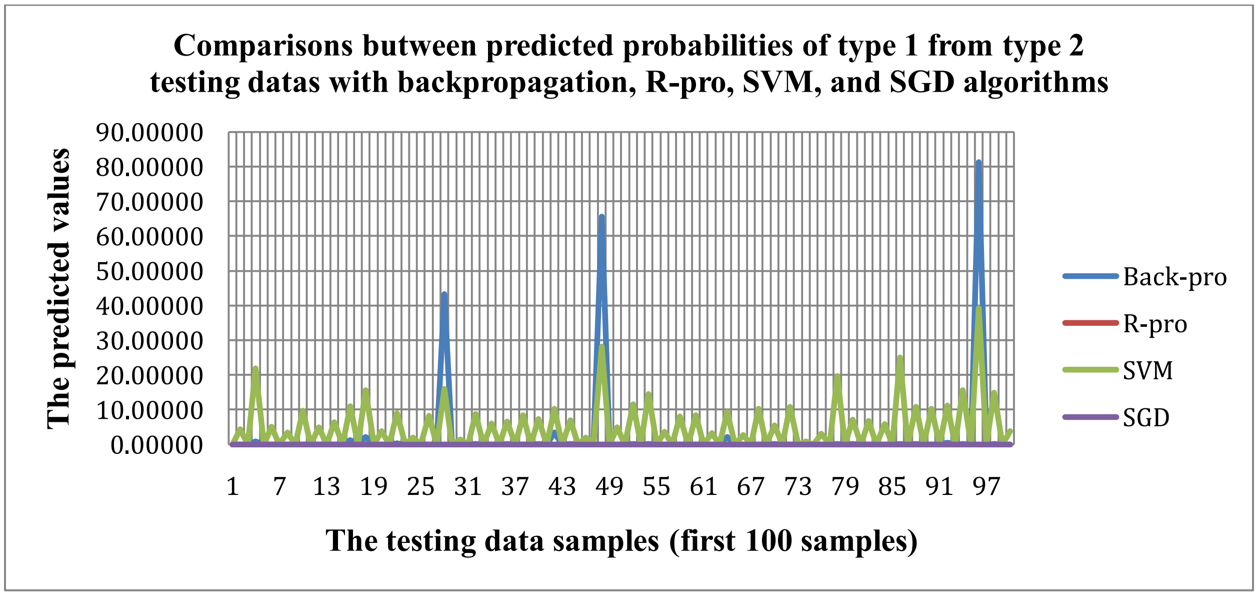


Figure 15. The comparison of type 1 outputs from type 2 testing dataset.

From figure 15, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 81.34%. The predicted results for SVM were between 0.00 % and 39.28%. The predicted results for SGD were all 0.00% in every sample.

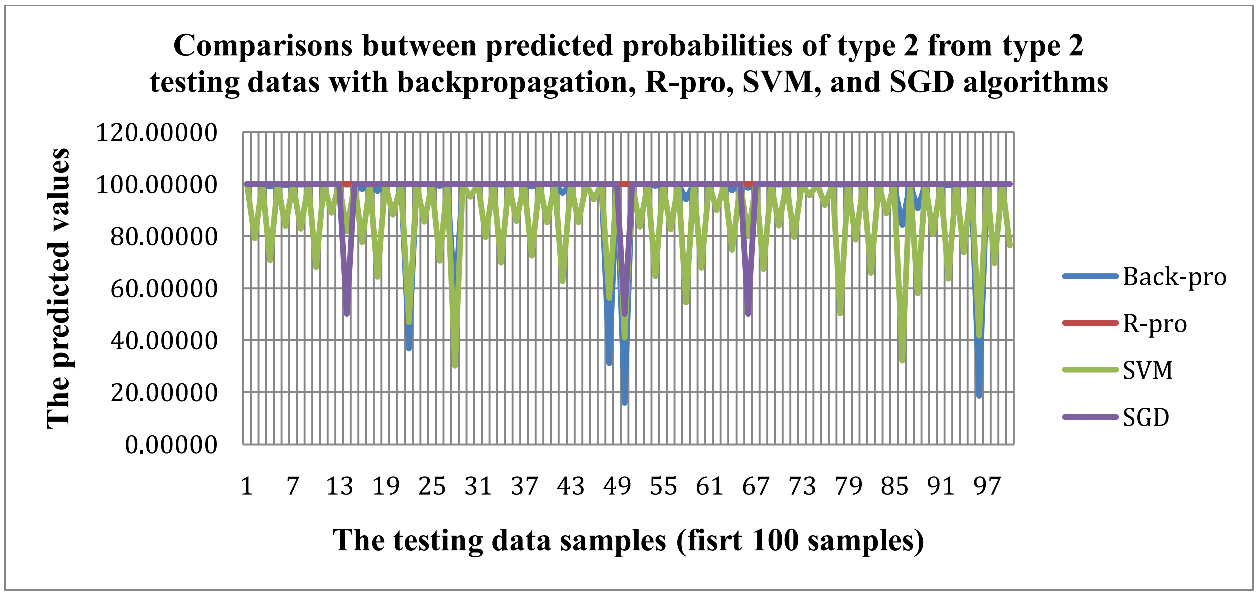


Figure 16. The comparison of type 2 outputs from type 2 testing dataset.

From figure 16, the predicted results for Rporp were all 100% in every sample. The predicted results for Backprop were between 16.00 % and 100.00%. The predicted results for SVM were between 30.16 % and 99.99%. The predicted results for SGD were between 50.00 % and 100.00%.

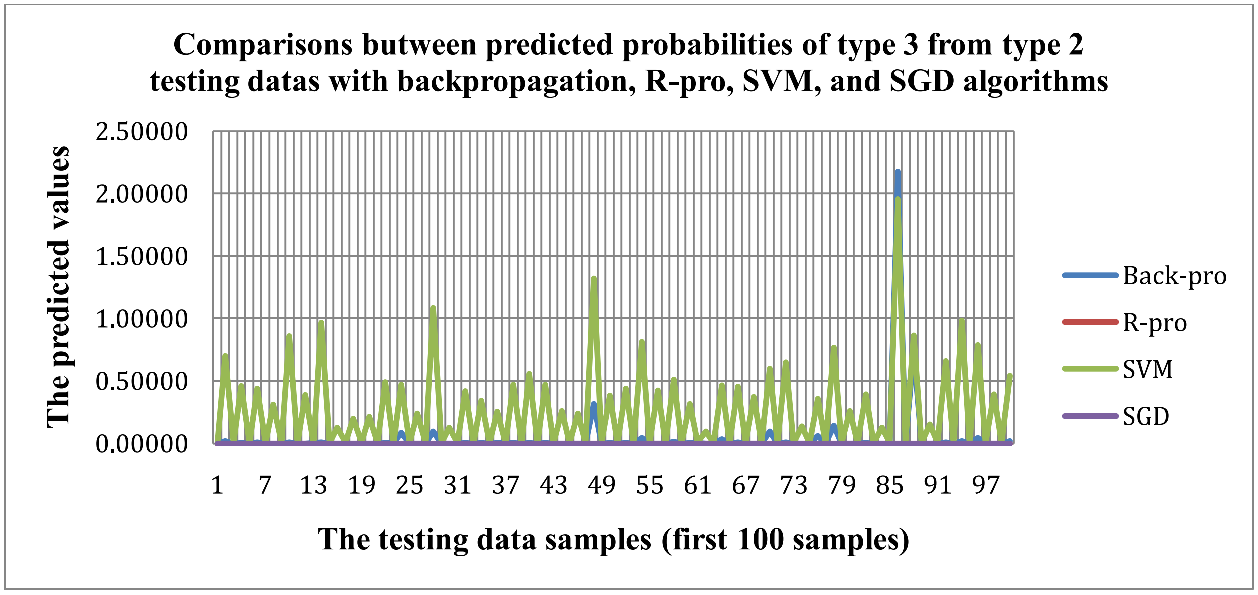


Figure 17. The comparison of type 3 outputs from type 2 testing dataset.

From figure 17, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 2.17%. The predicted results for SVM were between 0.003 % and 1.95%. The predicted results for SGD were all 0.00% in every sample.

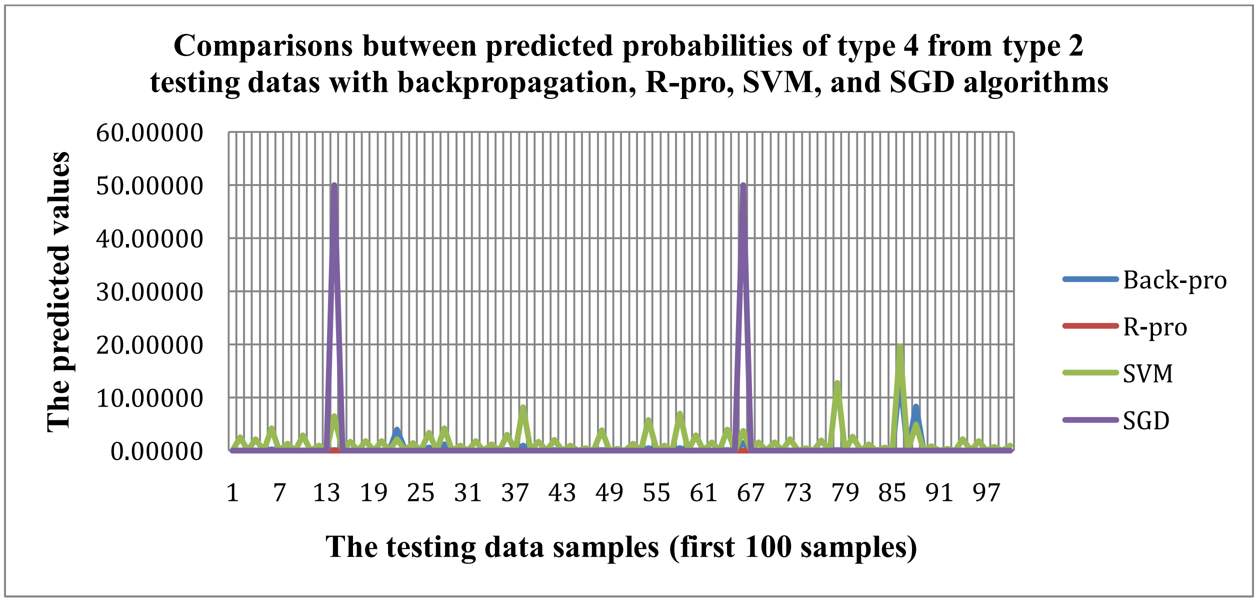


Figure 18. The comparison of type 4 outputs from type 2 testing dataset.

From figure 18, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 13.48%. The predicted results for SVM were between 0.001 % and 19.65%. The predicted results for SGD were between 0.00 % and 49.99%.

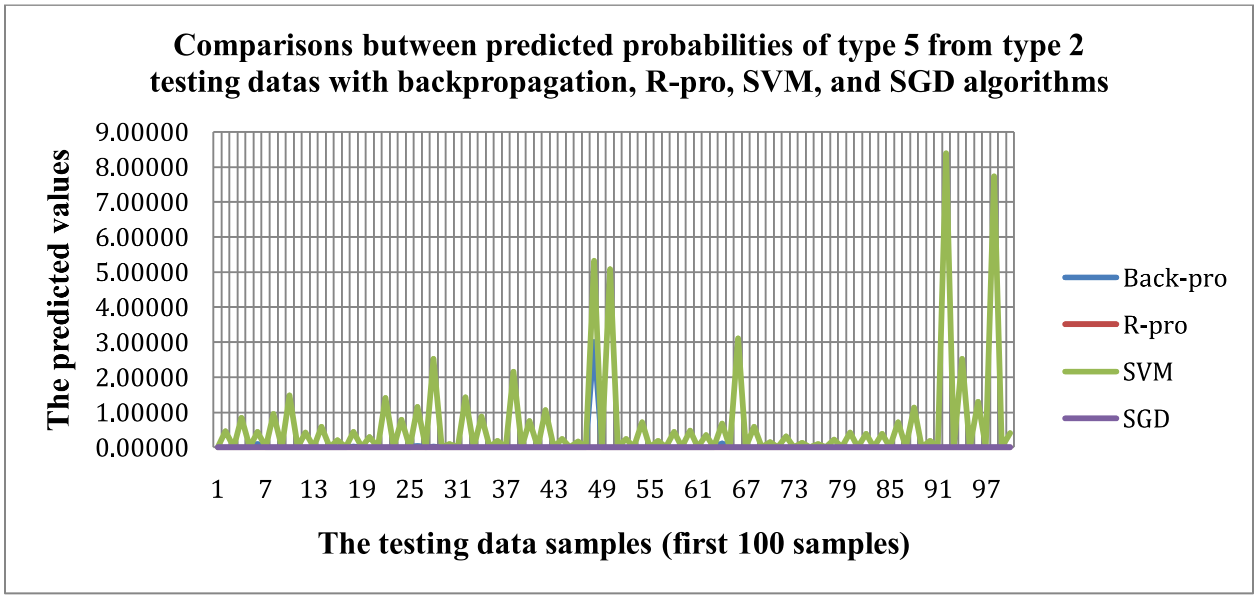


Figure 19. The comparison of type 5 outputs from type 2 testing dataset.

From figure 19, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 3.00%. The predicted results for SVM were between 0.002 % and 8.39%. The predicted results for SGD were all 0.00% in every sample.

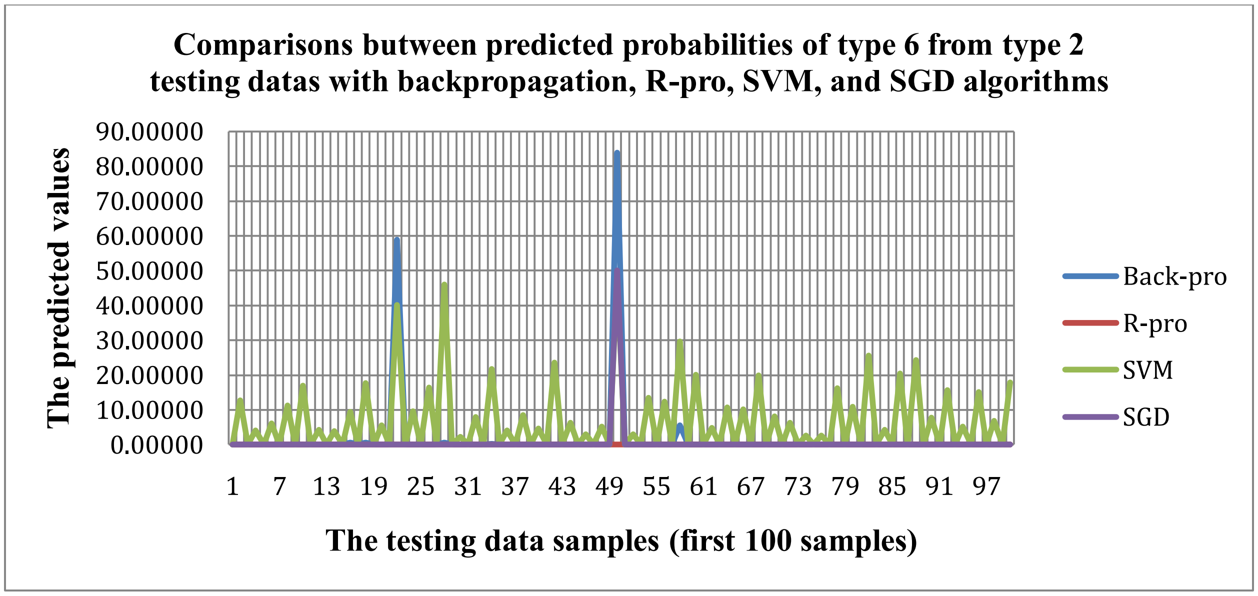


Figure 20. The comparison of type 6 outputs from type 2 testing dataset.

From figure 20, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 83.97%. The predicted results for SVM were between 0.001 % and 48.22%. The predicted results for SGD were between 0.00 % and 49.99%.

Type - 03 Testing Datasets

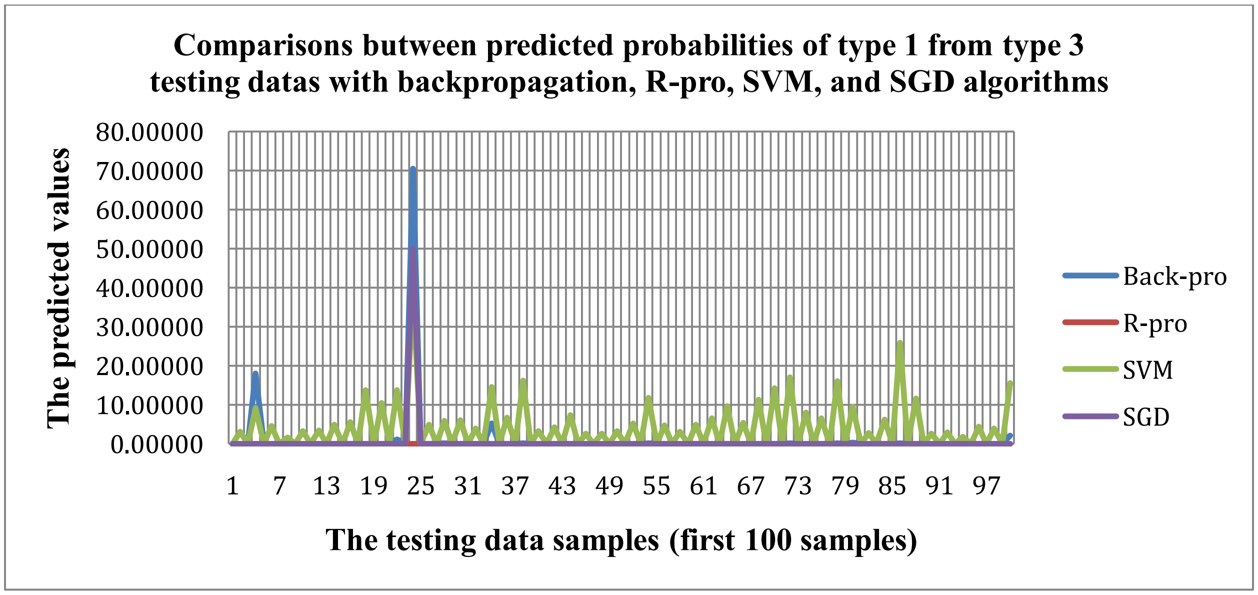


Figure 21. The comparison of type 1 outputs from type 3 testing dataset.

From figure 21, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 70.52%. The predicted results for SVM were between 0.00 % and 39.11%. The predicted results for SGD were between 0.00 % and 50.00%.

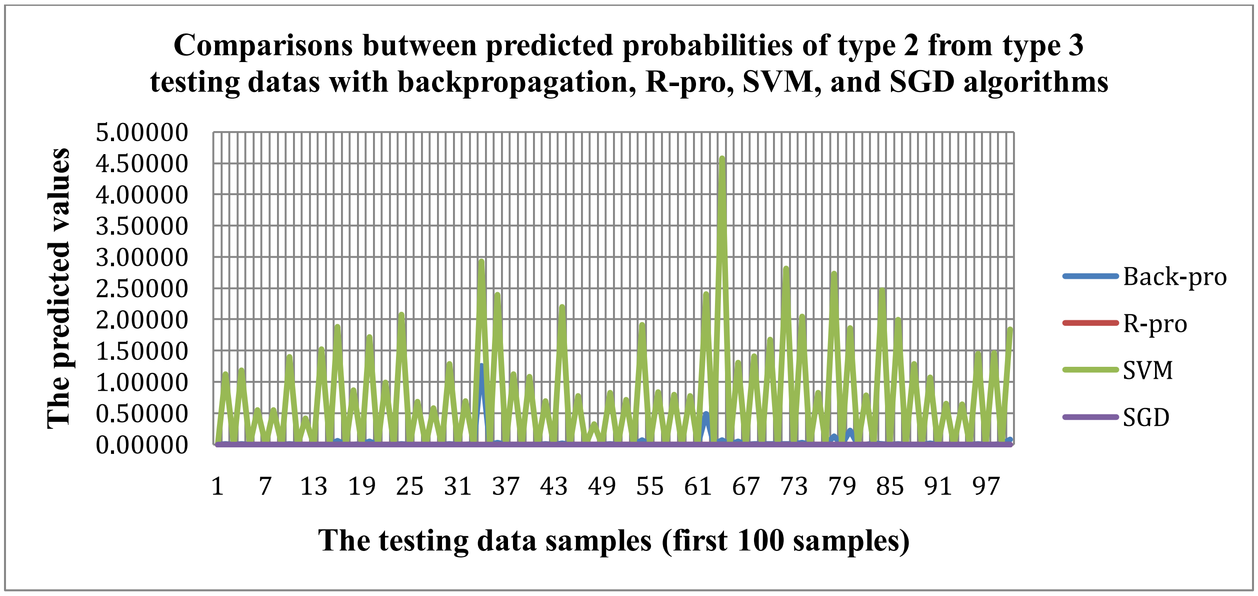


Figure 22. The comparison of type 2 outputs from type 3 testing dataset.

From figure 22, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 1.25%. The predicted results for SVM were between 0.00 % and 4.57%. The predicted results for SGD were all 0.00% in every sample.

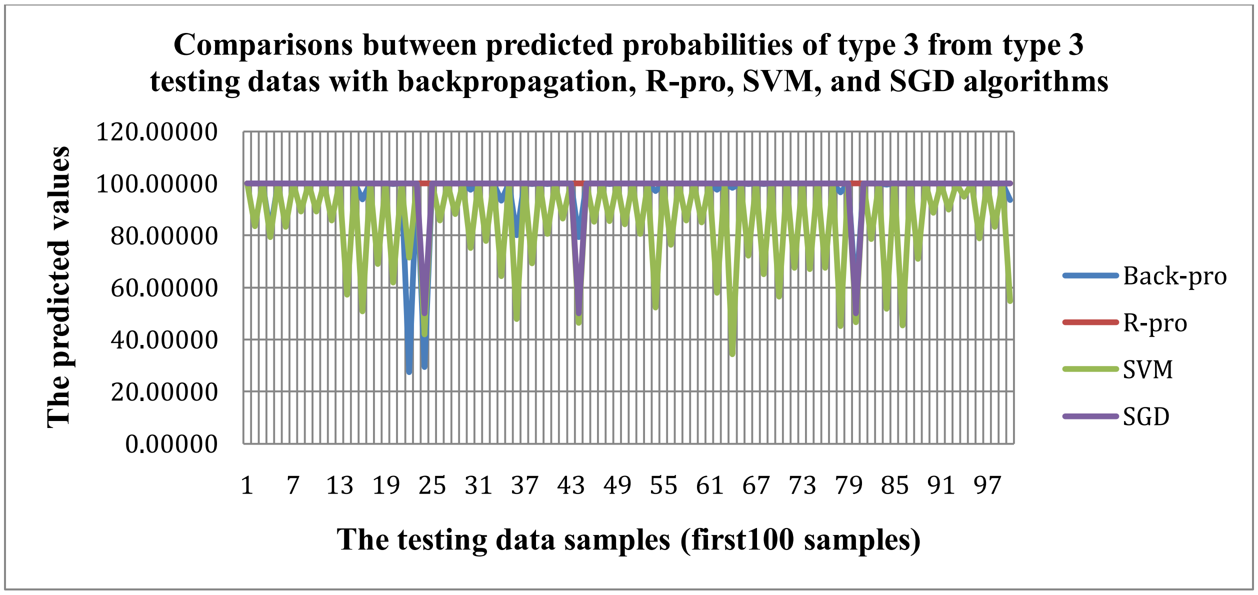


Figure 23. The comparison of type 3 outputs from type 3 testing dataset.

From figure 23, the predicted results for Rporp were all 100.00% in every sample. The predicted results for Backprop were between 27.46 % and 100.00%. The predicted results for SVM were between 34.36 % and 99.99%. The predicted results for SGD were between 49.99 % and 100.00%.

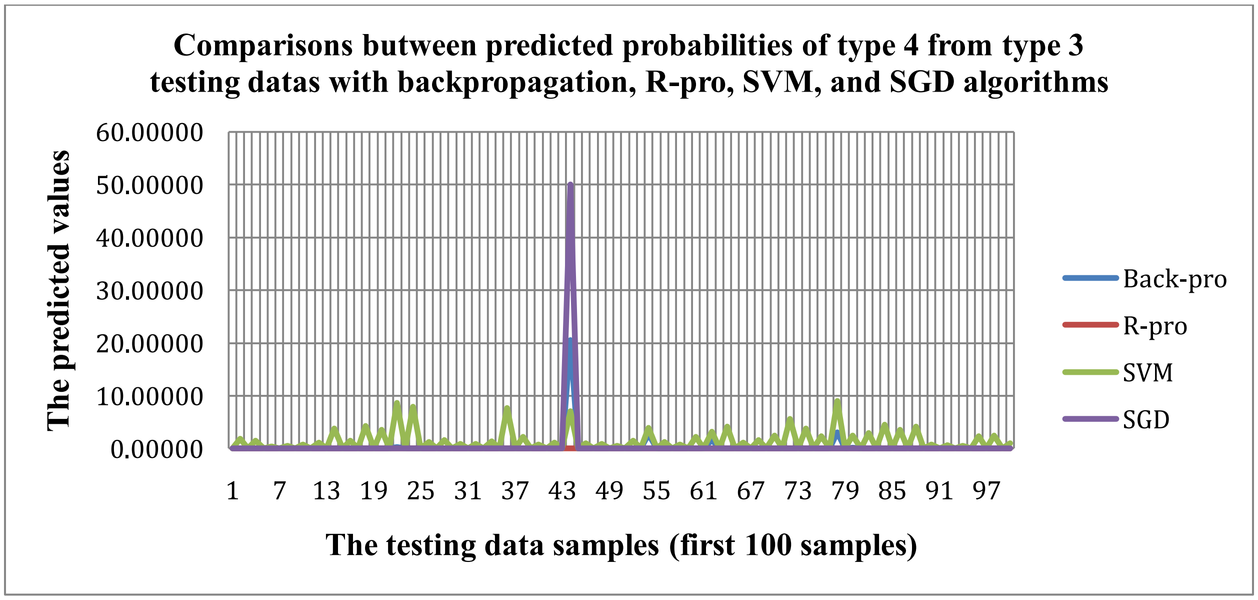


Figure 24. The comparison of type 4 outputs from type 3 testing dataset.

From figure 24, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 20.51%. The predicted results for SVM were between 0.0004 % and 9.09%. The predicted results for SGD were between 0.00 % and 49.99%.

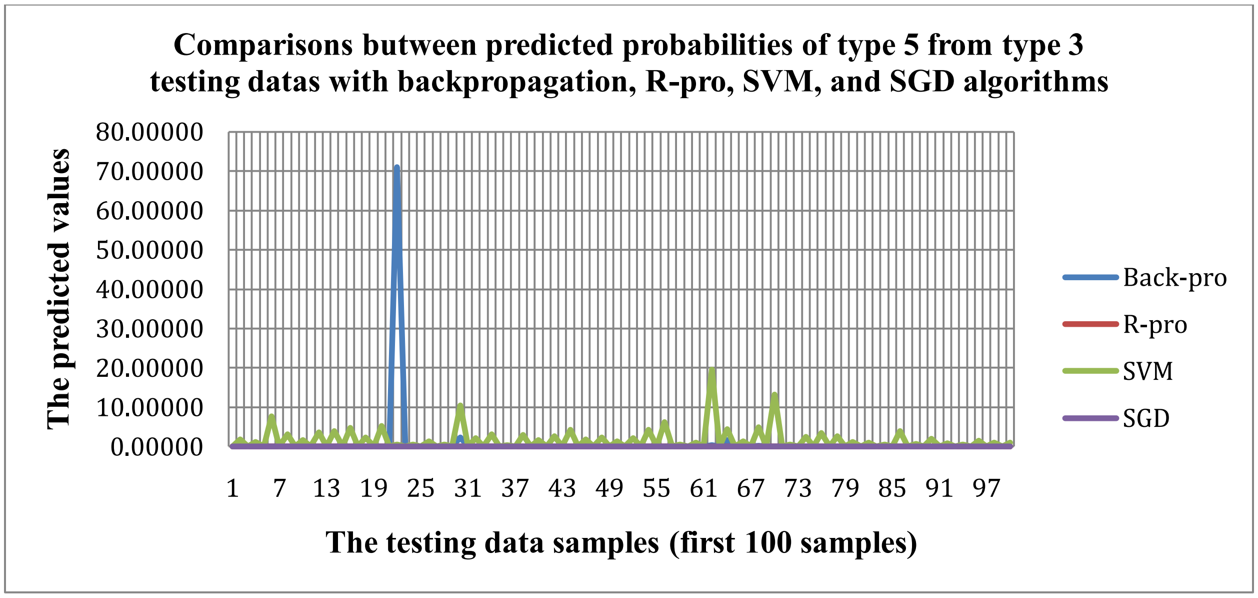


Figure 25. The comparison of type 5 outputs from type 3 testing dataset.

From figure 25, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 71.04%. The predicted results for SVM were between 0.001 % and 19.61%. The predicted results for SGD were all 0.00% in every sample.

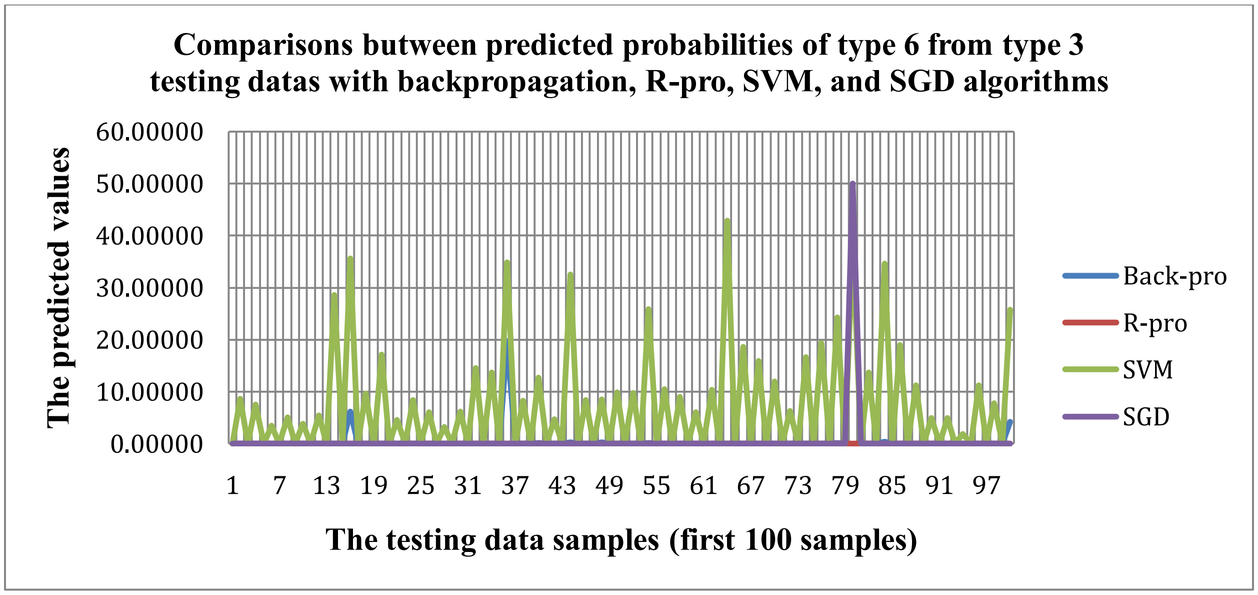


Figure 26. The comparison of type 6 outputs from type 3 testing dataset.

From figure 26, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 43.20%. The predicted results for SVM were between 0.0007 % and 42.87%. The predicted results for SGD were between 0.00 % and 49.99%.

Type - 04 Testing Datasets

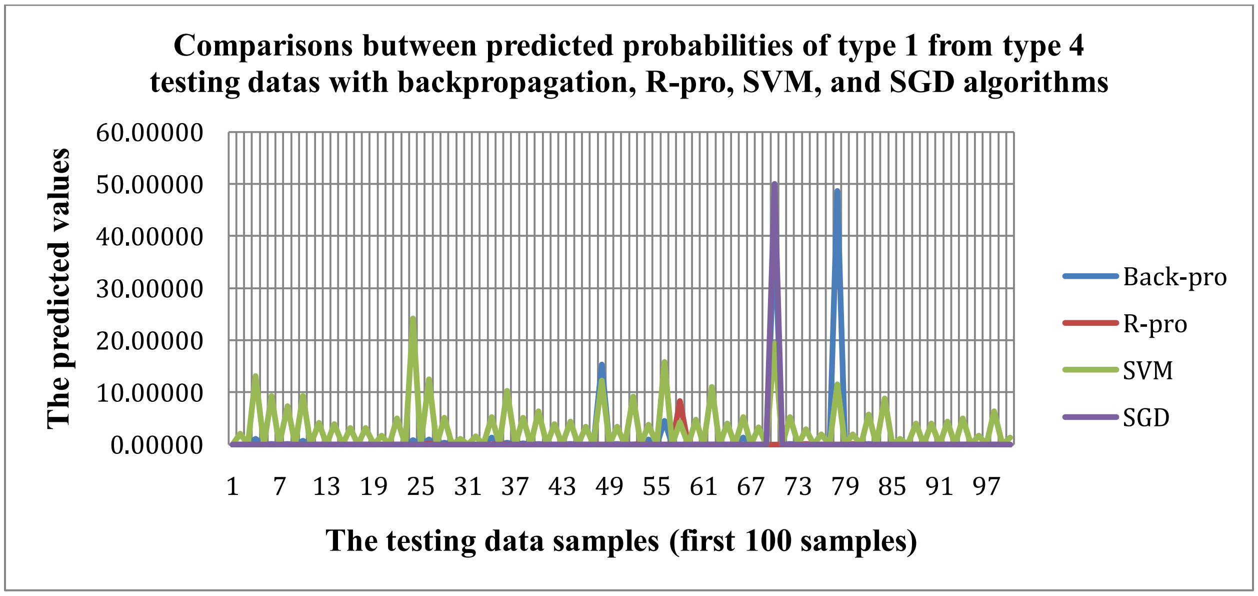


Figure 27. The comparison of type 1 outputs from type 4 testing dataset.

From figure 27, the predicted results for Rporp were between 0.00 % and 8.34%. The predicted results for Backprop were between 0.00 % and 48.65%. The predicted results for SVM were between 0.00 % and 24.21%. The predicted results for SGD were between 0.00 % and 49.99%.

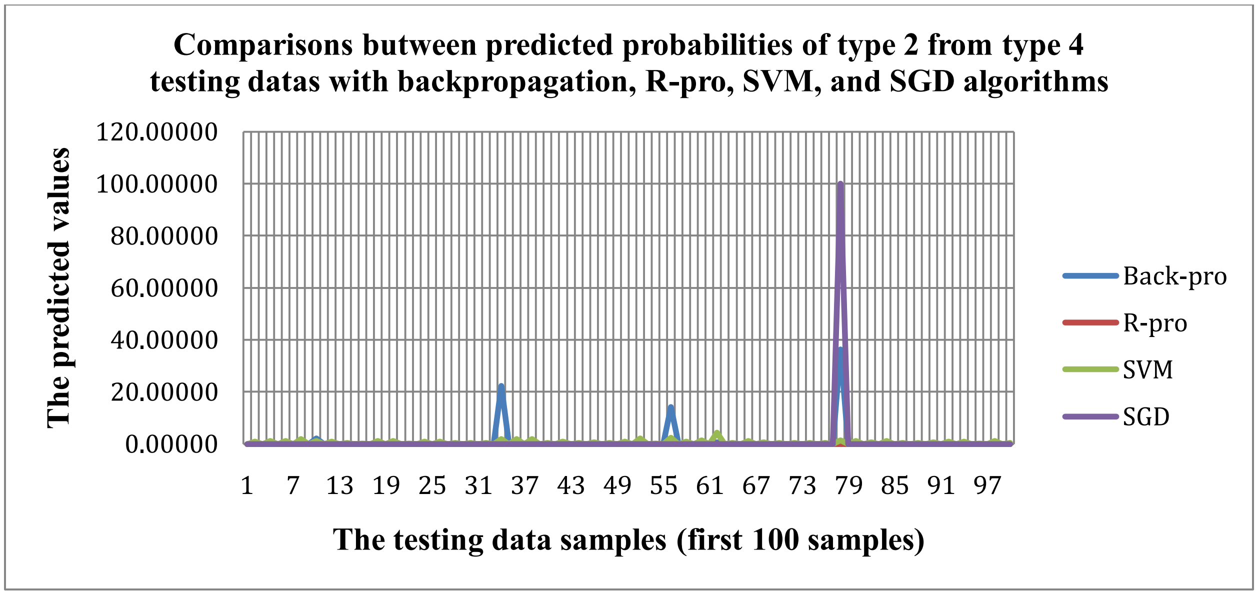


Figure 28. The comparison of type 2 outputs from type 4 testing dataset.

From figure 28, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 48.65%. The predicted results for SVM were between 0.00 % and 4.229%. The predicted results for SGD were between 0.00 % and 100.00%.

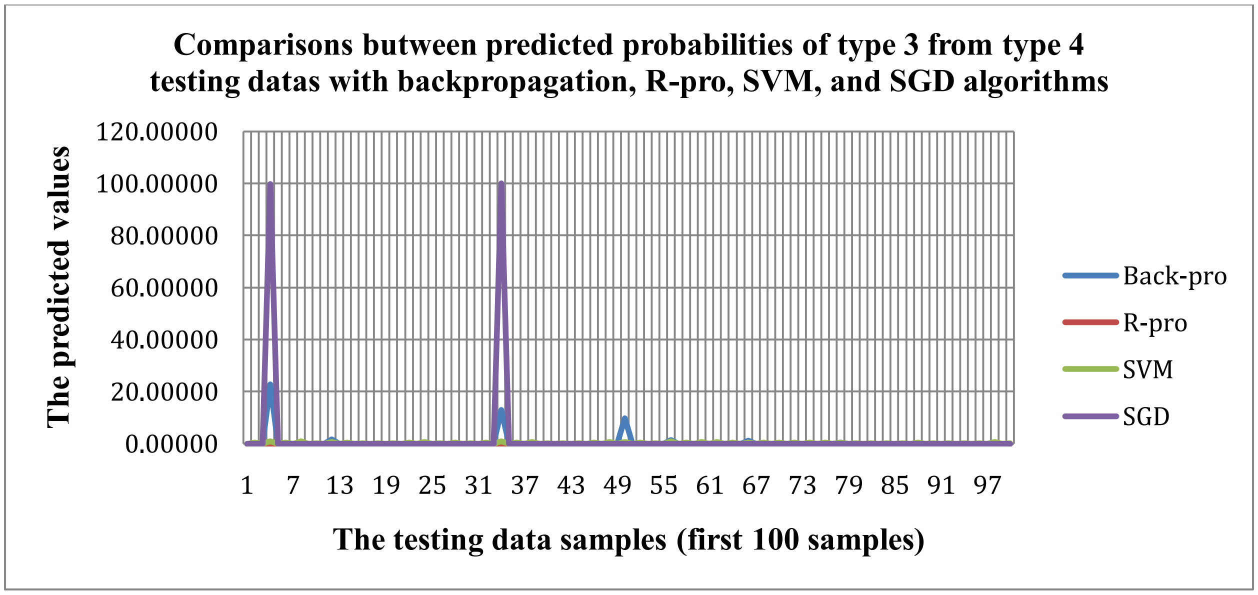


Figure 29. The comparison of type 3 outputs from type 4 testing dataset.

From figure 29, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 22.96%. The predicted results for SVM were between 0.00 % and 0.98 %. The predicted results for SGD were between 0.00 % and 100.00%.

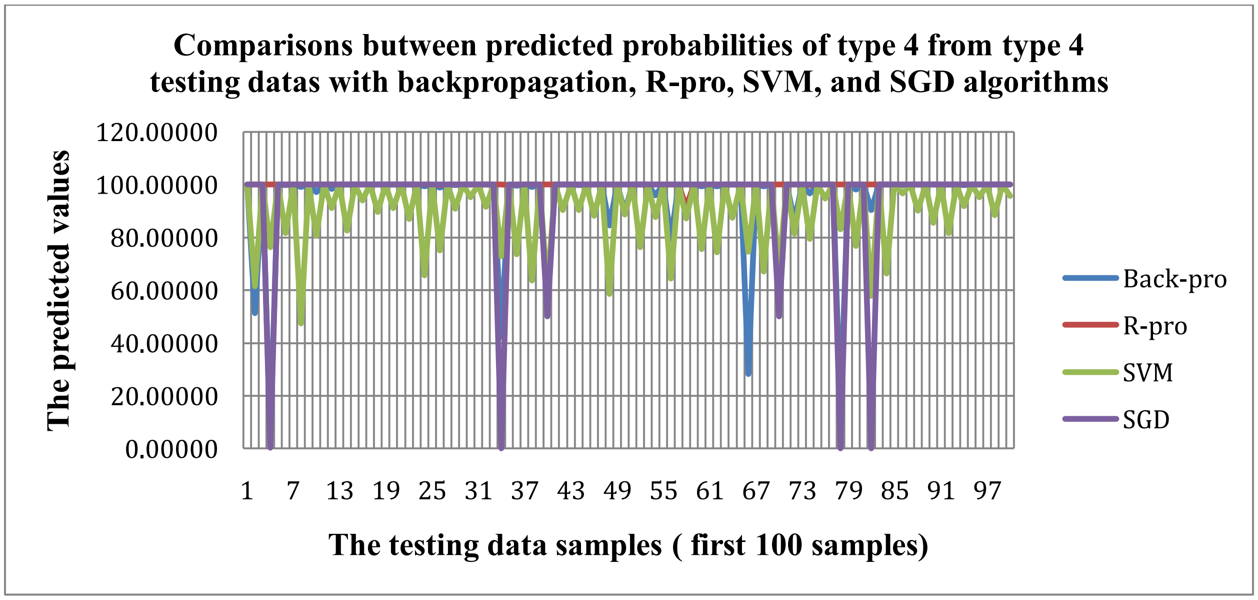


Figure 30. The comparison of type 4 outputs from type 4 testing dataset.

From figure 30, the predicted results for Rporp were between 91.65 % and 100.00%. The predicted results for Backprop were between 4.40 % and 99.99%. The predicted results for SVM were between 47.40 % and 99.99%. The predicted results for SGD were between 0.00 % and 100.00%.

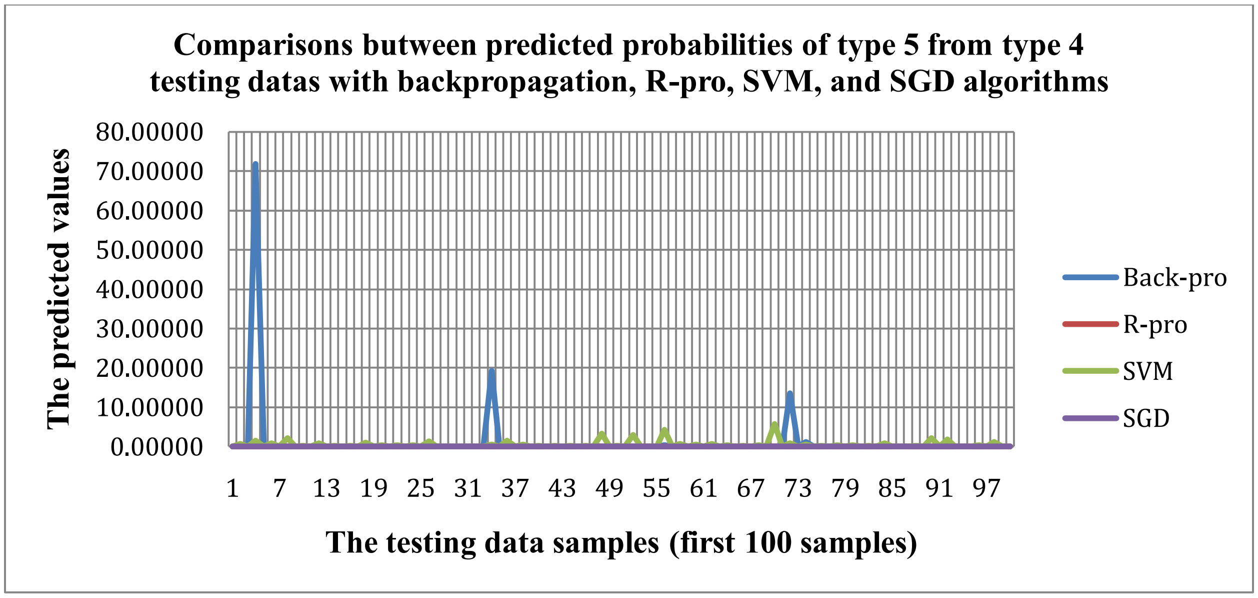


Figure 31. The comparison of type 5 outputs from type 4 testing dataset.

From figure 31, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 71.79 %. The predicted results for SVM were between 0.003 % and 5.77 %. The predicted results for SGD were all 0.00% in every sample.

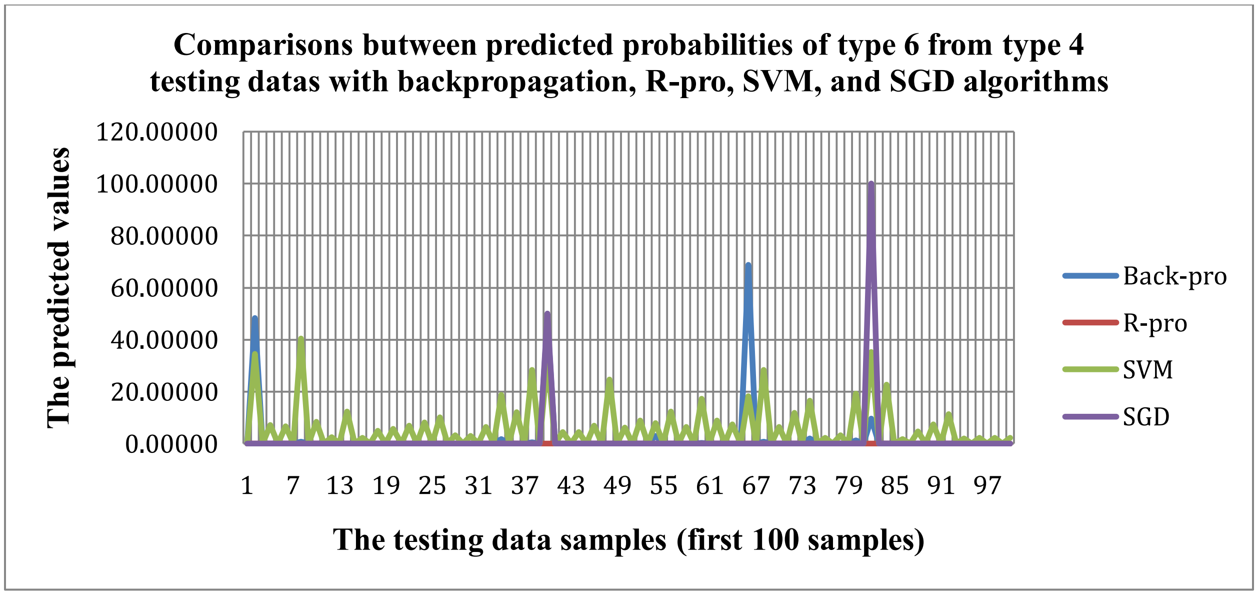


Figure 32. The comparison of type 6 outputs from type 4 testing dataset.

From figure 32, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 68.78 %. The predicted results for SVM were between 0.001 % and 40.48 %. The predicted results for SGD were between 0.00 % and 100.00 %.

Type - 05 Testing Datasets

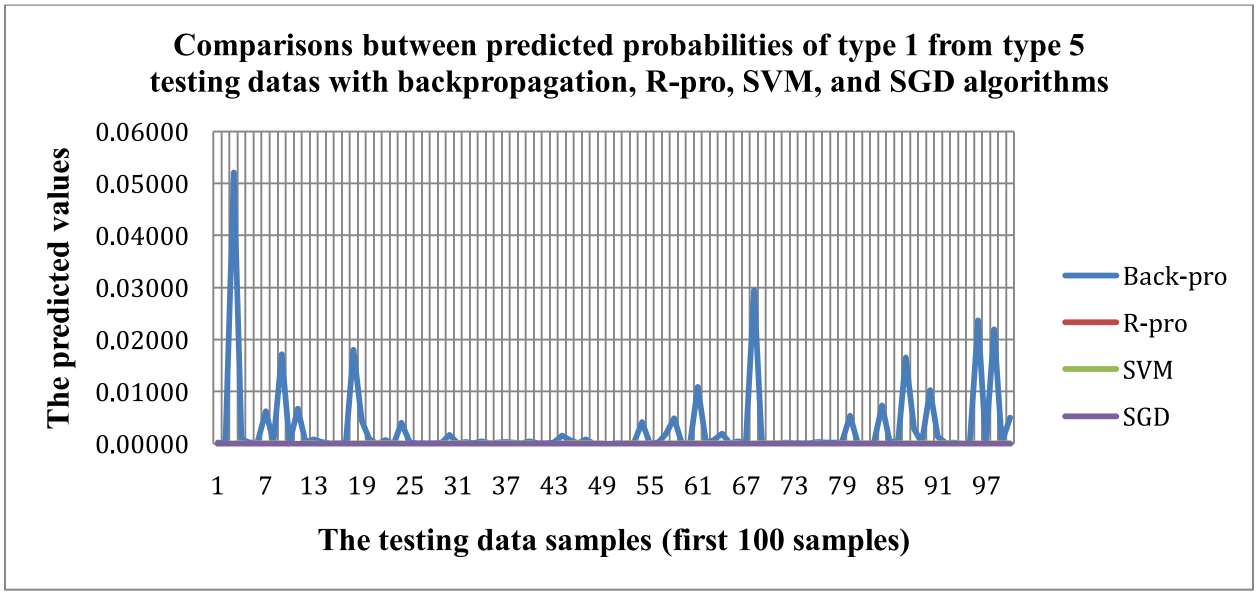


Figure 33. The comparison of type 1 outputs from type 5 testing dataset.

From figure 33, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 0.05 %. The predicted results for SVM were between 0.00 % and 0.0001 %. The predicted results for SGD were all 0.00% in every sample.

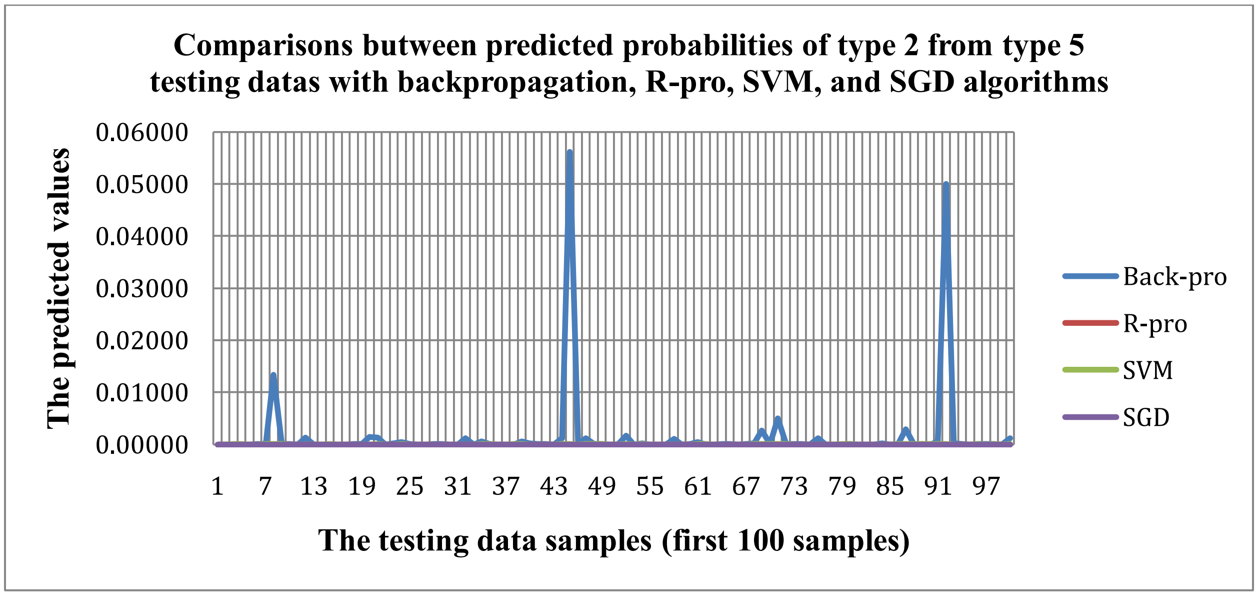


Figure 34. The comparison of type 2 outputs from type 5 testing dataset.

From figure 34, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 0.05 %. The predicted results for SVM were between 0.00 % and 0.00004 %. The predicted results for SGD were all 0.00% in every sample.

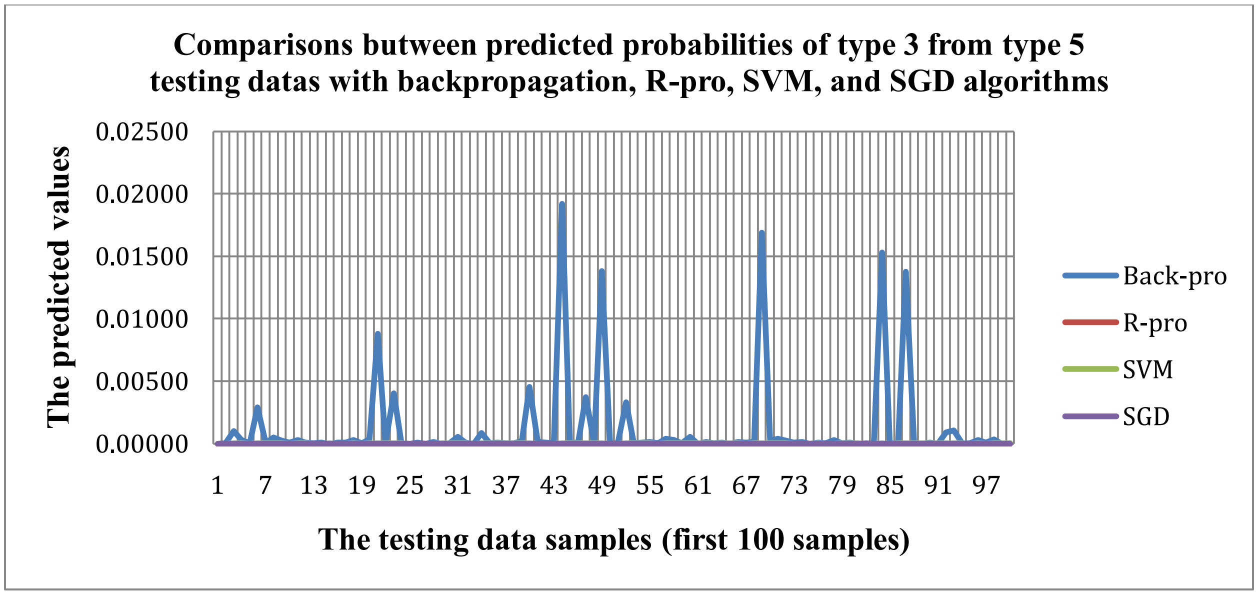


Figure 35. The comparison of type 3 outputs from type 5 testing dataset.

From figure 35, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 0.01 %. The predicted results for SVM were between 0.00 % and 0.00003 %. The predicted results for SGD were all 0.00% in every sample.

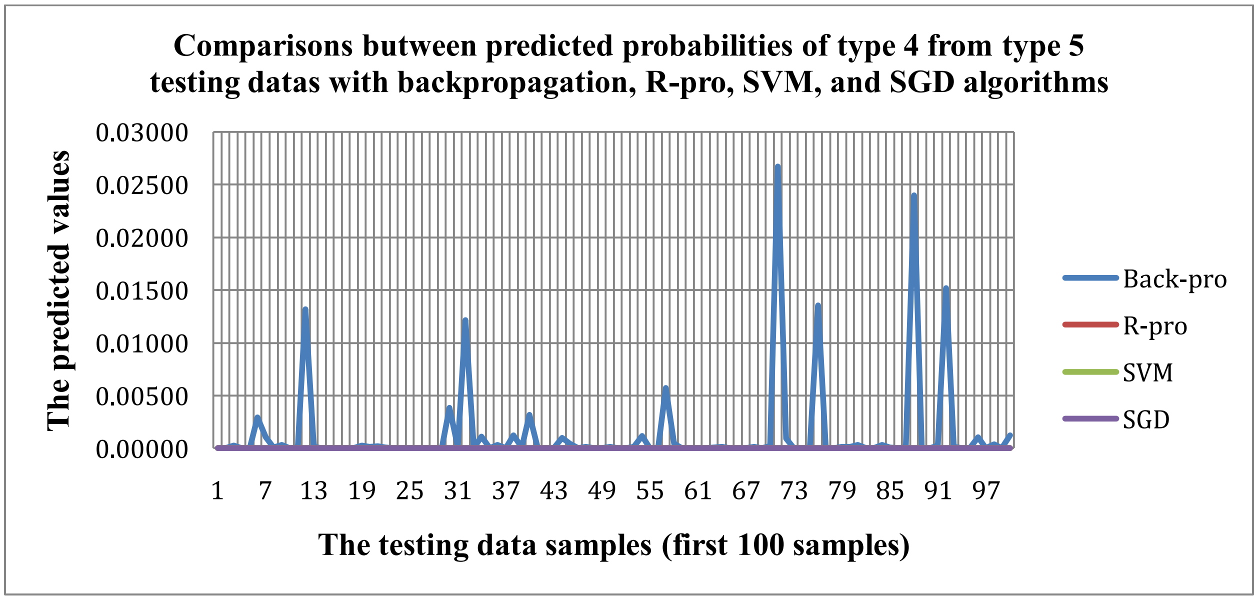


Figure 36. The comparison of type 4 outputs from type 5 testing dataset.

From figure 36, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 0.01 %. The predicted results for SVM were between 0.00 % and 0.00003 %. The predicted results for SGD were all 0.00% in every sample.

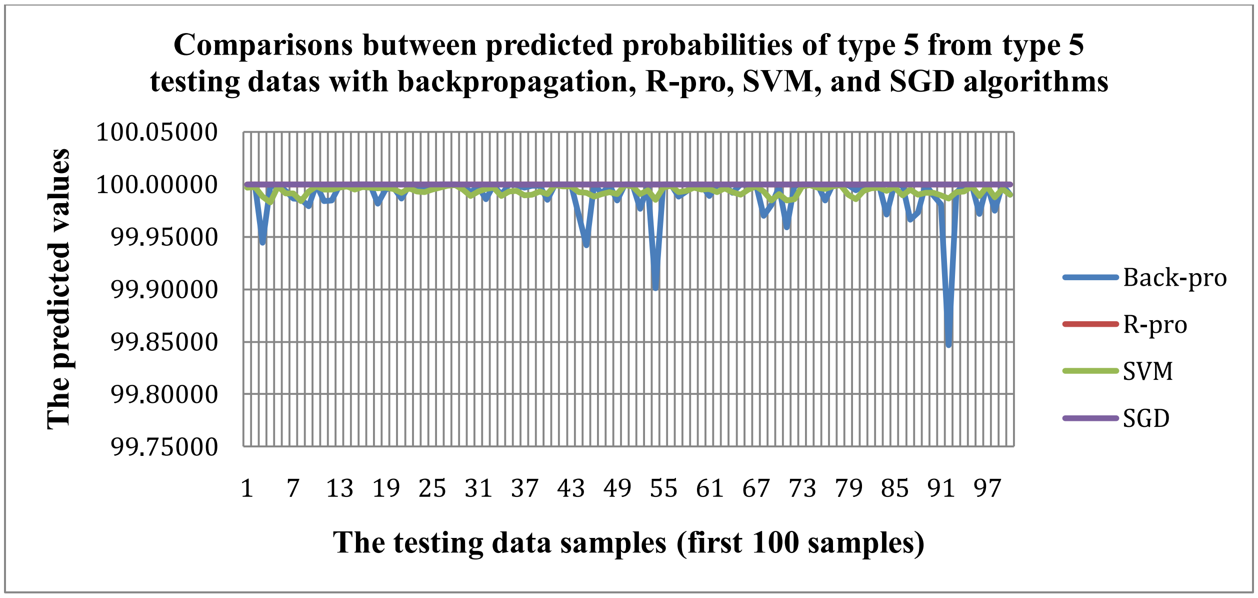


Figure 37. The comparison of type 5 outputs from type 5 testing dataset.

From figure 37, the predicted results for Rporp were all 100.00% in every sample. The predicted results for Backprop were between 99.84 % and 99.99 %. The predicted results for SVM were between 99.98 % and 99.99 %. The predicted results for SGD were all 100.00% in every sample.

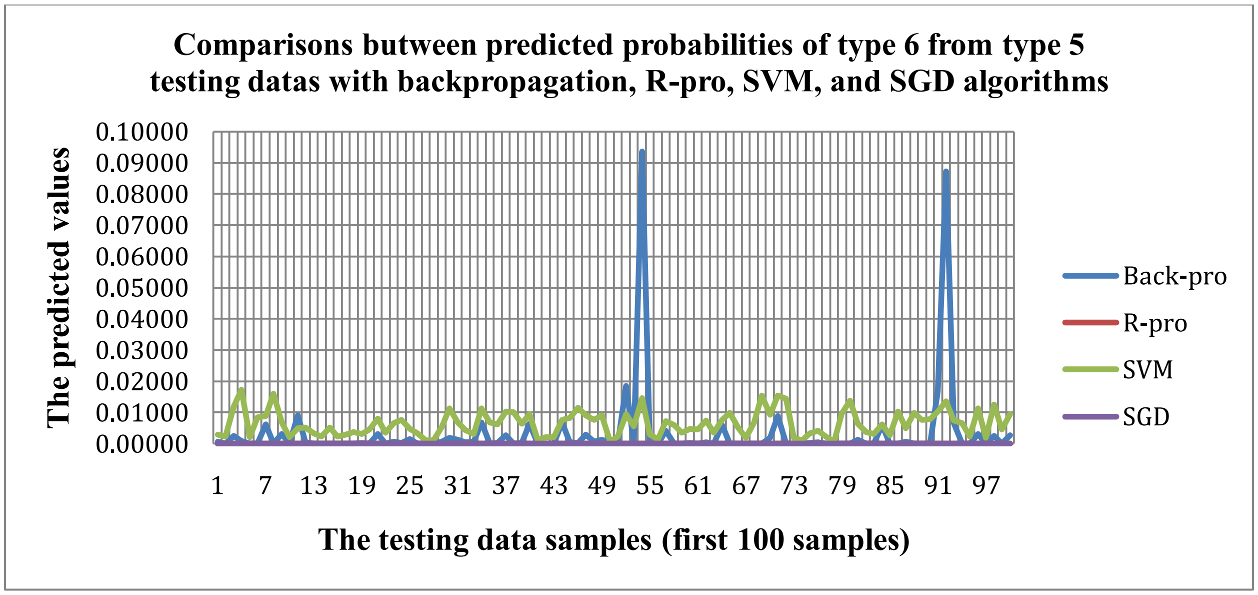


Figure 38. The comparison of type 6 outputs from type 5 testing dataset.

From figure 38, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 0.09 %. The predicted results for SVM were between 0.0005 % and 0.01 %. The predicted results for SGD were all 0.00% in every sample.

Type - 06 Testing Datasets

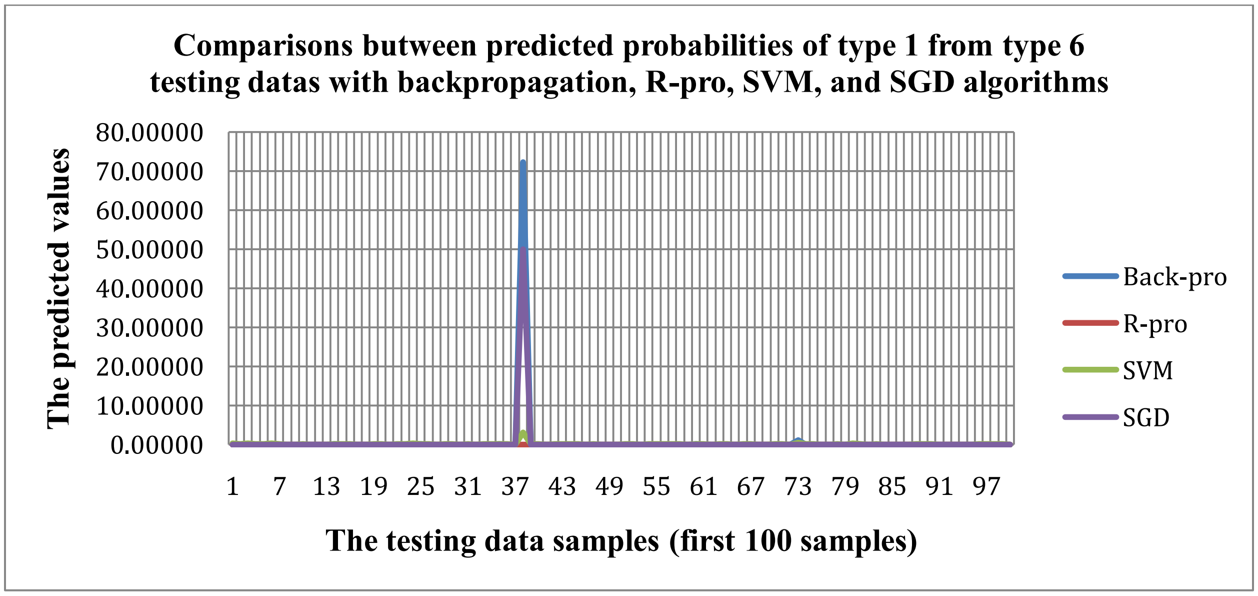


Figure 39. The comparison of type 1 outputs from type 6 testing dataset.

From figure 39, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 72.28 %. The predicted results for SVM were between 0.00 % and 3.06 %. The predicted results for SGD were between 0.00 % and 49.99 %.

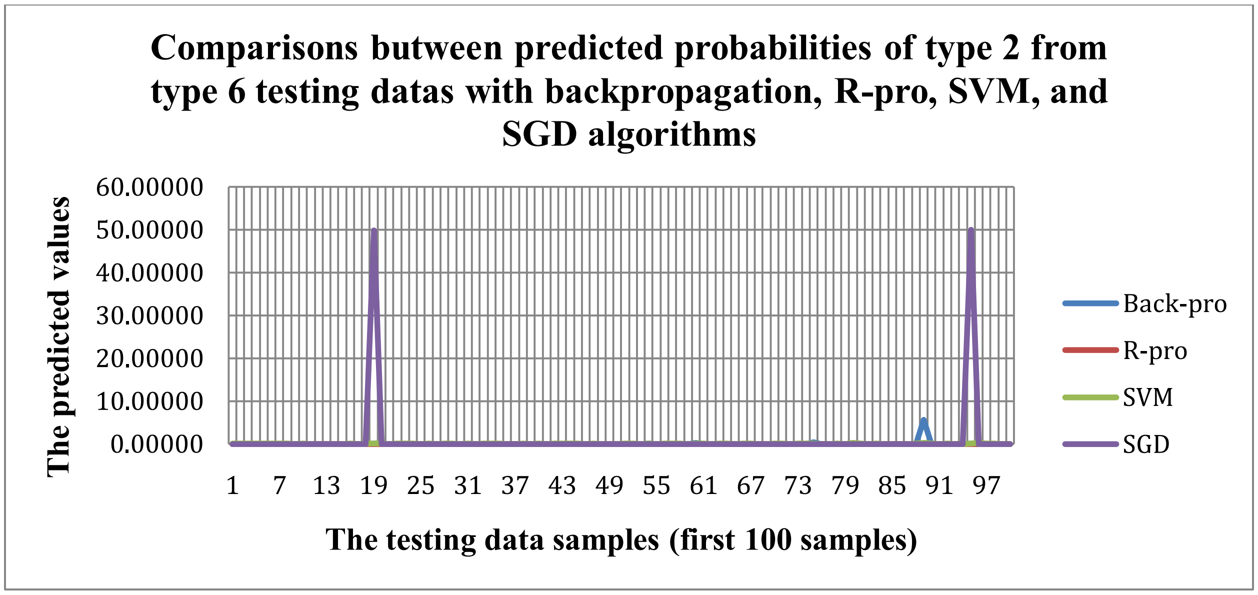


Figure 40. The comparison of type 2 outputs from type 6 testing dataset.

From figure 40, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 5.61 %. The predicted results for SVM were between 0.00 % and 0.21 %. The predicted results for SGD were between 0.00 % and 49.99 %.

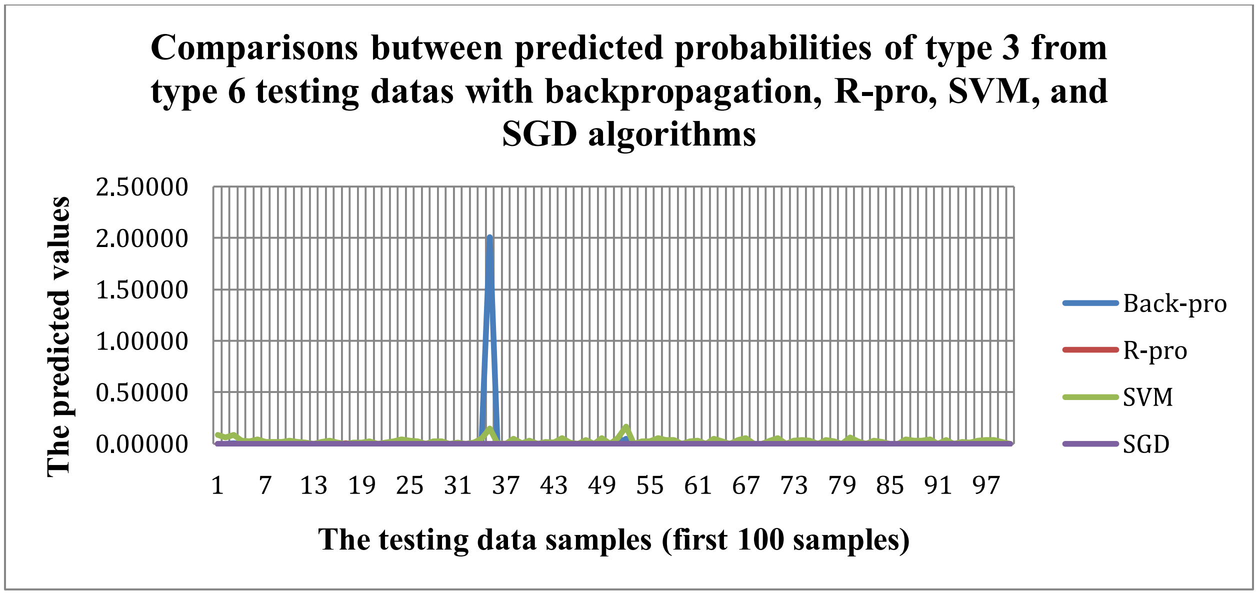


Figure 41. The comparison of type 3 outputs from type 6 testing dataset.

From figure 41, the predicted results for Rporp were between 0.00 % and 0.001 %. The predicted results for Backprop were between 0.00 % and 2.00 %. The predicted results for SVM were between 0.00 % and 0.16 %. The predicted results for SGD were all 0.00% in every sample.

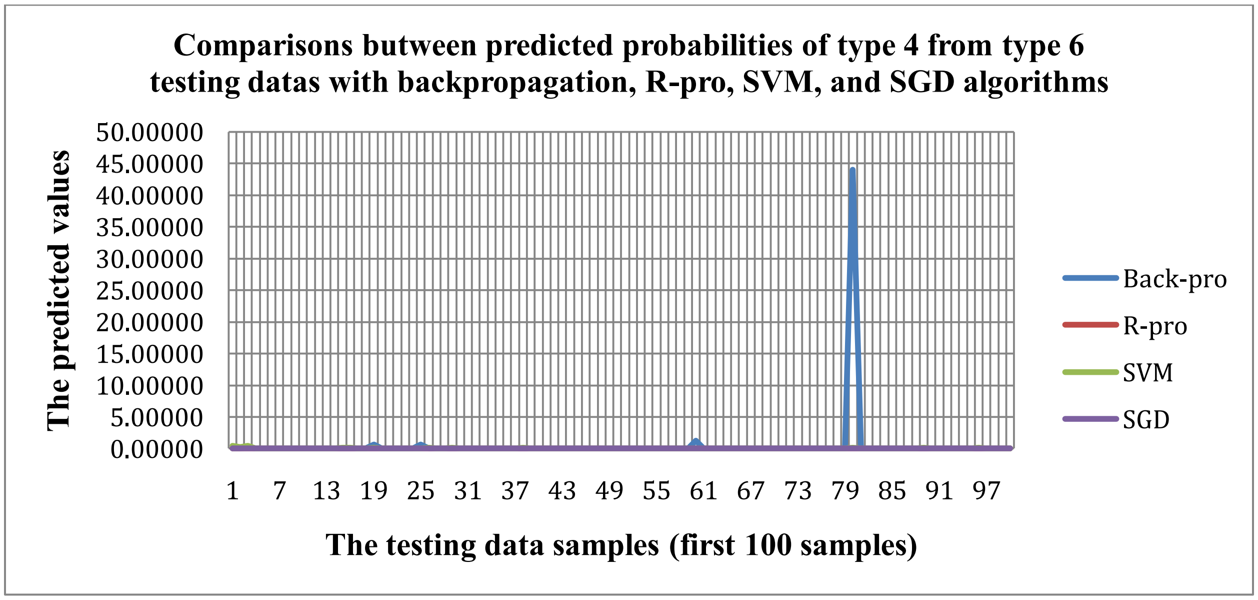


Figure 42. The comparison of type 4 outputs from type 6 testing dataset.

From figure 42, the predicted results for Rporp were between 0.00 % and 0.006 %. The predicted results for Backprop were between 0.00 % and 44.02 %. The predicted results for SVM were between 0.00 % and 0.45 %. The predicted results for SGD were all 0.00% in every sample.

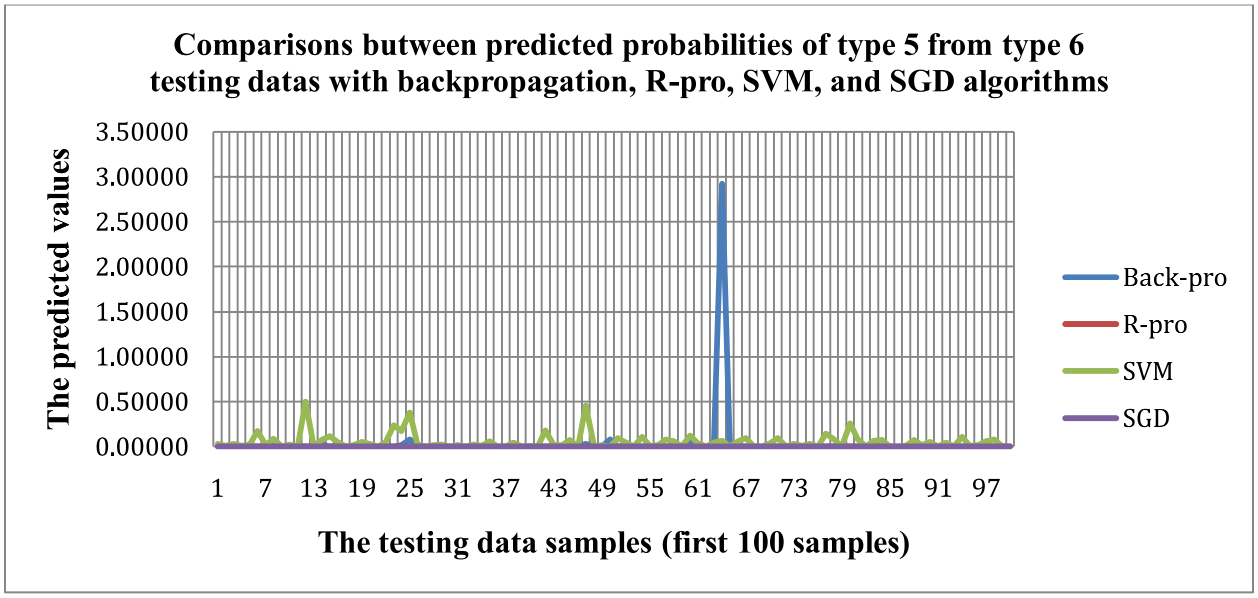


Figure 43. The comparison of type 5 outputs from type 6 testing dataset.

From figure 42, the predicted results for Rporp were all 0.00% in every sample. The predicted results for Backprop were between 0.00 % and 2.92 %. The predicted results for SVM were between 0.00 % and 0.49 %. The predicted results for SGD were all 0.00% in every sample.

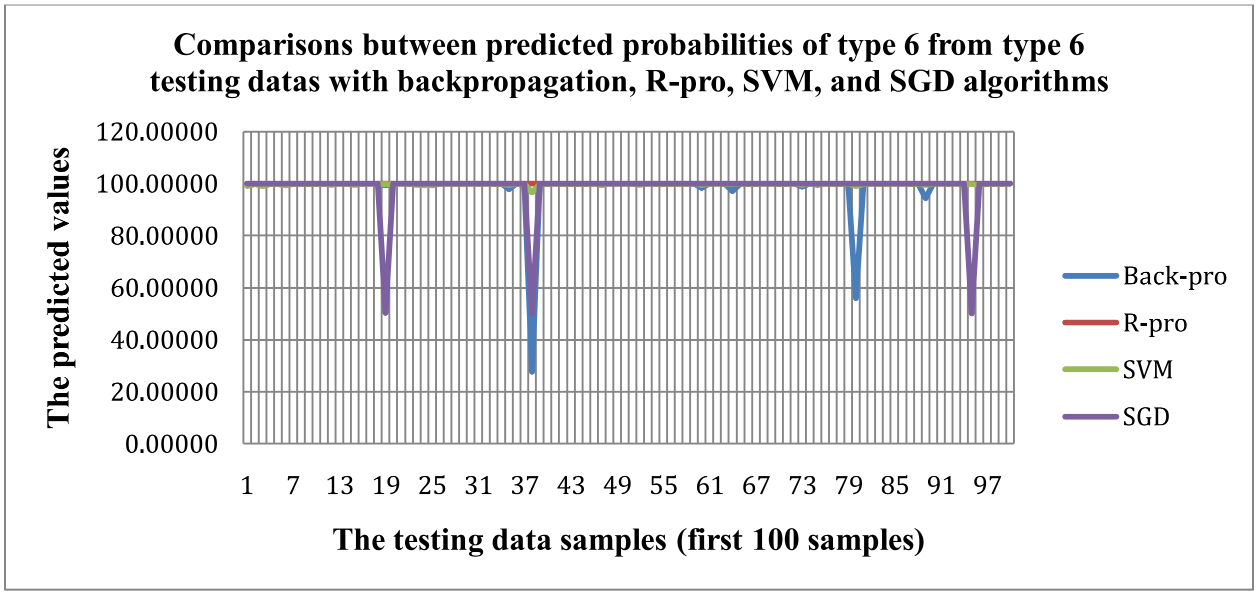


Figure 44. The comparison of type 6 outputs from type 6 testing dataset.

From figure 44, the predicted results for Rporp were between 99.99 % and 100.00 %. The predicted results for Backprop were between 99.99 % and 100.00 %. The predicted results for SVM were between 96.59 % and 100.00 %. The predicted results for SGD were between 50.00 % and 100.00 %.

The Total Predicted Results of the Machines by Using Testing Datasets as Inputs

Type - 01 Testing Datasets

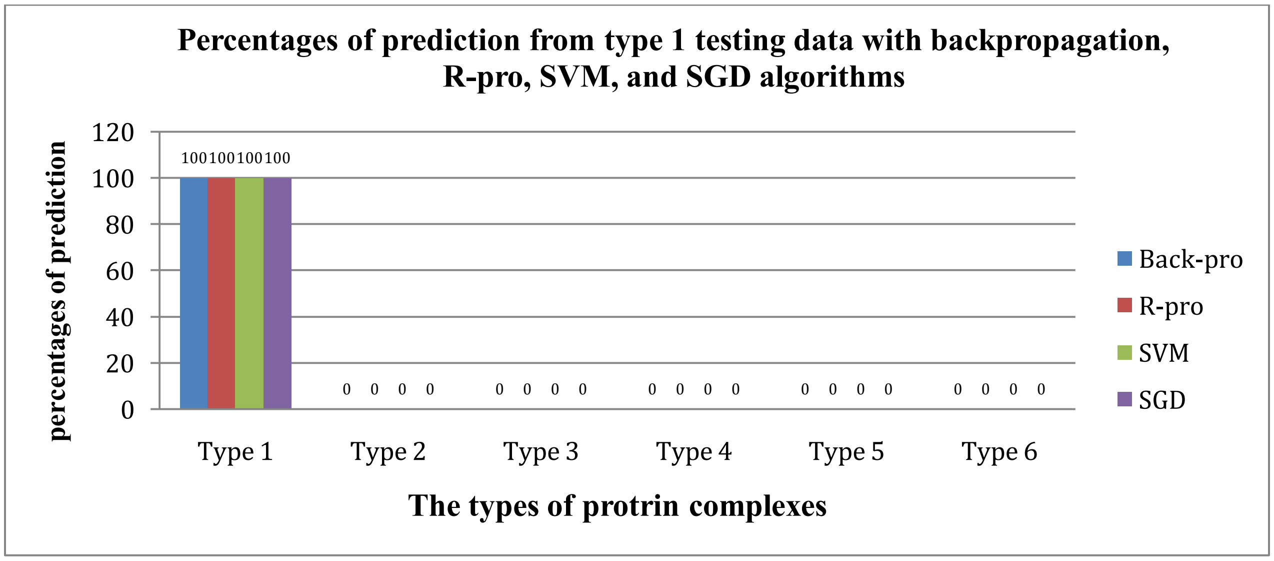


Figure 45. The percentages of the predictions from type 1 testing dataset.

From figure 45, the total results of using type 01 protein complex testing dataset on the learning machines are showed in this chart. All of the learning machines predicted 100% that the protein complex target was type 01.

Type - 02 Testing Datasets

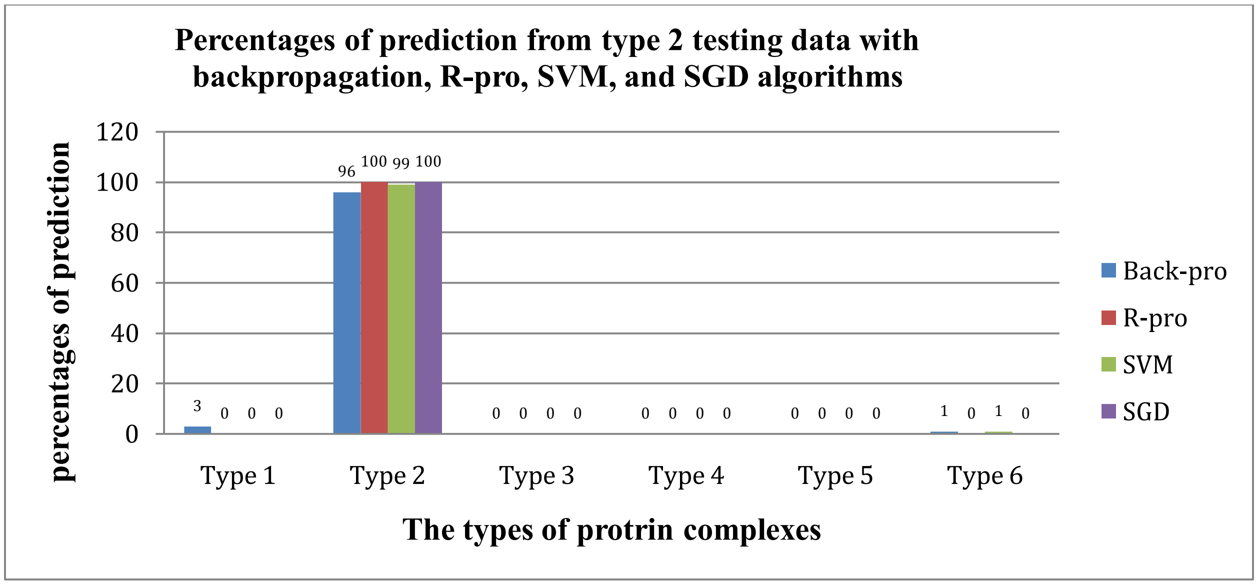


Figure 46. The percentages of the predictions from type 2 testing dataset.

From figure 46, the total results of using type 02 protein complex testing dataset on the learning machines are showed in this chart. The Rprop and SGD predicted 100% that the protein complex target was type 02. The Backprop predicted 96% for type 02, 3% for type 01, and 1% for type 06. The SVM predicted 99% for type 02 and 1% for type 06.

Type - 03 Testing Datasets

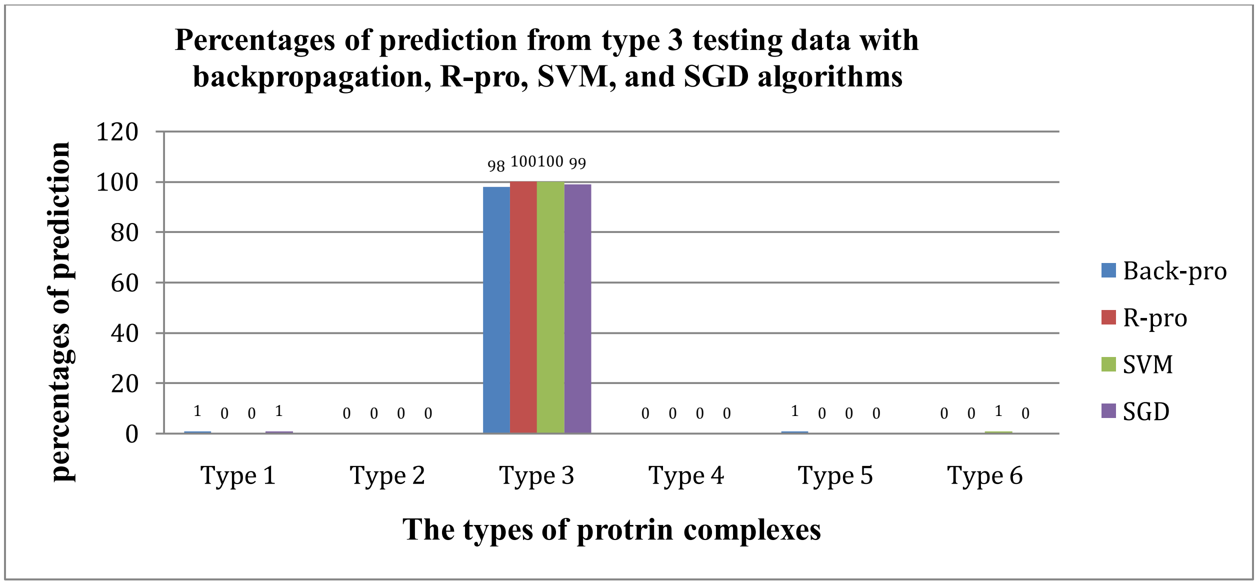


Figure 47. The percentages of the predictions from type 3 testing dataset.

From figure 47, the total results of using type 03 protein complex testing dataset on the learning machines are showed in this chart. The Rprop and SVM predicted 100% that the protein complex target was type 03. The Backprop predicted 98% for type 03, 1% for type 01, and 1% for type 05. The SGD predicted 99% for type 03 and 1% for type 01.

Type – 04 Testing Datasets

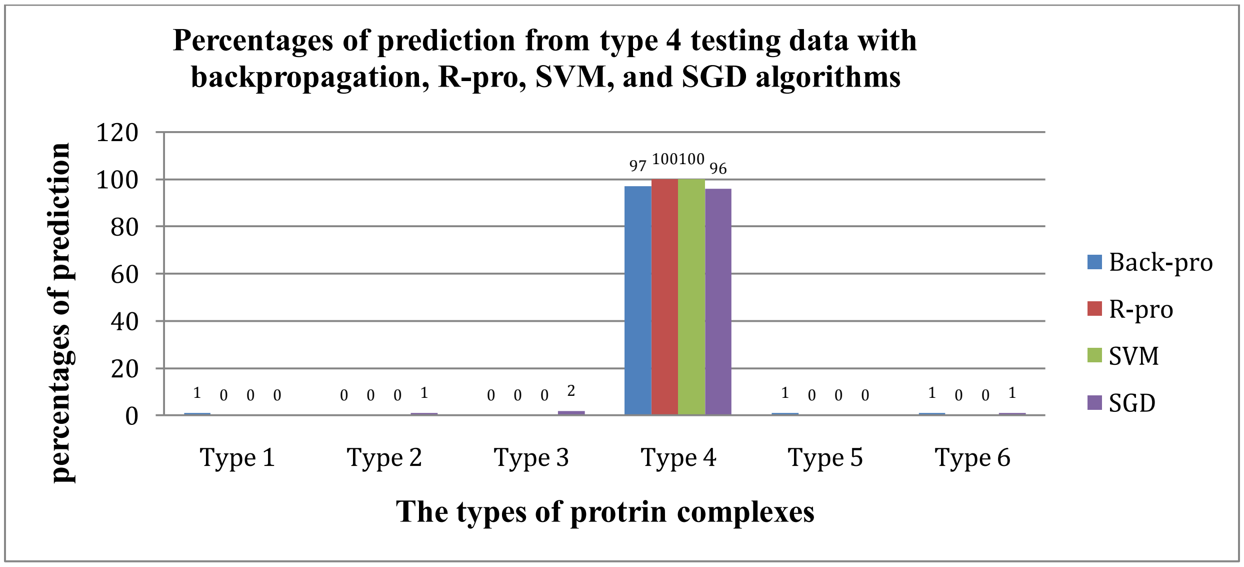


Figure 48. The percentages of the predictions from type 4 testing dataset.

From figure 48, the total results of using type 04 protein complex testing dataset on the learning machines are showed in this chart. The Rprop and SVM predicted 100% that the protein complex target was type 04. The Backprop predicted 97% for type 04, 1% for type 01, 1% for type 05, and 1% for type 06. The SGD predicted 96% for type 04, 1% for type 01, 2% for type 03, and 1% for type 06.

Type - 05 Testing Datasets

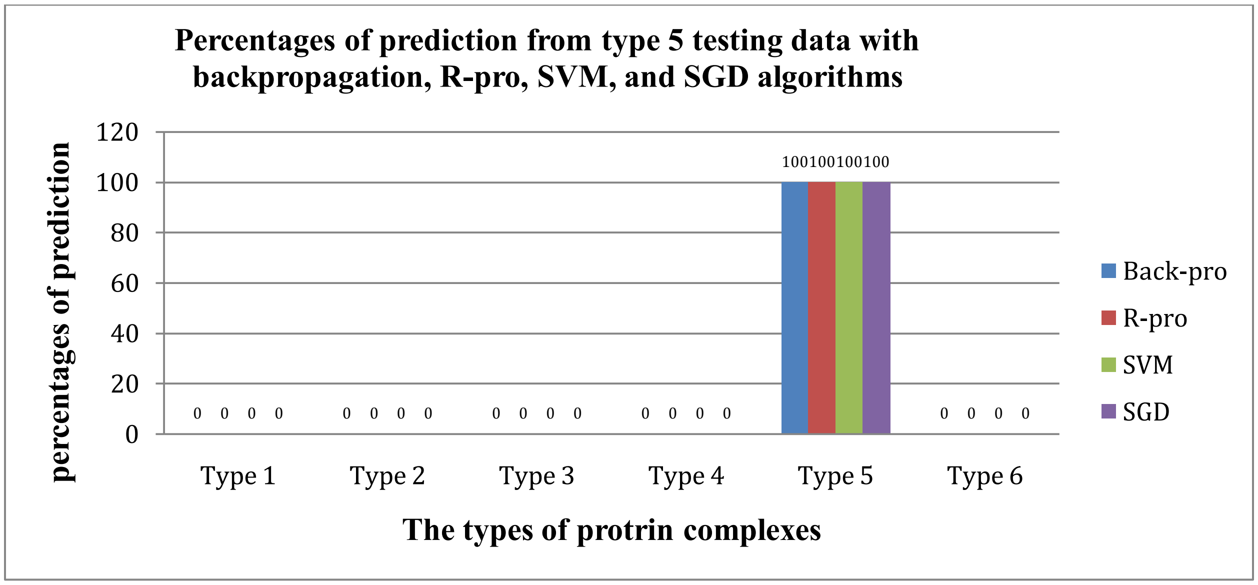


Figure 49. The percentages of the predictions from type 5 testing dataset.

From figure 49, the total results of using type 05 protein complex testing dataset on the learning machines are showed in this chart. All of the learning machines predicted 100% that the protein complex target was type 05.

Type - 06 Testing Datasets

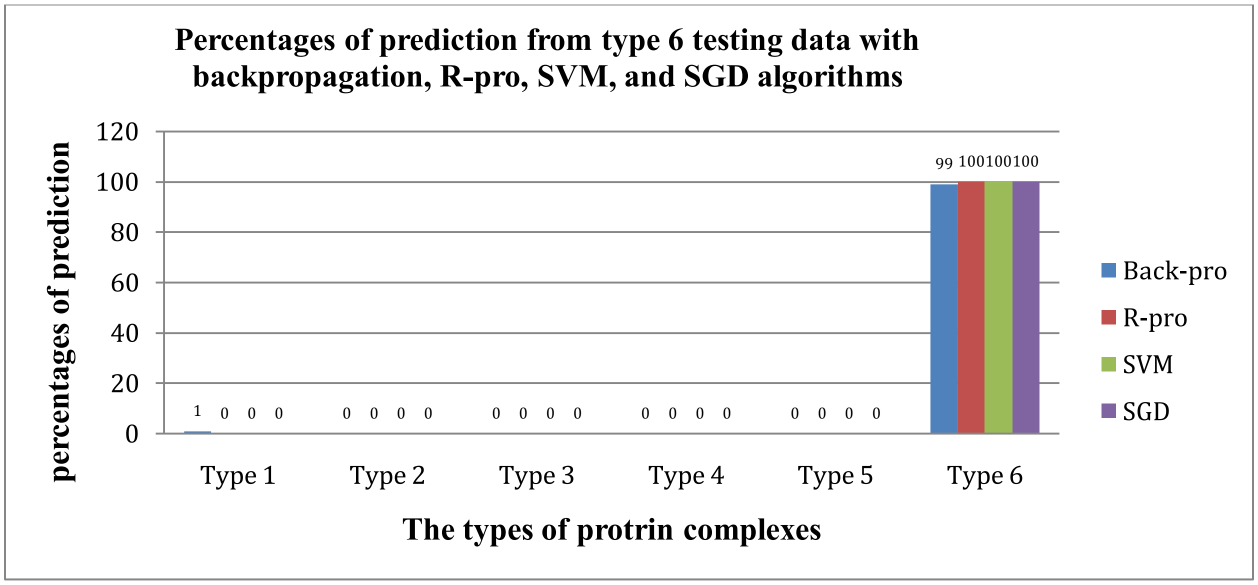


Figure 50. The percentages of the predictions from type 6 testing dataset.

From figure 50, the total results of using type 06 protein complex testing dataset on the learning machines are showed in this chart. The Rprop, SVM, and SGD predicted 100% that the protein complex target was type 06. The Backprop predicted 99% for type 06, and 1% for type 01.

The Total Accuracy Results of the Machines by Using Testing Datasets as Inputs

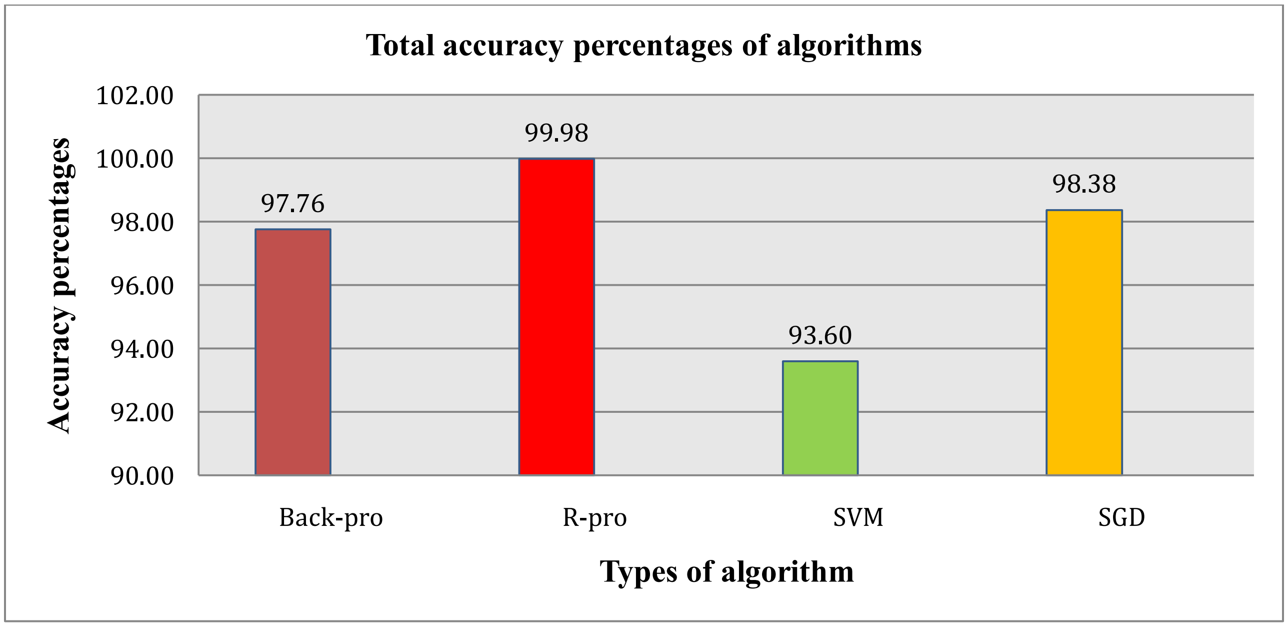


Figure 51. The accuracy of the machine models.

From figure 51, the total accuracy results of the models, by using testing datasets as inputs, are showed in the chart. The model that yielded the best accuracy was the R-pro model with 99.89% accuracy. The second was the SGD model with 98.38% accuracy. The third was the Back-pro model with 97.76% accuracy. The last one was the SVM model with 93.60% accuracy.

The Learning Rates of the Machines by Using Learning Datasets as Inputs

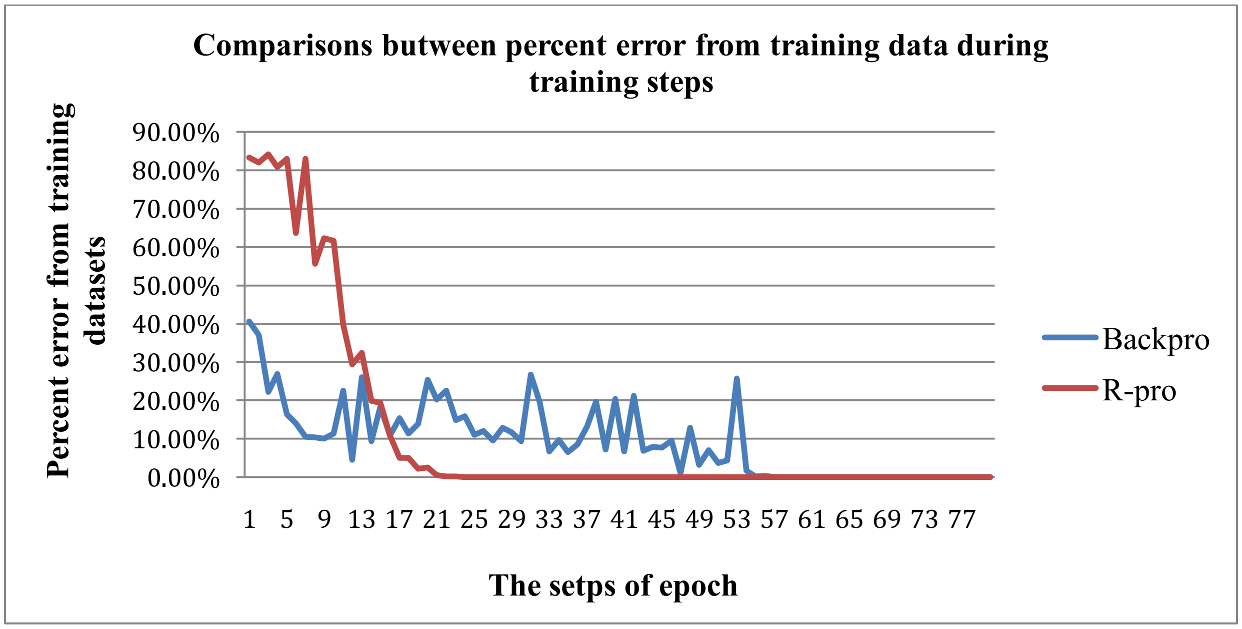


Figure 52. The comparison of the learning rates.

From figure 52, the learning rates of the machines, by using learning datasets as inputs, are shown in the chart. Only PyBrain library outputted the total percentage error value that means the comparison could conduct only on Backprop model and R-prop model. The result showed that the R-prop model had a faster learning rate than the Backprop model.

Chapter 5

CONCLUSION AND RESEACH WORK

This study began as a classification study of protein complexes. Over the course of two semesters, the project morphed into classification using the machines learning, and then finally performance comparisons of different machine learning models. Several problems and questions were posed as the designs of machines models began, and the types of supervised learning algorithms chosen. The problems partly directed the types of supervised learning algorithms as did the requirements of the overall study. Though, it was commonly the case with performance comparisons, there is always room for improvement and desire for additional study.

Conclusions

The performances of the machines were tested by used testing datasets as the input of the machines. From the results, the Rprop machine model was performed perfectly on every testing dataset. It gave the predictions of protein complex targets correctly from the protein complex data pool. The backprop machine model gave the perfect predictions on type 01 and type 05 testing datasets. On type 02, type 03, type 04, and type 06, the backprop machine model miss predicted the protein complex targets from the protein complex data pool. The SVM machine model gave the 100% predictions on every type of testing datasets, but the model miss classified the protein complex targets from type 02 testing datasets. The SGD machine model provided complete classification on type 01, type 02, type 05, and type 06 testing datasets. On type 03 and type 04 testing datasets, the models miss classified protein complex targets from protein complex data pool.

The accuracies of the machines for identification and prediction protein complex targets from the protein complexes data pool were calculated from the outputs of the previous process. From the result, the machine that has the best accuracy was Rprop machine model with 99.98 % accuracy. The second was SGD machine model with 98.38 % accuracy. The third was Backprop machine model with 97.76 % accuracy. The machine that had the worst accuracy was SVM machine model with 93.60 % accuracy.

The last question posed what the learning curves of the machine models during the training session. The parameter that used to create the learning curves to address this question was a total error. At the end of every iteration for training the machines, the training dataset was used as an input to perform classification of protein complex targets. The result of the classification was compared with the protein complex targets that already listed in the training dataset, and then the result of the comparison was used to calculate the total error. The Scilkit-learn library did not give the total error while the machines were trained, but the PyBrain library gave the total error while the machines were trained. From this reason, the comparison was performed only on the Brackprop machine model and the Rprop machine model. From the result, the Rprop learning rate was faster than the Backprop learning rate that means the Rprop machine model could learn faster than the Backprop machine model.

Recommendation for Further Research

At the top of the list for further research, the types of the protein complexes need to be added more into the sample pool. Increasing on the types of protein complexes should not affect the accuracies of the models. Another topic that can be explored more is the parameter for building the machines. In the creation of the machine model step, the condition of the parameters can be changed to test the model's performances or look for the parameter combination that performs better.

REFERENCES

Alpaydin, E. (2004). *Introduction to machine learning*. MIT press.

Andreas, B. G. (1960). *Experimental psychology*. Oxford, England: John Wiley.

Berg, J. (2007). *Biochemistry Jeremy M. Berg, John L. Tymoczko, Lubert Stryer* (6th ed.). New York: W.H.Freeman & Co.

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010* (pp. 177-186). Physica-Verlag HD.

Clancy, S. & Brown, W. (2008) Translation: DNA to mRNA to protein. *Nature Education* 1(1), 101.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*,*20*(3), 273-297.

Eisenach, P., Soeth, E., Röder, C., Klöppel, G., Tepel, J., Kalthoff, H., & Sipos, B. (2013). Dipeptidase 1 (DPEP1) is a marker for the transition from low-grade to high-grade intraepithelial neoplasia and an adverse prognostic factor in colorectal cancer. *British Journal Of Cancer, 109*(3), 694-703. doi:10.1038/bjc.2013.363

Enthought Canopy (Version1.3.0) [Software]. (2014). Software available at http://www.enthought.com.

Fausett, L. (1994). *Fundamentals of neural networks: Architectures, algorithms, and applications*. Englewood Cliffs, NJ: Prentice-Hall.

Gewirth, A. A., & Solomon, E. I. (1988). Electronic structure of plastocyanin: excited state spectral features. *Journal of the American Chemical Society*,*110*(12), 3811-3819.

Hartwell, L. H., & Hopfield, J. J. (1999). From molecular to modular cells biology. *Nature, 402*(6761), C47.

Hearst, M. A., Dumais, S. T., Osman, E., Platt, J., & Scholkopf, B. (1998). Support vector machines. *Intelligent Systems and their Applications, IEEE*,*13*(4), 18-2.

Hill, R. D., & Needham, J. (1970). *The Chemistry of life; eight lectures on the history of biochemistry, by Robert Hill [and others] Edited with an introduction by Joseph Needham*. Cambridge [Eng.] Univ. Press, 1970.

Kauzmann, W. (1956). Structural factors in protein denaturation. *Journal of Cellular and Comparative Physiology*, *47*(S1), 113-131.

Knerr, S., Personnaz, L., & Dreyfus, G. (1990). Single-layer learning revisited: a stepwise procedure for building and training a neural network. In*Neurocomputing* (pp. 41-50). Springer Berlin Heidelberg.

Lee, L., Leopold, J. L., & Frank, R. L. (2012). Protein secondary structure prediction using BLAST and exhaustive RT-RICO, the search for optimal segment length and threshold. In *Computational Intelligence in Bioinformatics and Computational Biology (CIBCB), 2012 IEEE Symposium on* (pp. 35-42). IEEE.

Long, X., Fu, S., Qi, Z., Yang, X., & Yu, Q. (2013). Digital image correlation using stochastic parallel-gradient-descent algorithm. *Experimental Mechanics*,*53*(4), 571-578.

Love, D. (2014). What The Heck Is Machine Learning? *BUDINESS INSIDER.* Retrieved October 18, 2014, from http://www.businessinsider.com/machine-learning-2014-5.

Ma, X., Wu, J., & Xue, X. (2013). Identification of DNA-Binding Proteins Using Support Vector Machine with Sequence Information. *Computational and mathematical methods in medicine*, *2013*.

Mccarty, R. (1992). A PLANT BIOCHEMIST'S VIEW OF H+-ATPases AND ATP SYNTHASES. *The Journal Of Experimental Biology, 172*(Pt 1), 431-441.

Mitchell, T. M. (1997). *Machine Learning / Tom M. Mitchell*. New York : McGraw-Hill, c1997.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *The Journal of Machine Learning Research*, *12*, 2825-2830.

Peltier, J. (2011). Clustered and Stacked Column and Bar Charts. *Peltier Tech Blog.* Retrieved October 11, 2014, from http://peltiertech.com/clustered-stacked-column-bar-charts/.

Perutz, M. F. (1965). Structure and function of haemoglobin: I. a tentative atomic model of horse oxyhaemoglobin. *Journal of molecular biology*, *13*(3), 646-IN2.

Qu, W., Yang, B., Jiang, W., & Wang, L. (2012). HYBP_PSSP: a hybrid back propagation method for predicting protein secondary structure. *Neural computing and applications*, *21*(2), 337-349.

Raven, P., & Evert, R. (1999). *Biology of plants* (6th ed.). New York: W.H. Freeman.

Riedmiller, M., & Braun, H. (1992). RPROP-A fast adaptive learning algorithm. In *Proc. of ISCIS VII), Universitat*.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1988). Learning representations by back-propagating errors. *Cognitive modeling*.

Schaul, T., Bayer, J., Wierstra, D., Sun, Y., Felder, M., Sehnke, F., ... & Schmidhuber, J. (2010). PyBrain. *The Journal of Machine Learning Research*,*11*, 743-746.

Scheuring, S. (2006). AFM studies of the supramolecular assembly of bacterial photosynthetic core-complexes. *Current Opinion In Chemical Biology, 10*(5), 387-393. doi:10.1016/j.cbpa.2006.08.007.

Shiblee, M., Chandra, B., & Kalra, P. K. (2010). Learning of geometric mean neuron model using resilient propagation algorithm. *Expert Systems with Applications*, *37*(12), 7449-7455.

Singer, E., & Magazine, Q. (2013). Biology's Big Problem: There's Too Much Data to Handle. *WIRED*. Retrieved October 17, 2014, from http://www.wired.com/2013/10/big-data-biology/all/.

Van Rossum, G. (1993). An introduction to Python for UNIX/C programmers.*Proc. of the NLUUG najaarsconferentie. Dutch UNIX users group*.

Watson, J. D., & Crick, F. H. C. (1953). A structure for deoxyribose nucleic acid. *Nature*, *421*(6921), 397-3988.

VITA

Wuthiwat Ruangchai received his Bachelor of Science degree (Biology) in Genetics from Kasetsart University – Thailand in 2009. Wuthiwat is currently pursuing his masters at Texas A&M University-Commerce, and he is going to graduate with a master's degree in Computational Science in December 2014. He worked as a graduate assistant at the Texas A&M University-Commerce and help with JAVA programming classes.

Address: 2411 Bryan St. APT 104 COMMERCE, TX 75428

Email: R.Wuthiwat@gmail.com