THE UNIVERSITY OF THE SOUTH PACIFIC LIBRARY

Author Statement of Accessibility- Part 2- Permission for Internet Access

Name of Candidate : Shaveen Singh (S02009725)
Degree : Master of Science in Computer Science
Department/School : School of Computing, Information & Math Sciences
Institution/University : The University of the South Pacific
Thesis Title : Intelligent Student Progress Monitoring in an ELearning environment with Neural Network Trained with Memetic Algorithm
Date of completion of requirements for award : 19 - June - 2015

1. I authorise the University to make this thesis available on the Internet for access by USP authorised users. [Yes/No]

2. I authorise the University to make this thesis available on the Internet under the International digital theses project [Yes/No]

Signed: [Signature]
Date: 19 - June -2015

Contact Address
SCIMS, FSTE, USP
Ph: 3232256 Email: singh_sv@usp.ac.fj

Permanent Address
Lot 19 Chanik Place, Caubati, Suva
Ph: 9953468 E: shaveen.singh@gmail.com

Feb2005
INTELLIGENT STUDENT PROGRESS MONITORING IN AN E-LEARNING ENVIRONMENT USING NEURAL NETWORK TRAINED WITH MEMETIC ALGORITHM

by

SHAVEEN SINGH

A thesis submitted in fulfillment of the requirements for the degree of Master of Science

Copyright © 2014 by Shaveen Singh

School of Computing, Information and Mathematical Sciences
Faculty of Science, Technology and Environment
The University of the South Pacific

October, 2014
Declaration

Statement by Author

I hereby solemnly and sincerely declare that the work presented in this dissertation is to the best of my knowledge my original work, except for citations which have been duly acknowledged in the text. I further attest that the material presented herein has not been submitted previously, either in whole or in part for a degree at any academic institution.

…………………………

Shaveen Singh
Student ID: S02009725

October, 2014
Acknowledgement

I am indebted to my supervisor, Dr. Sunil Lal, who played an instrumental role in the transition of my programme of study from Postgraduate Diploma to Masters Degree. Thus, I would like to express my profound gratitude and heartfelt thanks for his guidance and encouragement throughout the project, and his patience and understanding during the prolonged write-up period.

I would also like to express a deep sense of gratitude to my family for their support and understanding, which has been a major source of motivation during my MSc candidature.

My sincere appreciation also goes out to the following course coordinators at USP: Professor Ansgar Fehnker, Dr. M.G.M. Khan, Dr. Ashehad Ali, Mr. Avinesh Prasad, Mr. Ronal Singh and Mr. Dinesh Kumar whose courses were identified for experimentation in this research and their cooperation in providing the requested course statistics and reports.

A special note of thanks to Mr. Varunesh Rao, Learning Systems Administrator, Centre of Flexible Learning (CFL) at USP for his assistance in acquiring the required data for this project.

Finally, I offer my gratitude to the FSTE Faculty Research Committee and the Learning & Teaching office for funding and providing the relevant approval for this research and the School of Computing, Information Sciences (SCIMS) for sponsoring my MSc candidature.

Thank you very much
Abstract

Massive Open Online Courses (MOOCs) and other forms of digital learning have been widely heralded as practices that will serve to radically revolutionize education. However, offering such massive and open courses brings about its challenge and the need for monitoring large number of students. So much learner produced data is accumulated in the e-learning platform in the form of logs, which opens up opportunities to develop learning systems which can adapt to students abilities.

This thesis explores the potential of modelling student behaviour in an e-learning environment (Moodle) using Machine Learning (ML) techniques, which could act as a personalised feedback mechanism for a students’ academic progression, and a means of courseware evaluation and maintenance. Eleven different offerings from six courses and four disciplines were experimented from School of Computing, Information and Mathematical Sciences (SCIMS) at University of the South Pacific (USP). Feature selection and attribute ranking technique was employed to identify courseware modules and user actions that had greater influence in accurately predicting student performance. The most suitable training profile for the Artificial Neural Network (ANN) model was determined for predicting ‘at-risk’ students around the faculty mid-semester reporting deadline (Week 8).

The ANN model was implemented with a hybrid memetic learning algorithm. The architecture and parameters of the ANN was optimized using a past course offering and the learning model generated was then validated with the subsequent offering of the same course. An evaluation of the proposed system presented high prediction accuracy for identifying “at-risk” students for a good mix of courses experimented. The system also proved more reliable than the current mid-semester “at-risk” reporting practise implemented at USP.
To ensure that the model was able to adapt to the evolving nature of e-learning environment and improve with the availability of new learner profiles, an appropriate adaptive online learning strategy was also suggested for re-training the ANN. The hybrid meta-heuristic demonstrated that the incremental learning approach was able to cope with changes in data, and delivered trained neural networks that guaranteed faster convergence and finer performance without significantly increasing the learning cost.
### Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANN</td>
<td>Artificial Neural Network</td>
</tr>
<tr>
<td>BP</td>
<td>Back Propagation</td>
</tr>
<tr>
<td>CS</td>
<td>Chi-Squared</td>
</tr>
<tr>
<td>EA</td>
<td>Evolutionary Algorithms</td>
</tr>
<tr>
<td>EDM</td>
<td>Educational Data Mining</td>
</tr>
<tr>
<td>FFNN</td>
<td>Feed Forward Neural Network</td>
</tr>
<tr>
<td>FN</td>
<td>False Negative</td>
</tr>
<tr>
<td>FP</td>
<td>False Positive</td>
</tr>
<tr>
<td>GA</td>
<td>Genetic Algorithm</td>
</tr>
<tr>
<td>IG</td>
<td>Information Gain</td>
</tr>
<tr>
<td>IGR</td>
<td>Information Gain Ratio</td>
</tr>
<tr>
<td>LMS</td>
<td>Learning Management System</td>
</tr>
<tr>
<td>MA</td>
<td>Memetic Algorithms</td>
</tr>
<tr>
<td>MC</td>
<td>Misclassification</td>
</tr>
<tr>
<td>MLP</td>
<td>Multi-layer Perceptron</td>
</tr>
<tr>
<td>Moodle</td>
<td>Modular Object-Oriented Dynamic Learning Environment</td>
</tr>
<tr>
<td>MV</td>
<td>Majority Voting (Scheme)</td>
</tr>
<tr>
<td>RBF</td>
<td>Radial Basis Function</td>
</tr>
<tr>
<td>RCGA</td>
<td>Real Coded Genetic Algorithm</td>
</tr>
<tr>
<td>RF</td>
<td>Relief-F</td>
</tr>
<tr>
<td>SA</td>
<td>Simulated Annealing</td>
</tr>
<tr>
<td>SU</td>
<td>Symmetrical Uncertainty</td>
</tr>
<tr>
<td>TN</td>
<td>True Negative</td>
</tr>
<tr>
<td>TP</td>
<td>True Positive</td>
</tr>
<tr>
<td>USP</td>
<td>The University of the South Pacific</td>
</tr>
</tbody>
</table>
# Table of Contents

Abstract ............................................................................................................................. iv  
Chapter 1 ............................................................................................................................ 1  
Introduction ....................................................................................................................... 1  
1.1 Premise ................................................................................................................ 1  
1.2 Motivation ........................................................................................................... 2  
1.3 Background of Machine Learning in Education ................................................. 3  
1.4 E-Learning at USP ............................................................................................... 6  
1.5 Research Goals .................................................................................................... 8  
1.6 Structure of the Thesis ......................................................................................... 9  
Chapter 2 .......................................................................................................................... 10  
Artificial Neural Networks ............................................................................................... 10  
2.1 Background ....................................................................................................... 10  
2.2 Strengths and limitations of Neural Network .................................................... 13  
2.3 Feed-forward Neural Network .......................................................................... 14  
2.4 Learning in Feed-Forward Networks ................................................................ 17  
2.5 Network Architecture ....................................................................................... 18  
2.6 Performance of ANN Model on Spam Dataset ................................................. 19  
2.7 Summary ........................................................................................................... 24  
Chapter 3 .......................................................................................................................... 25  
Memetic Algorithms ........................................................................................................ 25  
3.1 Underlying Evolutionary Framework for Memetic Algorithm ......................... 25  
3.2 Using EAs to train ANN ................................................................................... 26  
3.3 Forming the Hybrid Meta-heuristic ................................................................... 29  
3.4 Introducing Local Search within the EA framework ........................................ 31  
3.5 Proposed Memetic Framework for ANN training ............................................. 33  
3.6 Performance of Memetic Framework on Spam Classification.......................... 34  
3.7 Chapter Summary .............................................................................................. 38  
Chapter 4 .......................................................................................................................... 39  
Intelligent Student Monitoring in an e-Learning Environment........................................ 39  
4.1 The Student Dataset ........................................................................................... 40  
4.2 Data Preparation ............................................................................................... 43  
4.3 Evaluating Performance ..................................................................................... 45
List of Figures

Figure 2.1: Working principle of a neuron.................................................................11
Figure 2.2: Feed-forward neural networks (FFNN) with two hidden layers ..........12
Figure 2.3: Architecture of a Recurrent Network .......................................................12
Figure 2.4: The Feed Forward Neural Network for Spam detection .....................15
Figure 2.5: Graph showing the early stopping point ..............................................20
Figure 2.6: MSE graph of BP getting trapped local minima after 120 epochs .........22
Figure 2.7: MSE graph using GA .............................................................................23
Figure 3.1: The general procedure of an evolutionary algorithm .........................26
Figure 3.2: Representation of ANN as a chromosome of weights (w) and thresholds (thresh) ..............................................................................................................27
Figure 3.3: Evolutionary Algorithm - possible hybridization points .....................29
Figure 3.4: Pseudo-code of simulated annealing algorithm ...................................32
Figure 3.5: Proposed Memetic Algorithm .................................................................33
Figure 3.6: Relationship between number of hidden layers neurons and MSE ........36
Figure 3.7: MSE graph for the best selected ANN trained with GA and MA. ...........36
Figure 4.1: A screenshot of the Moodle log report for UU100 ...............................41
Figure 4.2: Feature Selection Process .....................................................................47
Figure 4.3: Majority Voting (MV) scheme to resolve ranking ...............................49
Figure 4.4: Average F1 Score and optimal elimination point for each course .........53
Figure 4.5: Performance impact of feature selection versus using all attributes ...55
Figure 4.6: Performance of ANN model trained on Week 8 vs Week 15 profile and evaluated on UU100 test data .................................................................58
Figure 4.7: Graph showing progressing monitoring potential of ML approach ......63
Figure 4.8: Types of drifts: circle represents instance, different colours represent different classes .................................................................64
Figure 4.9: Concept drift: Patterns of change using one-dimensional data ..........65
Figure 4.10: Learning techniques and mechanisms for updating the ANN model ....66
Figure 4.11: UU100 results for adaptive learning Scenario I, II and III .................69
Figure 4.12: IS222 results for adaptive learning Scenario I, II and III .....................70
Figure 4.13: IS323 results for adaptive learning Scenario I, II and III .....................71
Figure 5.1: SOM architecture with a 2D hexagonal grid .........................................74
Figure 5.2: U-Matrix with class labels at 5 different points during the semester ........75
Figure 5.3: Graph showing the accuracy of the unsupervised learning algorithm on MA101 S2- 2010 ........................................................................................................................................................................75
Figure 5.4: Proposed simple architecture for implementation .................................................................79
Figure A.1: Architecture of the ANN used to test the prediction problem ......................91
List of Tables

Table 2.1: Attributes of the Spambase dataset ................................................................. 15
Table 2.2: Decomposition of Spam Training and Test Data ............................................ 20
Table 2.3: GA Classification results (5 run averages) ..................................................... 23
Table 2.4: Algorithm comparison on Spambase dataset ................................................ 24
Table 3.1: MA Classification Results (5 run average) .................................................... 37
Table 3.2: Algorithm comparison on Spambase dataset ................................................ 37
Table 4.1: List of USP courses experimented ................................................................. 41
Table 4.2: List of tracking attributes for UU100 ............................................................ 42
Table 4.3: Dataset for the Supervised Learning .............................................................. 44
Table 4.4: Result of ranking methods on the six experimented courses ......................... 51
Table 4.5: Results of MV Scheme showing changing ranks in UU100 across offerings ...... 52
Table 4.6: Effectiveness of Feature Selection to identify at-risk students in Week 8 ...... 56
Table 4.7: Average F1-Score of training with Week 8 versus Week 15 profile .......... 58
Table 4.8: Average F1-Score of ANN model versus Course Coordinators Report ...... 60
Table 4.9: Types of concept drift common in e-learning scenario ................................. 65
Table 5.1: Rules extracted using Apriori-frequent algorithm ........................................ 77
Table A.1: ANN parameters .......................................................................................... 91
Table A.2: MA parameters that controlled the training process ..................................... 92
Table A.3: Specifications of computers used for conducting tests ............................... 93
List of Publication(s) from this thesis

The research articles and presentations listed below were inspired from the work presented in this dissertation:

**Journal Publication:**


**Refereed Conference Proceedings:**


Chapter 1

Introduction

1.1 Premise

Wider adoption of ICT over the past decade has provided organizations an opportunity to capture rich data trails and activity streams emanating from its enterprise systems and various technological deployments. Lying hidden and undiscovered within all this captured data is information, potentially useful information that can be used to explain various business phenomena, improve predictions and support organizational decision making. Basing decision on data and evidence is gaining wider adoption in many fields (Siemens & Long, 2011). The key emphasis of data-driven decision making framework is that, data itself is a point of or a path to value generation in organizations. Data is not simply the byproduct of interactions and activities within an organization but rather, recognized as a critical value layer for informed decision making.

The volume, variety and velocity of data coming into the organizations continue to reach unprecedented levels. Marissa Mayer of Google (Mayer, 2009) suggests that data today is defined by three elements:

1. **Speed** – the increasing availability of data in real time, making it possible to process and act on it instantaneously.
2. **Scale** – increase in computing power: Moore’s law (number of transistors on a circuit board will double roughly every two years) continues to hold true.
3. **Sensors** – availability of new types of data: data published by physical real world objects like sensors, smart grids and connected devices, that is, the Internet of Thing (IoT).
Taken together, these three elements create a situation in which existing data management and decision making approaches are not feasible. This increasing complexity of data calls for new and novel approaches to understand the patterns of value that exist within the data.

A number of domains are already capitalizing on this “data explosion” as an opportunity to adopt a data-driven decision-making approach and analytics to improve organizational output and productivity. Notable fields that are transforming around us include health care, with the shift from clinical practice to evidence-based medicine; insurance companies relying on predictive models to determine its high-risk customers; businesses employing data mining models to track customer preferences and spending habits, and government departments resorting to “business intelligence” to rationalize the basis for future spending initiatives (Siemens & Long, 2011).

1.2 Motivation

In higher education, large amounts of data about teaching and learning interactions are endlessly generated and universally available. However, the astonishing array of data captured in this domain has traditionally been under-utilized, often applied with substantial delays in analysis and feedback.

The University of the South Pacific (USP) adopted Moodle as an e-learning platform in 2006 to support students from 14 campuses dispersed across 12 island nations. Over the years, Moodle has become the primary platform for supporting teaching and learning in the institution and is currently used in approximately 300 courses each semester (Watanabe, et al., 2011). The e-learning platform supports thousands of concurrent student logins and tens of thousands of interactions, each day. Moodle logs every click that students make for navigational purposes, and has a modest log viewing system built into it. The log reports provide a wealth of educational data which can be mined to enable better and informed decisions/practices in supporting online learning.
Articulating and identifying patterns of use in e-learning systems could provide better understanding of how students undertake web-based learning and provide guidance for better organization of online learning activities (Romero, et al., 2008). Analysis of learner data may also provide insights into which students are at risk of dropping out, or need additional support to increase their success, and confidence in the learning process. It can provide learners with insight into their own learning habits and can give recommendations for improvements. It also helps identify the learning model that is suitable for a specific course.

1.3 Background of Machine Learning in Education

In the last decade, there has been a significant growth in the use of the ML techniques in the area of higher education. Functionalities such as: student performances (Saini, et al., 2005) (Minaei-bidgoli, et al., 2003) (Ibrahim & Rusli, 2007) (Oladokun, et al., 2008), student modelling (Wang & Mitrovic, 2002), mediation of student e-discussions (Abel, et al., 2008), student retention (Dekker, et al., 2009) (Kotsiantis, et al., 2004) and other data mining applications have been established and researched into, with reference to the e-learning systems.

The commonality in all the above research is that they all attempt to predict a student’s ability to succeed in a course. However, their prediction models are based on grades such as coursework scores, online quiz marks, score in pre-enrolment diagnostic exams, grades from past courses and in a few cases based on personal background or results in past courses. Very limited research has been published on using the behavioural trait of learners within the e-learning environment to predict performance (Romero & Ventura, 2010).

(Macfadyen & Dawson, 2010) demonstrated the use of student log data and regression modelling to generate a best-fit predictive model for a course by incorporating key variables such as total number of discussion messages posted, total number of mail
messages sent, and total number of assessments completed. Results show that the model could explain 30% of the variation in student final grade. Their logistical modelling approach achieved the classification accuracy of 81%. Dawson et. al (Dawson, et al., 2008) analysed student log from BlackBoard Vista (LMS) using SPSS to identify dominant tools and used learner dis-engagement statistics in such activities as an indicator of at-risk students. (Greenland, 2011) presented exploratory examination of LMS log data and reported the strong correlation between LMS interaction and student results. Similarly (Romero, et al., 2008) and (Merceron & Yacef, 2008) used association rules based on interaction log to generate prediction models to classify student performance.

It is however, worth noting that all these publications are presented as merely a “proof of concept” using statistical modelling or standard data-mining techniques. The accuracy of each model is reported based on training and testing data from a single offering using a limited set of attributes easily extractable from the LMS. The model has been optimized for a single offering with no reference on how it would perform in subsequent offerings.

The eLearning environment however is continuously evolving. Inclusion or removal of activities, changes in assessment weightings, change in activity focus, changes in coordinators and extension to activity deadlines are common alterations that can affect the validity of the model. The success of an e-learning predictor model in the “true sense” is its robustness and adaptability to perform in future offerings. This demands that the deduced models are more flexible, generalized, practical and sustainable solutions.

Machine Learning (ML) is a scientific discipline that is concerned with the design and development of algorithms that allow machines (computers) to make robust decisions or evolve behaviours based on collection of empirical data (Saini, et al., 2005). The empirical data can be collected from sensors or interaction events and analysed comparing existing databases. The major focus of machine learning research is to automatically learn to recognize complex patterns and make intelligent decisions based
on the data. The triumph of ML techniques is primarily on its ability to derive predictive patterns from a set of features extracted from the data (Chen, et al., 2000). Recent developments in machine learning approaches have successfully produced models capable of capturing online user behaviour in e-commerce systems (Song, et al., 2013) (Yang & Deng, 2014). A classifier such as Artificial Neural Network (ANN) can be applied to such a task for behaviour recognition, feature extraction, approximation, and prediction.

The artificial neural network (ANN) is chosen in this dissertation, as it is not new to such dynamical and non-stationary time series; and possesses the ability to handle and process voluminous data in real-time. Additionally, neural networks are more robust in the presence of noise or missing data compared to many other ML algorithms, making them one of the practical ML algorithms for real world implementation.

ANN consists of a number of very simple and highly interconnected processors, also called neurons, which is analogous to the biological neurons of the brain (Negnevitsky, 2005). Learning typically occurs by example through training, where the training algorithm relies on features such as, number of logins per semester, number of resources viewed, number of assignments submitted, number of quizzes passed, and the scores obtained in various online assessments. Using these features, the training algorithm iteratively adjusts the connection weights (synapses) of the internal neurons to form a student behaviour model. The ANN would then be able to produce a binary prediction value acting as an indicator of satisfactory or unsatisfactory performance at any given point in the semester. In the long term, it could forecast whether the aggregate achievement and online contribution is satisfactory enough to succeed in the course. Coordinators could easily identify underperformers and critical learning activities. Instructional designers can use this finding when designing new courses, improving the structure of existing courses, appropriately weighting assessment tasks and also use the data to provide guidance to students to improve their learning experience.
1.4  E-Learning at USP

The University of the South Pacific (USP) was set up in the South Pacific region in 1968 by its 12 member countries - Cook Islands, Fiji Islands, Kiribati, Marshall Islands, Nauru, Niue, Samoa, Solomon Islands, Tokelau, Tonga, Tuvalu and Vanuatu. A total of 14 campuses are spread over an area of 30 million square kilometres of the Pacific Ocean. Due to this geographical separation, the university is expected to take its products and services to the doorstep of each and every household in the USP region (University of the South Pacific, 2013). This has resulted in a pedagogical shift from traditional face-to-face or chalk-and-talk to more flexible learning modes of delivery. The print-based mode has been the preferred mode of delivery, but the low pass rates have prompted the educational practitioners to shift emphasis to blended and fully online modes (Watanabe, et al., 2011).

While, the blended and fully online modes introduced in USP are seen as being cost-effective, learner-centred, and flexible, a common problem faced is that some courses are ill designed or lack motivating cues thus student lose interest. As a consequence, many students fail to engage sufficiently in the coursework resulting in many of them failing the courses offered through these delivery modes. An example is ST131 (Introduction to Statistics), a first year service course, which is offered in both online and face-to-face mode every year. The course reported a 10% lower pass rate for its online offering compared to the face-to-face offering last year.

Moodle (LMS) has some reports aimed at monitoring students’ participation, but these are seldom used. The log reports show summarized student statistics but lacks features to help explain the contextual meaning of data. The data logged sometimes seem incomplete or isolated and thus, do not provide a holistic indicator of learner progression.

To allow for student monitoring, USP in its current academic regulation has mandated the course coordinators to prepare mid-semester reports on a semesterly basis. This report is to allow for the identification “at-risk” students at the mid-point of the semester
and the various remedial interventions undertaken. The due date of the report is week 8 making it a critical reporting point in the semester which runs for 14 teaching weeks. The mid-semester report is submitted to various stakeholders such as the respective Head of School (HoS), Faculty Learning and Teaching (L&T) office, Student Administrative Services (SAS) office, Campus Directors (CDs) and all the sponsors (donors and governments) for their respective interests (refer to Appendix B for more details). The report is largely based on snapshot of assessments (if any) carried out during the first half of the semester. These reports fail to capture the continuous student learning progression and therefore have inherent inaccuracies in precisely identifying at-risk student.

Therefore, there is a genuine need of an automated real-time system to monitor students’ performances throughout the semester, which provides the early warning flags and reports which online activities are the most effective learning tools.

The findings of this research also carry profound implications for universities such as USP, which heavily rely on e-learning system such as Moodle to support distance learning and teaching. In particular the contributions of this research has a bearing on objective 7.1 of the USP strategic plan 2013-2018 (University of the South Pacific, 2013) which calls for increase in online student support to reduce attrition and encourage asynchronous learning. Furthermore, objective 1.4 of the strategic plan calls for the transformation of USP’s pedagogy and curriculum and mandates for a significant increase in the number of online and flexible delivery undergraduate programmes and courses.

Insightful discoveries would also provide a deeper understanding of the learning process in our unique institutional setting and may contribute towards innovating and transforming the pedagogical approaches at USP.
1.5 Research Goals

The principle objective of this research is to investigate how accurately we can gauge the progress of students based on the history of interactions within the LMS. Being able to articulate and identify patterns of use could provide better understanding of how students undertake web-based learning and provide insight on the nature of online interaction that greatly aid the students in fulfilling the learning outcomes of the course. The research also attempts to discover which training point (week 8 or week 15) is most suitable for training the ANN model for predicting 'at-risk' students around the faculty mid-semester reporting deadline (week 8). Finally, the goal is to explore a life-long learning strategy that allows for prediction model to sustain itself in subsequent offerings as courses and online learning behaviours evolve over time.

Our aim can be realised by achieving the following specific objectives:

- To implement an optimized Artificial Neural Network (ANN) architecture with a hybrid memetic algorithm for the e-learning environment.
- To identify the most suitable features for predicting 'at-risk' students in real-time environment while the course is in progress.
- To evaluate and report on the accuracy of the ANN model in identifying under-performing students when compared with the:
  1. progress reports prepared by the course coordinators such as the Faculty Mid-Semester Report for identify under-performing students.
  2. students’ overall course performance.
- To identify the nature of interactions and online activities on Moodle which greatly aid students in fulfilling the learning outcomes of the course successfully.
- Investigate and report on the suitability of the above algorithm to be used as an “online” learning strategy for life-long learning.
Structure of the Thesis

This dissertation provides a demonstration of harnessing the LMS data to develop predictive models that can identify at-risk students to allow for more timely pedagogical interventions at course level and validate the various learning tools and activities, and their effectiveness within the course.

The chapters in this dissertation are organized as follows:

- **Chapter 1**: Outlines the premise, motivation, research goal and the structure of the thesis.

- **Chapter 2**: Introduces the background of machine learning, and particularly ANNs and its robustness for classification and prediction problems in online environments. The architecture, parameters and the challenges in the training process of ANN is further explained with the aid of a spam classification problem.

- **Chapter 3**: Presents the hybrid Memetic Algorithm to address the challenges of optimization and convergence of neural network weights identified in Chapter 2. The chapter explicitly discusses a generalized architecture, and the hybridization scheme used to improve the performance of the ANN for the spam dataset.

- **Chapter 4**: Documents the performance of the improved ANN architecture on USP’s e-Learning platform for monitoring student progression. An appropriate online learning strategy is also presented to sustain the prediction model across multiple course offerings.

- **Chapter 5**: Reports on some the insightful discoveries, challenges and opportunities arising from this research and how it could contribute towards innovating and transforming the university system and pedagogical approaches at USP.

- **Chapter 6**: Summarizes the work that has been done and provides pointers for further research.
Chapter 2

Artificial Neural Networks

University learning environment are complex systems with a plethora of interacting and interdependent agents such as teachers, instructional designers, students, technologies, and external systems. Modelling in these systems requires a revolutionary approach as small actions or events can have disproportionate and non-linear consequences.

This chapter outlines potential of Machine Learning (ML), and more specifically ANN, in such composite online environments. It further highlights the design consideration and the learning strategies that form the foundation of ANN implementation. The architecture, parameters and the challenges in the training process of ANN is further explained with the aid of a spam classification problem.

2.1 Background

Artificial Neural Networks (ANN) is an attempt at modelling the information processing capabilities of the brain. Neural networks are used to model relationships between input and output values in data and are also used to find patterns in data. The popularity of neural networks is based on their remarkable versatility, abilities to handle both binary and continuous data, and to produce good results in complex domains. They are capable of performing tasks that include classification, function approximation, prediction or forecasting, clustering or categorizations, optimization and control (Paliwal & Kumar, 2009). The basic building block of every artificial neural network is a neuron; a simple
mathematical model (function) comprising of three simple sets of operations: multiplication, summation and activation (Suzuki, 2011).

The neuron is a simple computing element that receives several signals from its input links (synapses), computes a new activation level and sends it as an output signal through the output links. The input signal can be raw data or outputs of other neurons. The output signal can be either a final solution to the problem or an input to other neurons. Figure 2.1 shows the operation of a typical neuron.

![Figure 2.1: Working principle of a neuron.](image)

The full potential and calculating power of these nodes come to life when we start to interconnect them into multi-layer perceptron networks (Figure. 2.2). These networks are based on the simple notion that complexity can be grown out of merely a few basic and simple rules. Neural networks learn by training on data using an algorithm that modifies the interconnection weights as directed by a learning objective for a particular application. The knowledge learnt is distributed over a set of trained networks weights.

Feed-forward neural networks (FFNN) are the most commonly used form of ANN, although many more sophisticated neural networks have been proposed. Feed-forward neural networks are also known as *Multi-Layer Perceptron* (MLP). In a FFNN, the
information moves only forward, from the input to the output nodes. Figure 2.2 shows an example of a feed-forward neural network with two hidden layers.

![Feed-forward neural networks (FFNN) with two hidden layers.](image)

**Figure 2.2: Feed-forward neural networks (FFNN) with two hidden layers.**

In contrast to FFNN, other architectures such as recurrent neural networks have feedback connections or loops (Figure 2.3). It thus acts like a dynamical system whose next state and output depends on the present network state and input; this makes them capable of modelling dynamical systems.

![Architecture of a Recurrent Network](image)

**Figure 2.3: Architecture of a Recurrent Network**
2.2 Strengths and limitations of Neural Network

It is important to recognize the various capabilities and limitations of the chosen prediction model before further venturing into this research. Neural networks present various strengths that make them suitable for regression analysis and prediction tasks for online domains. One of the main advantages of ANN is that they are universal function approximators. They can approximate arbitrary continuous functions to any degree of accuracy (Negnevitsky, 2005) (Suzuki, 2011) (Kondo, 2007). As a result, neural networks have the ability to efficiently map nonlinear relationships between their input and output.

Secondly, neural networks are data-driven instead of model-driven. This means that they do not assume an explicit relationship model among the data, but rather make their predictions based on the actual model that exists within the data. This concept is particularly useful as real-world problems are often non-linear and the relationships are difficult to describe analytically. The characteristics of ANN – arbitrary function approximation, non-linearity and generalization capability makes it more appropriate for such situations.

Additionally, neural networks are more robust in the presence of noise or missing data compared to many other ML algorithms making them one of the practical ML algorithms for real world implementation.

However, it must be noted that although neural networks are continuously used for solving a variety of problems, they still present some limitations. Neural networks are often classified as a black-box method and cannot be analysed or described in great detail unlike linear models or expert systems (Andrews, et al., 1995). In neural networks, knowledge is embedded in the entire network and cannot be broken into individual pieces. Any change of a synaptic weight in isolation may lead to unpredictable results.
Secondly, the training time of neural networks is significantly higher than linear methods. This is usually the case due to the tendency of the ANN to get trapped in local minima while minimizing the cost function during training (Manel, et al., 1999).

Furthermore, since neural networks are data-driven, its performance is only as good as the quality and diversity of the training set (Haykin, 1999). Knowledge in neural networks is stored as synaptic weights between neurons. This knowledge is obtained during the learning phase when a training set of data is presented to the network. The more indicative the examples of the problem they are presented with, the more accurate the predictions the ANNs are expected to make.

However, with improvements over the last decade in the processing capability of computers, the increased availability of data and the unification of ANNs with other advanced computing methods, it is expected that neural networks will continue to triumph as a powerful technique for classification and regression problems.

### 2.3 Feed-forward Neural Network

Feed-forward neural network (FFNN) is the ANN architecture employed in this dissertation. It consists of the input layer, one or more hidden layers of computation nodes and an output layer of computation nodes. The input signal propagates through the network in a forward direction, on a layer by layer basis. The computation of the function signal appearing at the output of the neuron can be expressed as a continuous non-linear function of the input signal and the synaptic weights associated with that neuron.

In evaluating the suitability of FFNN for the e-learning environment, we carried out preliminary investigations by benchmarking its performance on the publically available UCI Spambase dataset\(^1\). This side research therefore serves to illustrate the operations on FFNN, as well as highlight the need for better algorithms to train the FFNN, which we

---

\(^1\) [http://archive.ics.uci.edu/ml/datasets/Spambase](http://archive.ics.uci.edu/ml/datasets/Spambase)
have factored in our final design. The class distribution comprised of 1813 Spam messages (39.4%) and 2788 Ham messages (60.6%), totalling 4601 instances. The objective was to classify an email message as either spam or ham given the list of features. The attribute information of the dataset is provided in Table 2.1, and the corresponding FFNN for the Spam problem is shown in Figure 2.4.

Table 2.1: Attributes of the Spambase dataset

<table>
<thead>
<tr>
<th>Num. of Attributes</th>
<th>Data type</th>
<th>Range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>48</td>
<td>Real</td>
<td>[0,100]</td>
<td>Word frequency expressed as a percentage</td>
</tr>
<tr>
<td>6</td>
<td>Real</td>
<td>[0,100]</td>
<td>Char frequency expressed as a percentage</td>
</tr>
<tr>
<td>1</td>
<td>Real</td>
<td>[1,...]</td>
<td>Average length of uninterrupted sequences of capital letters</td>
</tr>
<tr>
<td>1</td>
<td>Integer</td>
<td>[1,...]</td>
<td>Average length of uninterrupted sequences of capital letters</td>
</tr>
<tr>
<td>1</td>
<td>Integer</td>
<td>[1,...]</td>
<td>Total number of capital letters in the e-mail</td>
</tr>
<tr>
<td>1</td>
<td>Nominal</td>
<td>{0,1}</td>
<td>Class attribute {0=Ham, 1=Spam}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>58</td>
<td>Total Attributes</td>
</tr>
</tbody>
</table>

Figure 2.4: The Feed Forward Neural Network for Spam detection
The standard model of the neuron comprises a set of input connections, a linear combiner and a transfer or activation function. For the spam classifier, the input for the neural network are the set of 57 input features as shown in Figure 2.4 (features indicate the frequency of certain keywords and consecutive characters in the email message). The weight $w_{ij}$ is defined as the connection between the input signal $x_j$ to neuron $i$. The linear combiner computes the weighted sum of input signals and adds a bias. The use of biases in a neural network increases the capacity of the network to solve problems by allowing the hyperplanes that separate individual classes to be offset for superior positioning. The dynamics of a feed-forward network is described by Equation (2.1), where the input activation value $y_i$ of a given neuron $i$ is given for $N$ input connections.

$$y_i = \sum_{j=1}^{N} (w_{ij}x_j) + \theta_i$$  \hspace{1cm} (2.1)$$

where $\theta_i$ represents the bias applied to neuron $i$. The transfer function $f(y_i)$ computes the output $a_i$ of the unit. Some common transfer functions are threshold, sigmoid, hyperbolic tangent and piece-wise linear functions. The sigmoid and threshold transfer function is used to classify the result as spam or ham. The two functions are outlined in Equation (2.2) and Equation (2.3).

$$f(y_i) = \frac{1}{1 + e^{-y_i}}$$ \hspace{1cm} (2.2)$$

$$f(y_i) = \begin{cases} 1, & \text{if } y_i \geq k \\ 0, & \text{if } y_i < k \end{cases}$$ \hspace{1cm} (2.3)$$

where $k$ is the classification threshold in the range $[0,1]$. To emulate this in a computer program, the inputs $x_j$ and weights $w_{ij}$ where $j = 1 \ldots N$, can be implemented as input vector $x = [x_0, x_1, \ldots, x_N]^t$ and weight vector $w_i = [w_{i0}, w_{i1}, \ldots, w_{iN}]^t$ respectively. Thus, the activation value for the $i^{th}$ neuron can be computed as $y_i = w_i \cdot x$ and output as $a_i = f(w_i \cdot x)$.
2.4 Learning in Feed-Forward Networks

The process of learning in ANNs involves adaptive mechanisms that enable it to learn from experience, learn by example and learn by analogy. Learning capabilities can improve the performance of an intelligent system over time.

The spam classifier employs a supervised learning paradigm whereby after being exposed to a sufficient number of samples, the ANN can generalize new instances it may have not yet encountered. This is done by repeated adjustments in the weights of the network to reduce the difference between the actual and desired outputs of the perceptron during the training process.

(Rumelhart, et al., 1986) introduced the Back propagation (BP) algorithm which employs gradient descent for training feedforward networks. The algorithm requires the transfer function that governs the neurons to be differentiable. The Back-propagation algorithm adjusts the connection weights in the neural network in a two-phase process which consists of a forward and a backward pass. The input signals \( [x_1, x_2, \ldots, x_n] \), are propagated through the network in the forward direction, and error signals, \( [e_1, e_2, \ldots, e_n] \), are calculated using an objective function and passed from in the reverse/backward direction. The objective function is usually to minimize the mean square error (MSE) and is calculated for the respective neuron in the output layer as:

\[
MSE = \frac{1}{N} \sum_{i=1}^{N} (y_{d,i} - y_i)^2
\]  

where \( N \) is the number of training instances and \( y_{d,i} \) and \( y_i \) are the desired and actual output for the \( i^{th} \) pattern. The error gradient \( \frac{\delta E}{\delta w} \) for each weight \( w \) is determined as the derivative of the activation function. It is propagated starting from the output layer and passed backwards to the hidden layer analogues to the way activations are propagated forward. This scheme is also referred to as the delta rule. This is done for all instances in the training data at the completion of each epoch. This is an iterative process which
ideally terminates when the error signal converges to a minimum. In fact, one of the limitations of this method is convergence to poor local optima and their dependence on the initial solution. In addition, they are not applicable to network models that use transfer functions for which derivatives cannot be calculated.

Variations of BP algorithm have also been introduced (Gauss–Newton, Levenberg–Marquardt, QPROP, RPROP) but have faced similar optimization issues (Yu & Wilamowski, 2011). The more recent global optimization methods, especially evolutionary approaches, have generally obtained better results than gradient descent based methods (Han, et al., 2013) (Vui, et al., 2013). Early successes of Genetic Algorithms to train ANNs have been highlighted in the works of (Montana & Davis, 1989), (Schaffer, Whitley, & Eshelman, 1992) and (Dorsey, et al., 1994).

Genetic Algorithm (GA) is a class of stochastic search algorithm developed by (Holland, 1992) inspired by the Neo Darwinism process of reproduction (crossover), mutation, competition and selection. Given a quality function to be maximized, a set of random candidate solutions are created. Based on their relative fitness, the algorithm proceeds to perform re-combination on two or more suitable selected chromosomes (parents). This results in one or more new candidate solutions. Random genes in the newly created chromosomes are mutated based on the mutation probability. Through subsequent generations the quality of the candidate solution should increase as less fit chromosomes are replaced with better-suited ones.

2.5 Network Architecture

Apart from the learning algorithms, the architecture of the neural network significantly affects the generalization ability of the neural network. A poor choice of the network architecture can result in poor generalization even with optimal weight selection (Andersen & Martinez, 1999).
Although MLPs can be a universal approximator with a sufficiently large number of hidden nodes, having excessive number of nodes in the hidden layer may endanger the network to begin memorizing the input-output relationships, which may lead to over-fitting (Paliwal & Kumar, 2009). Such a network may fit training data points well but may not be able to successfully interpolate and extrapolate testing data points. On the contrary, having too few neurons in the hidden layer will result in under-fitting. The initial configuration (rule of thumb) generally chosen for the neural network, in each experiment when commencing the optimization process is:

\[ N = \frac{(m + n)}{2} \]  \hspace{1cm} (2.5)

, where \( m \) and \( n \) are the number of input and output nodes of the ANN respectively.

Over-fitting has been a central problem in MLPs, and much work has been devoted to preventing over-fitting. Some popular techniques to improve network generalization include early stopping, weight decay, and pruning. Literature also suggests that the case of over-fitting may be improved by increasing the number of instances in the training set (Yanbo, 2009) for smaller dataset.

### 2.6 Performance of ANN Model on Spam Dataset

Spam classification holds some resemblance with student classification problem in a sense that both are two-class classification problems originating in the online context. Also, the features of spam are complex and continuously evolve overtime very much like student learning profiles in the online environment, making it a challenging yet worthwhile system to model (Dimitrakakis & Bengio, 2005).

For the training and testing the classifier, the spambase corpus has to be divided into training and test sets. The proposition in most literature is to have 80:20 split for training and testing data (Andersen & Martinez, 1999).
For spam data set, the 4601 instances corpus is sufficiently large and can be split in the 80:20 ratio. Table 2.2 shows the final decomposition of the training and testing set that we will use in our future experimentations. The two sets have been randomly selected and the same data has been maintained for all experimentation to ensure consistency. The original (39.4%) to ham (60.6%) composition ratio has also been retained.

### Table 2.2: Decomposition of Spam Training and Test Data

<table>
<thead>
<tr>
<th>Description</th>
<th>Ratio</th>
<th>Total Num. of Instances</th>
<th>Num. of Spam</th>
<th>Num. of Ham</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training Set</td>
<td>80%</td>
<td>3680</td>
<td>1450</td>
<td>2230</td>
</tr>
<tr>
<td>Testing Set</td>
<td>20%</td>
<td>921</td>
<td>363</td>
<td>558</td>
</tr>
<tr>
<td>Total Corpus</td>
<td>100%</td>
<td>4601</td>
<td>1813</td>
<td>2788</td>
</tr>
</tbody>
</table>

#### 2.6.1 Stopping the learning process

The training algorithm implements the early stopping criterion in this research which also assists in addressing the over-fitting problem common to the ANN training.

It embodies a validation approach, whereby a randomly selected non-overlapping subset of the training data set is used to validate and terminate the training process. The training process is terminated once the minimum training error value is reached or the validation error begins to increase in consecutive epochs (Figure. 2.5).

![Figure 2.5: Graph showing the early stopping point](image.png)
2.6.2 Evaluating Performance

The performance of the ANN based spam filter in the experiment is evaluated on precision and recall measure, in line with other research and publications on the spam dataset. These measures are computed as follows:

\[
\text{Spam Precision (SP)} = \frac{N_{SS}}{N_{SS} + N_{LS}} \quad (4.1)
\]

\[
\text{Legitimate Precision (LP)} = \frac{N_{LL}}{N_{LL} + N_{SL}} \quad (4.2)
\]

\[
\text{Spam Recall (SR)} = \frac{N_{SS}}{N_{SS} + N_{SL}} \quad (4.3)
\]

\[
\text{Legitimate Recall (LR)} = \frac{N_{LL}}{N_{LL} + N_{LS}} \quad (4.4)
\]

where:

- \(N_{SS}\) = the number of spam messages correctly classified as spam.
- \(N_{SL}\) = the number of spam messages incorrectly classified as legitimate.
- \(N_{LL}\) = the number of legitimate messages correctly classified as legitimate.
- \(N_{LS}\) = the number of legitimate messages incorrectly classified as spam.

2.6.3 Learning with Back Propagation Algorithm

The ANN requires a learning procedure to optimize its weights and thresholds. The gradient based Back Propagation (BP) learning approach is employed first. A series of experiments with varying parameters (learning rate \(\{0.1, 0.3, \text{and } 0.5\}\), momentum \(\{0.2, 0.5, \text{and } 0.9\}\) and hidden layers comprising of \(\{24, 29, \text{and } 34\}\) neurons is conducted. Each training run cycle comprised of 1000 epochs. The experiment is repeated five times for each training scenarios and the five-run average is considered for analysis.

Figure 2.6 suggests that the performance of BP as a trainer does not seem effective in reducing the mean squared error (MSE). The spam class obtained the precision accuracy of 0% and legitimate (ham) class got the precision accuracy of 100%. This suggests that
no effective training took place. This may be very likely due to the fact that BP is a gradient based search method and got trapped in the local minima in the very early stage of the training process. Introducing higher learning rate and/or momentum also does not bring any improvement to the learning process except to have a lower initial MSE for the training set. Figure 2.6 shows the convergence graph of BP algorithm with 29 neurons in the hidden layer, learning rate $\alpha = 0.3$ and momentum $\beta = 0.2$.

![BP Convergence](image)

**Figure 2.6: MSE graph of BP getting trapped local minima after 120 epochs**

### 2.6.4 Learning with Genetic Algorithm

The parameters for the GA were set as follows:

- Population Size = 50,
- Selection Ratio = 10%,
- Mutation Rate = 4%,
- Crossover rate = 0.5.

Figure 2.7 shows more promising result obtained with GA with progressive reduction in MSE. This continuous reduction in the MSE shows that learning is taking place.
2.6.5 Improving the ANN architecture

With the promise of convergence shown by GA, we further investigated the possibility of improving the classifier by varying the number of the neurons in the hidden layer starting with the rule of thumb of 57 input, 29 hidden, and 1 output layer neurons.

The result of this exploration in terms of its classification accuracy is captured in Table 2.3.

Table 2.3: GA Classification results (5 run averages)

<table>
<thead>
<tr>
<th>Neurons in Hidden Layer</th>
<th>Spam Precision</th>
<th>Legitimate Precision</th>
<th>Spam Recall</th>
<th>Legitimate Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>19</td>
<td>91.06%</td>
<td>93.43%</td>
<td>89.81%</td>
<td>94.27%</td>
</tr>
<tr>
<td>24*</td>
<td>92.31%</td>
<td>93.16%</td>
<td>89.26%</td>
<td>95.16%</td>
</tr>
<tr>
<td>29</td>
<td>89.86%</td>
<td>93.71%</td>
<td>90.06%</td>
<td>93.37%</td>
</tr>
<tr>
<td>34</td>
<td>90.17%</td>
<td>93.30%</td>
<td>89.67%</td>
<td>93.64%</td>
</tr>
</tbody>
</table>

TESTING DATA SUMMARY (24 Neurons)

<table>
<thead>
<tr>
<th></th>
<th>Total Instances</th>
<th>Correctly Classified</th>
<th>Incorrectly Classified</th>
<th>Classification Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spam</td>
<td>363</td>
<td>32</td>
<td>30</td>
<td>8.3% (FN)</td>
</tr>
<tr>
<td>Legitimate</td>
<td>558</td>
<td>515</td>
<td>43</td>
<td>7.7% (FP)</td>
</tr>
<tr>
<td>Test Corpus</td>
<td>921</td>
<td>848</td>
<td>73</td>
<td>7.9% (MC)</td>
</tr>
</tbody>
</table>

*Optimal Architecture
FP (False Positive) – Legitimate message classified in Spam.
FN (False Negative) – Spam classified as legitimate e-mail.
MC (Misclassification) – Total Misclassified instances
It is evident from Table 2.3 that having 56 input, 24 hidden and 1 output layer neuron is the best architecture for the spam dataset. This architecture produced the lowest misclassification error and had slightly better convergence characteristics compared to other architectures tested.

### 2.7 Summary

In this chapter, an attempt was made to highlight potential of Machine Learning (ML), specifically ANN, in the online environment. The analogy of spam classification was used to explain the ANN architecture, the learning strategies and the design challenges.

Two popular ANN learning procedures were used to train the ANN. By comparing Figure 2.6 and Figure 2.7, it can be stated that GA produces better/quicker convergence. This gradual decline in the MSE over consecutive epochs also suggests that effective learning is taking place until the point where the MSE decelerates. The result of the GA is also comparable to other algorithms reported in the literature applied to the Spambase dataset (Table 2.4).

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Misclassification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adaptive Boost (AdaBoost)*</td>
<td>6.48%</td>
</tr>
<tr>
<td>Reinforcement Learning (RL)*</td>
<td>7.41%</td>
</tr>
<tr>
<td>Mixtures of Experts (MOE) architecture*</td>
<td>7.75%</td>
</tr>
<tr>
<td>Single MLP*</td>
<td>8.33%</td>
</tr>
<tr>
<td>Genetic Algorithm (Table 2.3)</td>
<td>7.93%</td>
</tr>
</tbody>
</table>

*Source (Dimitrakakis & Bengio, 2005)

In the next chapter, we introduce a local search technique, namely Simulated Annealing (SA), within the GA framework to enhance the solution quality and robustness of the ANN. The chapter also sheds light on some of the design issues of the genetic-hybrid algorithm, more commonly known as the Memetic Algorithm, and also discusses varying mechanisms of utilizing the local search to achieve the balance between exploration and exploitation in the algorithm.
Chapter 3

Memetic Algorithms

Memetic Algorithms (MAs) are a class of stochastic global search heuristics in which evolutionary algorithms (EA) are combined with local search techniques to improve the quality of the solutions created by evolution. MAs have been proven very successful across a wide range of problem domains such as bioinformatics (Krasnogor, 2004), multi-objective optimization (Knowles & Corne, 2001), optimization of non-stationary functions (Vavak, et al., 1996) and combinatorial optimization (Merz, 2006). The algorithm is also referred in literature by other names such as: hybrid GAs, genetic local search algorithms, Baldwinian EAs, Lamarckian EAs, and others. The name memetic algorithm was coined by (Moscato, 1989) to cover a wide range of techniques where evolutionary algorithm based search was augmented by the addition of one or more phases of local search.

In this chapter, we present the Memetic algorithm, a hybridized evolutionary approach and explore its ability to improve the learning process by combining two meta-heuristics, namely Genetic Algorithm and Simulated Annealing.

3.1 Underlying Evolutionary Framework for Memetic Algorithm

The common principle of all EAs is the same, given a population of individuals the environmental pressure causes natural selection (survival of the fittest) and this causes a rise in the fitness of the population. Figure 3.1 gives a general procedural overview on the GA life-cycle.
Figure 3.1: The general procedure of an evolutionary algorithm

It is important to note that many components of evolutionary algorithms are stochastic. During selection, fitter individuals have a higher chance to be selected than less fit ones, however even the weak individuals have a small chance to become a parent or to survive. For recombination of individuals the choice of which pieces will be recombined is random. Similarly for mutation, the pieces that will be mutated within a candidate solution, and the new pieces replacing them are chosen randomly.

3.2 Using EAs to train ANN

To use evolutionary algorithms to train neural networks, the weights and thresholds of ANN have to be represented in a chromosome (Fig. 3.2). These weights and thresholds are initialized to random numbers uniformly distributed over a range,

\[
\left(-\frac{2.4}{F_i},+\frac{2.4}{F_i}\right)
\]

(3.1)

where \(F_i\) is the number of inputs for the given neuron. Haykins in (Haykin, 1999) demonstrates this range to be particularly more effective when using a bipolar signal function, such as the sigmoid activation function used in this dissertation, to avoid slow learning.
Figure 3.2: Representation of ANN as a chromosome of weights ($w$) and thresholds ($\text{thres}$)

One of the major improvements in GAs has been the development of real coded scheme. The advantage of using real-coded genetic algorithms (RCGA) over using standard binary-coded GA is the maintainability of precision of real numbers which would otherwise be lost in binary GAs. In addition, it brings about a natural way of approximating real world situations.

3.2.1 Evaluation Function

The evaluation function represents the quality of an individual and is commonly called the **fitness function** in EA. The fitness for a classification problem is a measure of how accurately the algorithm is able to represent the training pattern. For classification problems such as spam classification, mean square error (MSE) is a good indicator of this value. The fitness of a chromosome is inversely proportional to the classification error.

\[
Fitness(p) = \frac{1}{Error_p} \quad (3.2)
\]

where $Error_p$ is the MSE of the chromosome at iteration $p$.

3.2.2 Parent Selection Mechanism

The role of parent selection is to allow for better individuals to become parents in the next generation. The process is typically probabilistic and the aim is to push for quality improvements. The most commonly used selection techniques in EA include the roulette wheel selection and elite based selection (Negnevitsky, 2005). In roulette wheel selection, the probability of selection is proportional to the fitness of the chromosomes.
The elitist based method however is biased towards selecting fitter chromosomes from a pool of top \( n \) percent of the population.

The EA population size is almost always constant, thus a choice has to be made on which individuals will be allowed in the next generation. For this reason, the survivor selection approach is also often called the replacement strategy.

### 3.2.3 Variation Operators – Recombination or Crossover

The crossover operator combines good solutions with different but desirable features. It produces new offspring from two parent chromosomes, by exchanging selected genes from each parent. Point crossover schemes choose a crossover point where the two parent strings ‘break’ and then exchanges the chromosome parts after that point.

A second crossover scheme commonly used is uniform crossover. Uniform crossover combines genes sampled uniformly from two parents. Each gene is chosen at random, independent of the others, to be included in the offspring. The resulting offspring is a more uniform representation of the parents. The decision to apply crossover is stochastic.

### 3.2.4 Variation Operators – Mutation

Mutation has low probability of occurrence, and is a means to introduce diversity in a population by introducing random variations. Mutation transforms a randomly selected gene in the chromosome. If the genotypes are bit strings, then inverting a 0 to a 1 (1 to a 0) can be used as a mutation operator. For a RCGA, this would mean adding a random noise to a random gene. The noise would be a random value bounded by the mutation amplitude.
3.3 Forming the Hybrid Meta-heuristic

Hybrid meta-heuristics refers to the combination of two or more meta-heuristic search techniques to solve a given complex problem. The major goal of hybrid meta-heuristic is to balance the diversification and intensification in the search process. Diversification is often referred to as global search or exploration. Intensification on the other hand is synonymous to local search, local refinement or exploitation.

Fig 3.3 shows some of hot-spots where local search could have been ideally implemented within the evolutionary algorithm.

![Evolutionary Algorithm - possible hybridization points](image)

**Figure 3.3: Evolutionary Algorithm - possible hybridization points**
3.3.1 Using a Hybrid Meta-heuristic to train ANNs

Some of the most important design consideration one needs to make while using a hybrid algorithm includes:

1. **Frequency of Local Search** - This refers to the number of continuous uninterrupted generations that GA performs before applying the local search algorithm. Lobo and Goldberg (Lobo & Goldberg, 1997) addressed the problem of when to employ the global search and local search in order to make the most of either technique. Utilizing the local search too frequently may not necessarily have significant improvement in the convergence but will incur more computational overhead. One way to balance this would be to adopt a technique of probability matching, where local search is employed depending on the efficiency of both genetic and local techniques as the search progresses. (Hacker, et al., 2002) on the other hand proposed that local search should be ignored till the global search algorithm has defined its basin of attraction. The homogeneity of the population is then checked using the values of coefficient of variance. If the variance is small (population has converged to small area), the local search is employed else the global search is continued. For ANN training, we consider this to be an effective technique to employ for our research.

2. **Duration of Local Search** – For a fixed duration, long local search duration will require fewer generations of the global algorithm to be executed than a hybrid system with shorter local search duration. Short local search are more likely to keep the population diverse. However, the depth of the search is very much dependent on the type of problem at hand.

A study by (Hart, 1994) found that using short duration local search performed best results for Griewank function (Griewank, 1981), whereas long local search produced better results for Rastrigin functions (Torn & Zilinskas, 1989). Hart et.
al. (Hart, et al., 2000) concluded that hybrid systems with long local search will be most effective for complex problems.

3. **Probability and Selection of Local Search Chromosome** - A local search can be applied to either every individual in the population or only a few individuals. Traditional hybrid genetic algorithms apply local search to every individual in the population. This approach however leads to high computational overhead. (Hart, 1994) investigated the impact of the fraction of population that undergo local search with respect to the overall performance of the hybrid algorithm. He concluded that a more selective use of local search produced better results. We adopted this approach to minimize the computation burden of training large networks.

### 3.4 Introducing Local Search within the EA framework

Local search is a search method that iteratively explores the promising regions identified by the EA (Hacker, et al., 2002). This accelerates the convergence and drastically reduces the time needed to reach optimum solution.

To implement local search, a simplified Simulated Annealing (SA) technique is incorporated within GA to explore the local promising region of the search space. The pseudocode for the SA local search algorithm is outlined in Figure 3.4.

SA is a smart heuristic for optimization having some similitude with the metal annealing analogy. Given a cost function in a search space, SA replaces the current solution by a random "nearby" solution generated as a function of the global parameter $T$ (temperature). The new solution is chosen with a probabilistic measure that depends on the difference between the corresponding function values and on a global parameter $T$. The temperature is gradually decreased during the annealing cycle. This fit chromosome produced after annealing is introduced back into the GA population and is expected to improve the fitness of the entire population.
Begin

generate initial solution
evaluate initial solution
set initial temperature ($T$)

Repeat Until (termination criterion is satisfied) Do

generate new solution
evaluate new solution
If new better than old
   replace old solution with new
Else
   compute $\Delta E (\Delta E = |fitness_{old} - fitness_{new}|)$
   compute acceptance probability ($p = e^{\Delta E / T}$)
   generate random probability ($r$)
   If ($r \leq p$)
      replace old solution with new
   EndIf
EndIf
Lower $T$
End Do
End

Figure 3.4: Pseudo-code of simulated annealing algorithm.

There are three principle components that affect the working on the local search algorithm. These include the pivot rule, the depth of local search and the neighbourhood generating function.

The pivot rule defines the criteria of accepting an improved solution. A steepest ascent pivot rule terminates the search only after the entire neighbourhood has been searched while greedy descent or first ascent pivot rule terminates the search as soon as an improvement is found.

The depth of local search is the terminating criteria of the number of improving steps to take and has a major effect on the performance of the local search algorithm, both in terms of time taken and the quality of solution found. The neighbourhood generating function defines the next set of points that could be reached by the application during exploration.
3.5 Proposed Memetic Framework for ANN training

It is worth emphasising that the benefit of combining two meta-heuristics must be balanced against an increase in the complexity in the design of the algorithm. Therefore, careful consideration must be placed on exactly how and where the hybridisation will be done.

In our scenario, EA is chosen for diversification and SA for intensification. As per the suggestion in a number of literature (El-Milhoub, et al., 2006) (Merz, 2006) (Suzuki, 2011), SA has been instigated selectively and only when the population becomes homogenous. This is to enhance the efficiency of the algorithm (reduce the training overhead). The proposed hybrid MA architecture is shown in Figure 3.5.

Figure 3.5: Proposed Memetic Algorithm
Some of the key design considerations incorporated for our memetic algorithm includes:

- Utilization of high mutation rate to promote diversity.
- Simplified form of SA eliminated the probability based hill-climbing.
- Selecting only a single chromosome from the parent population for local search and replacing the weakest solution with the new solution to reduce the computational overhead.
- Automatically instigate local search once the population becomes homogenous ($\Delta$MSE $< \beta$ in 10 consecutive epochs, where $\beta$ is a constant equal to 0.005).
- Fighting genetic drift by combining the concept of cooling schedule of simulated annealing (Kirkpatrick & Vecchi, 1983).
- Self-terminating training process as soon as the validation error starts to rise.

### 3.6 Performance of Memetic Framework on Spam Classification

The following section outlines the performance of the proposed memetic algorithm on the spambase dataset.

#### 3.6.1 MA Parameters

The global control parameters of the MA was set as follows:

- Population size = 50
- Selection ratio = 10%
- Mutation rate = 30%
- Crossover rate = 0.5

The optimized ANN architecture was derived empirically and has 57 input neurons, 19 hidden layer neurons and one (1) output layer neuron (Table 3.1). This meant that each chromosome in the population had 1102 weights and 20 thresholds to optimize.

A roulette wheel based selection was employed to select chromosome and uniform crossover scheme was applied. The crossover rate controls the frequency with which the crossover operator was applied. The only GA parameter changed from chapter 2 was the
mutation rate, which was increased to 30%. In fact, the experiment confirms the claim by (Rosin, et al., 1997) that the mutation rate in MA can be made more adventurous in its role. This aspect allowed us to significantly reduce the overhead of SA to only accepting fitter solution.

3.6.2 Initiation of local search

Empirical results in chapter 2 showed that GA exploration slows after 120 generations, thus a monitoring scheme is implemented to detect stagnation in MSE. The local search algorithm SA is activated when the MSE fails to improve by a threshold $\beta = 0.005$ over ten (10) consecutive epochs. This is in line with (Hacker, et al., 2002) who stated that an effective methodology to employ local optimization is only when the population becomes too homogeneous. Thus, the local search is only instigated once the learning process stagnates. Then a roulette wheel based selection is used to pick a single chromosome. As opposed to annealing the entire population, this selective approach will incur substantially less computational overhead.

The SA parameters controlled the local search process. The duration of the search is set at 100 iterations with the initial temperature, $T$, for annealing set at 10. $T$ is then gradually decreased to 0.01 (stop temperature) by a logarithmic cooling rate over the entire training process. This ensures higher randomness being added to the chromosome initially and lower randomness as the network stabilizes. Our training process runs over 1000 epochs. The primary objective is to introduce new and fitter chromosomes in the population by replacing the weakest individual. This technique ensures the population of good chromosomes is consistently maintained.

3.6.3 Results and Discussion

It can be noted from Figure 3.6 and Table 3.1 that 19 hidden layer neurons produced the best results. This architecture (56 input, 19 hidden, and 1 output layer neuron) produced smaller misclassification error and better convergence characteristics as shown in Figure 3.7 compared to the best-selected ANN trained using traditional GA.
Figure 3.6: Relationship between number of hidden layers neurons and MSE

Figure 3.7: MSE graph for the best selected ANN trained with GA and MA.
Table 3.1: MA Classification Results (5 run average)

<table>
<thead>
<tr>
<th>Neurons in Hidden Layer</th>
<th>Spam Precision</th>
<th>Legitimate Precision</th>
<th>Spam Recall</th>
<th>Legitimate Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>93.12%</td>
<td>93.75%</td>
<td>90.19%</td>
<td>95.66%</td>
</tr>
<tr>
<td>19*</td>
<td>93.14%</td>
<td>94.63%</td>
<td>91.67%</td>
<td>95.61%</td>
</tr>
<tr>
<td>24</td>
<td>93.21%</td>
<td>94.06%</td>
<td>90.70%</td>
<td>95.70%</td>
</tr>
<tr>
<td>29</td>
<td>92.46%</td>
<td>93.56%</td>
<td>89.92%</td>
<td>95.23%</td>
</tr>
<tr>
<td>34</td>
<td>92.27%</td>
<td>94.06%</td>
<td>90.74%</td>
<td>95.05%</td>
</tr>
<tr>
<td>39</td>
<td>92.58%</td>
<td>93.63%</td>
<td>90.03%</td>
<td>95.30%</td>
</tr>
<tr>
<td>44</td>
<td>92.07%</td>
<td>94.07%</td>
<td>90.80%</td>
<td>94.91%</td>
</tr>
<tr>
<td>29</td>
<td>15 **</td>
<td>91.95%</td>
<td>93.17%</td>
<td>89.31%</td>
</tr>
</tbody>
</table>

TEST DATA SUMMARY (19 NEURONS)

<table>
<thead>
<tr>
<th></th>
<th>Total Instances</th>
<th>Correctly Classified</th>
<th>Incorrectly Classified</th>
<th>Classification Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spam</td>
<td>363</td>
<td>333</td>
<td>30</td>
<td>8.26% (FN)</td>
</tr>
<tr>
<td>Legitimate</td>
<td>558</td>
<td>534</td>
<td>24</td>
<td>4.30% (FP)</td>
</tr>
<tr>
<td>Test Corpus</td>
<td>921</td>
<td>867</td>
<td>54</td>
<td>5.86% (MC)</td>
</tr>
</tbody>
</table>

* Optimal structure as per MSE(Figure 3.6)
** Architecture with 2 hidden layers
FP (False Positive) – legitimate email instances classified as spam.
FN (False Negative) – spam email instances classified as legitimate.
MC (Misclassification) – total misclassified instances

The proposed neural network classifier trained with memetic algorithm also outperformed other algorithms reported in literature (Table 3.2) which is a major achievement. This can be attributed to the Memetic algorithm’s ability to leverage local search capability of Simulated Annealing (SA) and global search capability of Genetic Algorithm (GA), to better optimize the parameters of the ANN.

Table 3.2: Algorithm comparison on Spambase dataset

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Misclassification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adaptive Boost (AdaBoost)*</td>
<td>6.48%</td>
</tr>
<tr>
<td>Reinforcement Learning (RL)*</td>
<td>7.41%</td>
</tr>
<tr>
<td>Mixtures of Experts (MOE) architecture*</td>
<td>7.75%</td>
</tr>
<tr>
<td>Single MLP*</td>
<td>8.33%</td>
</tr>
<tr>
<td>Genetic Algorithm (Table III)</td>
<td>7.93%</td>
</tr>
<tr>
<td>Memetic Algorithm (MA) (Table IV)</td>
<td>5.86%</td>
</tr>
</tbody>
</table>

*Source (Dimitrakakis & Bengio, 2005)
Its accuracy in modelling the spam patterns and precisely classifying new instances affirms the potential that hybrid algorithms have in solving complex real world problems.

### 3.7 Chapter Summary

This chapter discussed the improvements in the efficiency and performance of the ANN trained by memetic algorithm. The write-up highlights the various techniques, design consideration and hybridizing approaches considered in merging the two algorithms.

MA brings about capability enhancement to the traditional GA through improved solution quality and convergence efficiency. It also incorporates number of control parameters to selectively instigate procedures making the algorithm more robust for complex problem-solving.

The results show that hybridizing is a successful way to build a competent evolutionary algorithm that solves difficult problems quickly, reliably and accurately without the need for any forms of human intervention.

With a reliable prediction model now defined, we adopt it in the next chapter to answer some crucial pedagogical questions originating from the eLearning context at USP.
Chapter 4

Intelligent Student Monitoring in an e-Learning Environment

The use of eLearning platform has become indispensable for higher education institutions to deliver standardized and quality education to its campuses and its distant learners. The learning platform supports thousands of concurrent student logins and tens of thousands of interactions each day. This accumulates a lot of data that can introduce significant improvements in the way courses are taught and delivered.

In fact, earlier studies have suggested that higher education institutions could harness the Learning Management System (LMS) data to develop reporting tools that identify at-risk students and allow for more timely pedagogical interventions (Dawson, et al., 2008). One has to be cautious, however, about relying too heavily on the predictive power of simple correlation based models and the danger of confusing correlation with causation (Beer, et al., 2012) while interpreting the results. Students do not exhibit simple univariate patterns of online behaviour, but instead undertake complex composite behaviours in choosing different learning paths and time on activities.

Thus, this research adopts the machine learning approach to tackle this composite modelling problem unlike the statistical and/or correlation based methods adopted in many earlier research reported in (Romero & Ventura, 2010). The study adopts the optimized ANN model presented in the previous chapter to exploit the knowledge accumulated within the LMS. Analysis of this data is expected to answer some crucial questions originating from the eLearning context at USP.
The key research questions we attempt to answer in this section includes:

1. Which online interaction activities greatly aid the students in fulfilling the learning outcomes of the courses successfully?

2. Which training point (week 8 or week 15) is most suitable for training the ANN model for predicting 'at-risk' students around the faculty mid-semester reporting deadline (week 8)?

3. How accurately can we gauge the progress of students based on the history of interactions within the LMS?

4. Can the prediction model sustain itself in subsequent offerings as courses and online learning behaviours evolve over time?

### 4.1 The Student Dataset

For experimentation, six (6) different courses were identified from the four different disciplines within the School of Computing, Information & Math Sciences (SCIMS). SCIMS was one of early adaptors of Moodle at USP back in 2006, and the courses chosen have strong online presence. The course identified employed various levels of Moodle activities to support the teaching and learning process. In addition, the courses provided a lot of historical data to test the broader generalization ability of the ANN. The details of the courses are presented in Table 4.1:
Moodle accumulates a detailed account of every interaction a student makes with the system together with the completion status of tasks and the marks acquired in the various online activities. The detailed activity reports for all students (last access, number of times read, etc.) as accessible from Moodle is outlined in Figure 4.1.

Figure 4.1: A screenshot of the Moodle log report for UU100

Table 4.1: List of USP courses experimented

<table>
<thead>
<tr>
<th>Course</th>
<th>Discipline</th>
<th>Mode</th>
<th>Offering Semester (Experimented)</th>
<th>Class Count</th>
<th>Online Activities</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS111: Intro to Computing Science</td>
<td>Computing Science</td>
<td>F2F</td>
<td>Sem 1, 2012</td>
<td>227</td>
<td>Forums, Resources, Assignment</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Sem 1, 2013</td>
<td>260</td>
<td></td>
</tr>
<tr>
<td>MA101: Mathematics for Social Science</td>
<td>Math</td>
<td>Online</td>
<td>Sem 1, 2012</td>
<td>32</td>
<td>Forums, Resources, Quiz, Lesson, Glossary, Chat, Blog, Assignment</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Sem 1, 2013</td>
<td>88</td>
<td></td>
</tr>
<tr>
<td>IS222: Database Systems</td>
<td>Information Systems</td>
<td>Online</td>
<td>Sem 1, 2012</td>
<td>181</td>
<td>Forums, Resources, Quiz, Assignment, Lesson, Books, Glossary, Database</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Sem 2, 2012</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Sem 1, 2013</td>
<td>183</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Sem 2, 2013</td>
<td>84</td>
<td></td>
</tr>
<tr>
<td>IS323: Info Systems Analysis &amp; Design</td>
<td>Information Systems</td>
<td>F2F</td>
<td>Sem 1, 2011</td>
<td>220</td>
<td>Forums, Resources, Assignment</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Sem 1, 2012</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Sem 1, 2013</td>
<td>169</td>
<td></td>
</tr>
<tr>
<td>ST131: Introduction to Statistics</td>
<td>Statistics</td>
<td>F2F</td>
<td>Sem 1, 2012</td>
<td>80</td>
<td>Forums, Resources, Quiz, Assignment</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Sem 1, 2013</td>
<td>98</td>
<td></td>
</tr>
<tr>
<td>UU100: Comm. &amp; Information Literacy</td>
<td>University-wide</td>
<td>Blended</td>
<td>Sem 1, 2011</td>
<td>1462</td>
<td>Forums, Resources, Quiz, Assignment, Lessons</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Sem 2, 2011</td>
<td>1721</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Sem 1, 2012</td>
<td>1563</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Sem 2, 2012</td>
<td>2172</td>
<td></td>
</tr>
</tbody>
</table>
These reports, if further analysed may help identify the popular courseware items but are not very effective for progression diagnosis within the course. To derive usefulness from the user log and to extract the behavioral patterns embedded within it, the model should be able to deduce compound relationships and prediction maps from all the identifiable features. Table 4.2 lists twenty (20) course tracking variables extracted from the student log for one of experimented courses, UU100. Class is the target variable for prediction.

**Table 4.2: List of tracking attributes for UU100**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoginFreq</td>
<td>Login Frequency</td>
</tr>
<tr>
<td>NumAssignSubmitted</td>
<td>Num. of Assignment Submitted</td>
</tr>
<tr>
<td>DistAssignSubmitted</td>
<td>Num. of Distinct Assignments submitted</td>
</tr>
<tr>
<td>NumForumViews</td>
<td>Num. of Forums Read</td>
</tr>
<tr>
<td>DistForumViews</td>
<td>Num. of Distinct Forums Read</td>
</tr>
<tr>
<td>NumForumPosts</td>
<td>Num. of Forum Posting</td>
</tr>
<tr>
<td>DistForumPosts</td>
<td>Num. of Distinct Forum Postings</td>
</tr>
<tr>
<td>NumQuizStarted</td>
<td>Num. of Standard Quizzes Attempted</td>
</tr>
<tr>
<td>NumQuizCompleted</td>
<td>Num. of Standard Quizzes Completed</td>
</tr>
<tr>
<td>DistQuizAttempt</td>
<td>Num. of Distinct Standard Quizzes Attempted</td>
</tr>
<tr>
<td>NumRQuizStarted</td>
<td>Num. of Review Quizzes Attempted</td>
</tr>
<tr>
<td>NumRQuizPassed</td>
<td>Num. of Review Quizzes Passed</td>
</tr>
<tr>
<td>NumRQuizFailed</td>
<td>Num. of Review Quizzes Failed</td>
</tr>
<tr>
<td>DistRQuizPassed</td>
<td>Num. of Review Quizzes Passed</td>
</tr>
<tr>
<td>AvgRQuizScore</td>
<td>Average Review Quiz Score</td>
</tr>
<tr>
<td>NumResourceViewed</td>
<td>Num. of Resources Viewed</td>
</tr>
<tr>
<td>DistResourceViewed</td>
<td>Num. of Distinct Resources Viewed</td>
</tr>
<tr>
<td>NumBlogViewed</td>
<td>Num. of Blog/Wiki Participation</td>
</tr>
<tr>
<td>NumBookViewed</td>
<td>Num. of Book Views</td>
</tr>
<tr>
<td>Class</td>
<td>PASS or FAIL</td>
</tr>
</tbody>
</table>

The data for each course offering has been extracted at 15 different points during the semester (Week 1, Week 2, Week 3 … up til Week 15) from Moodle database using structured query language. The extracted features are stored as a CSV file (comma separated version). The privacy and confidentiality of student data is preserved. All easily identifiable markers, such as the student id number, has been masked when preparing the data for analysis.
4.2 Data Preparation

4.2.1 Data Normalization

Since the features from the LMS are measured using a variety of scales, normalization becomes a crucial step in data preparation. Normalization equalises the importance of variables and allows for symmetric comparability at different extraction points. It also allows student profiles from current offerings to be comparable to student profiles from previous offerings.

All the feature vectors have been scaled as follows:

\[
scaledY = \frac{Y - \min Y}{\max Y - \min Y} = \frac{X - \min X}{\max X - \min X} \quad (4.1)
\]

where \( \max Y \) and \( \max X \) are the maximum and minimum \( X \) values for that particular attribute respectively.

The interaction profile for week 1 is dropped for the experimentation. This is because the registration period continues at USP till the end of Week 1. Thus, with students still settling down in the new semester and limited teaching activities happening during the week, there are very limited online interactions resulting in a large number of null log entries for students in the LMS database. Cases of compassionate pass, “COM” grade, and withdrawal from course (where students never logged into Moodle) were also removed to maintain the integrity of the data.

The list of all the cleansed datasets available after pre-processing for experimentation is summarised in Table 4.3.
Table 4.3: Dataset for the Supervised Learning

<table>
<thead>
<tr>
<th>Course</th>
<th>Offering</th>
<th>Total Instances</th>
<th>Pass:Fail</th>
</tr>
</thead>
<tbody>
<tr>
<td>UU100</td>
<td>Sem1, 2011</td>
<td>1462</td>
<td>1056:406</td>
</tr>
<tr>
<td>UU100</td>
<td>Sem2, 2011</td>
<td>1721</td>
<td>1416:305</td>
</tr>
<tr>
<td>UU100</td>
<td>Sem1, 2012</td>
<td>1563</td>
<td>1261:302</td>
</tr>
<tr>
<td>UU100</td>
<td>Sem2, 2012</td>
<td>2172</td>
<td>1800:372</td>
</tr>
<tr>
<td>CS111</td>
<td>Sem1, 2012</td>
<td>227</td>
<td>165:62</td>
</tr>
<tr>
<td>CS111</td>
<td>Sem1, 2013</td>
<td>260</td>
<td>177:83</td>
</tr>
<tr>
<td>MA101</td>
<td>Sem1, 2012</td>
<td>32</td>
<td>24:8</td>
</tr>
<tr>
<td>MA101</td>
<td>Sem1, 2013</td>
<td>88</td>
<td>43:45</td>
</tr>
<tr>
<td>IS323</td>
<td>Sem1, 2011</td>
<td>220</td>
<td>189:31</td>
</tr>
<tr>
<td>IS323</td>
<td>Sem1, 2012</td>
<td>170</td>
<td>157:13</td>
</tr>
<tr>
<td>IS323</td>
<td>Sem1, 2013</td>
<td>169</td>
<td>156:13</td>
</tr>
<tr>
<td>ST131</td>
<td>Sem1, 2012</td>
<td>80</td>
<td>55:45</td>
</tr>
<tr>
<td>ST131</td>
<td>Sem1, 2013</td>
<td>98</td>
<td>64:34</td>
</tr>
<tr>
<td>IS222</td>
<td>Sem1, 2012</td>
<td>181</td>
<td>157:24</td>
</tr>
<tr>
<td>IS222</td>
<td>Sem2, 2012</td>
<td>58</td>
<td>46:12</td>
</tr>
<tr>
<td>IS222</td>
<td>Sem1, 2013</td>
<td>183</td>
<td>165:18</td>
</tr>
<tr>
<td>IS222</td>
<td>Sem2, 2013</td>
<td>84</td>
<td>72:12</td>
</tr>
</tbody>
</table>

4.2.2 Dealing with Class Imbalance

Class imbalance problem occurs where one of the two classes have more sample than other classes. Most algorithms, in an attempt to reduce its mean square error (MSE) focus more on classifying the major sample, while ignoring or misclassifying minority samples. The minority samples are those that rarely occur but are very important. Data from Table 4.3 show that there is on average approximately 4 times more pass samples than the fail samples for each of the respective course datasets.

To tackle the class imbalance issue a number of methods have been suggested in (Longadge & Dongre, 2013). The traditional approaches include under-sampling; which tries to balance the distribution of class by randomly removing majority class sample, or over-sampling; which achieves balanced class distribution by replication minority class sample.

We did not want to reduce the samples nor saturate the data and thus adopted a third widely used approach, commonly referred to as cost-sensitive learning (Longadge & Dongre, 2013). This method tries to maximize a loss function associated with a data set.
The cost-sensitive learner shifts the bias of a machine to favour the minority class. For the student classifier, we introduce a bias $\beta$ such that:

$$Y(x) = \begin{cases} 0, & x > \beta \\ 1, & x \leq \beta \end{cases}$$ (4.2)

More simply put, it is better to mistakenly identify a student as at risk of failure than to neglect a student requiring additional learning support. The value for $\beta$ has been determined empirically and has been set to $\beta = 0.7$ for all our experimentation.

### 4.3 Evaluating Performance

To determine the success of the prediction model, we employ the $F_1$ score (also known as $F$-score or $F$-measure). This is a widely used composite measure (Dembczynski, et al., 2013) that considers both the precision $p$ and the recall $r$ to eliminate the biasness of imbalanced data. $F_1$ score can be interpreted as a weighted average of the precision and recall, where an $F_1$ score reaches its best value at 1 and worst score at 0. The $F_1$ score of the class $k$ is computed as:

$$F_1(c_k) = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$ (4.3)

$$\text{precision} = \frac{N_{UU}}{N_{UU} + N_{SU}}$$ (4.4)

$$\text{recall (sensitivity)} = \frac{N_{LU}}{N_{LU} + N_{LS}}$$ (4.5)

where:

- $N_{UU}$ = the number of underperforming students correctly classified as underperforming.
- $N_{US}$ = the number of underperforming students incorrectly classified as satisfactory performers.
- $N_{SU}$ = the number of satisfactory performers incorrectly classified as underperforming students.
- $N_{SS}$ = the number of satisfactory performers correctly classified as satisfactory performers.
Intuitively, the precision measure identifies the percentage of predicted underperformers’ that are correct, whilst the recall highlights the percentage of actual underperformers that were predicted correctly.

4.4 Identifying significant online activities for accurate performance prediction

E-learning platform provide a spectrum of learning activities (about 25 different types of online activities are available in Moodle (Romero, et al., 2008)) such as quizzes, lessons, forums, blogs, assignments, glossaries, surveys, wiki, and workshops. Each activity allows a learner to engage through a number of action types, such as view, add, post, start, complete, and submit. This accumulates a large array of interaction trail within Moodle.

A central problem in machine learning is identifying the most salient features to construct a sustainable classification model for future prediction (Talavera, 1999). All features may be important for some problems, but for some target concepts, only a small subset of features is usually relevant. Feature selection, eliminates irrelevant and redundant data and, in many cases, improves the performance of learning algorithms.

4.4.1 Feature Selection Process

The process of feature selection (Novaković, et al., 2011) in ML consists of the following four steps:

- subset generation,
- subset evaluation,
- stopping criterion and
- result validation.

The aim is to create a subset of features, evaluate it using a given classification or correlation routine, and loop until an ending criterion is satisfied (Figure 4.2). Finally, the subset found is validated by the classifier algorithm using some unseen test dataset.
4.4.2 Attribute Ranking

For the purpose of this experiment, all the four steps of the feature selection process in Figure 4.2 were followed. The primary purpose of feature selection and ranking is to discard any irrelevant or redundant features from a given feature vector (Talavera, 1999).

To evaluate each subset, the following commonly used statistical and entropy-based methods were used.

- Information Gain (IG),
- Gain Ratio (IGR),
- Symmetrical Uncertainty (SU),
- Relief-F (RF),
- Chi-Squared (CS).

Entropy based methods are commonly used in the information theory and is a measure of how "mixed up" an attribute is. It is sometimes equated to the purity or impurity of a variable and is the foundation of the IG, IGR, and SU ranking methods. The entropy of $Y$ is computed as:

$$H(Y) = -\sum_{y \in Y} p(y) \log_2(p(y))$$  \hspace{1cm} (4.6)
where \( p(y) \) is the marginal probability density function for the random variable \( Y \). Features or variables are not mutually exclusive in all situations. If a relationship exists between features \( Y \) and \( X \), the entropy of \( Y \) after observing \( X \) is then calculated by:

\[
H(Y | X) = -\sum_{x \in X} p(x) \sum_{y \in Y} p(y | x) \log_2(p(y | x))
\]  

(4.7)

where \( p(y|x) \) is the conditional probability of \( y \) given \( x \).

\textit{i.) Information Gain}

Information Gain (IG) is a measure of the uncertainty associated with a random feature \( X \) and is computed as:

\[
IG(Y, X) = H(Y) - H(Y | X)
\]  

(4.8)

where \( H \) is the information entropy. Although IG is commonly used, it is biased towards tests with many outcomes (attributes having a large number of values).

\textit{ii.) Gain Ratio}

The Gain Ratio (GR) overcomes the bias of IG by measuring the IG with respect to the class.

\[
IGR(Y, X) = \frac{IG}{H(X)}
\]  

(4.9)

\textit{iii.) Symmetrical Uncertainty}

Symmetrical Uncertainty (SU) evaluates the worth of an attribute with respect to the class and is good if the attribute highly correlates to the class but not with the other attributes. It is computed as:

\[
SU(Y, X) = 2 \cdot \frac{IG}{H(Y) + H(X)}
\]  

(4.10)
iv.) **Chi-Squared**

*Chi-Squared* statistical measure is an overall measure of how close the observed frequencies are to the expected frequencies and is computed with the following formula:

\[
\chi^2 = \sum_{i=1}^{r} \sum_{j=1}^{c} \frac{(O_{ij} - E_{ij})^2}{E_{ij}}
\]  

(4.11)

where \(O_{ij}\) is the observed frequency and \(E_{ij}\) is the expected (theoretical) frequency.

v.) **Relief-F**

*Relief-F* is a simple measure to estimate the quality of attributes in problems with strong dependencies between attributes. It is based on the probability of the nearest neighbors from two different classes having different values for a feature and the probability of two nearest neighbors of the same class having the same value of the feature. The higher the difference between these two probabilities, the more significant is the feature.

### 4.4.3 Identifying the Final Ranks

Each ranking method mentioned above may generate a different rank for each feature due to its differing internal computation apparatus. In order to resolve the ranking, we employ a “panel of judges” approach in our experiment. The “panel of judges” approaches uses a majority voting (MV) scheme to ascertain the final list of ranked features using the rank commonality of each algorithm for each ranked attribute. The MV scheme is demonstrated in Figure 4.3.

![Figure 4.3: Majority Voting (MV) scheme to resolve ranking](image)

49
4.4.4 Validation of Ranked Features

In order to validate the results of the ranking algorithm, four commonly used supervised learning algorithms are adopted. These are, namely, IB1, Naïve Bayes, C4.5 decision tree and the radial basis function (RBF) network.

IB1 is a nearest neighbour classifier that uses normalized Euclidean distance to find the training instance closest to the given test instance, and predicts the same class as the training distance (Novaković, et al., 2011). Naïve Bayes, on the other hand, is a simple probabilistic classifier based on the elementary Bayes Theorem (Rish, 2001). The advantage of Naïve Bayes classifier is that it requires a small amount of training data to estimate the parameters (means and variances of the variables) necessary for classification. C4.5 is an algorithm used to generate a classification tree using the concept of information entropy (Ruggieri, 2002). It is simple to understand and interpret, requires little data preparation, is robust, and performs well with large data in a short time. Radial Basis Function (RBF) network is an artificial neural network that uses radial basis functions as activation functions. It has many uses, including function approximation, time series prediction, classification, and system control (Orr, 1999). RBF network offers a number of advantages, including requiring less formal statistical training, ability to implicitly detect complex nonlinear relationships between dependent and independent variables, ability to detect all possible interactions between predictor variables, and the availability of multiple training algorithms.

4.4.5 Results and Discussion

All the six (6) courses listed in Table 4.1 underwent feature selection in this experimentation. Ranking conflicts as stated earlier were resolved majority voting (MV) scheme.

Table 4.4 shows the final ranking obtained for the features in the six courses:
Table 4.4: Result of ranking methods on the six experimented courses

<table>
<thead>
<tr>
<th>UU100 Features</th>
<th>IG</th>
<th>IGR</th>
<th>SU</th>
<th>CS</th>
<th>RF</th>
<th>MV</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoginFreq</td>
<td>2</td>
<td>5</td>
<td>4</td>
<td>6</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>AssignSubmitted</td>
<td>4</td>
<td>9</td>
<td>8</td>
<td>4</td>
<td>14</td>
<td>4</td>
</tr>
<tr>
<td>DistAssignSubmitted</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>ForumViews</td>
<td>14</td>
<td>13</td>
<td>13</td>
<td>13</td>
<td>18</td>
<td>13</td>
</tr>
<tr>
<td>DistForumViews</td>
<td>13</td>
<td>14</td>
<td>14</td>
<td>14</td>
<td>3</td>
<td>14</td>
</tr>
<tr>
<td>ForumPosts</td>
<td>16</td>
<td>16</td>
<td>15</td>
<td>12</td>
<td>16</td>
<td>12</td>
</tr>
<tr>
<td>DistForumPosts</td>
<td>15</td>
<td>15</td>
<td>16</td>
<td>4</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>QuizStarted</td>
<td>9</td>
<td>8</td>
<td>9</td>
<td>8</td>
<td>11</td>
<td>8</td>
</tr>
<tr>
<td>QuizCompleted</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>DistQuizAttempt</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>16</td>
<td>2</td>
</tr>
<tr>
<td>NumRQuizStarted</td>
<td>8</td>
<td>6</td>
<td>7</td>
<td>9</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>NumRQuizFailed</td>
<td>7</td>
<td>7</td>
<td>6</td>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>DistRQuizPassed</td>
<td>17</td>
<td>17</td>
<td>17</td>
<td>17</td>
<td>6</td>
<td>17</td>
</tr>
<tr>
<td>DistRQuizFailed</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>15</td>
<td>3</td>
</tr>
<tr>
<td>AvgRQuizScore</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>8</td>
<td>12</td>
</tr>
<tr>
<td>DistResourceViews</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>13</td>
<td>13</td>
<td>11</td>
</tr>
<tr>
<td>DistResourceViews</td>
<td>6</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>BlogViews</td>
<td>18</td>
<td>18</td>
<td>18</td>
<td>17</td>
<td>18</td>
<td>18</td>
</tr>
<tr>
<td>BookViews</td>
<td>19</td>
<td>19</td>
<td>19</td>
<td>19</td>
<td>19</td>
<td>19</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ST131 Features</th>
<th>IG</th>
<th>IGR</th>
<th>SU</th>
<th>CS</th>
<th>RF</th>
<th>MV</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoginFreq</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>ForumViews</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>DistForumViews</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>ForumPosts</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>DistForumPosts</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>ResourceViews</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>8</td>
<td>6</td>
</tr>
<tr>
<td>DistResourceViews</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>BlogViews</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td>BookViews</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MA101 Features</th>
<th>IG</th>
<th>IGR</th>
<th>SU</th>
<th>CS</th>
<th>RF</th>
<th>MV</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoginFreq</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>6</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>ForumViews</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>11</td>
<td>15</td>
<td>10</td>
</tr>
<tr>
<td>DistForumViews</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>10</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>ForumPosts</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>10</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>DistForumPosts</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td>QuizStarted</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>QuizCompleted</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>DistQuizAttempt</td>
<td>17</td>
<td>17</td>
<td>17</td>
<td>17</td>
<td>3</td>
<td>17</td>
</tr>
<tr>
<td>LessonStarted</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>LessonEnded</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>DistGlossaryAdd</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>5</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>DistGlossaryView</td>
<td>14</td>
<td>14</td>
<td>14</td>
<td>7</td>
<td>13</td>
<td>14</td>
</tr>
<tr>
<td>GlossaryView</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>14</td>
<td>8</td>
</tr>
<tr>
<td>GlossaryAdd</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>7</td>
<td>15</td>
</tr>
<tr>
<td>ResourceViews</td>
<td>13</td>
<td>13</td>
<td>13</td>
<td>14</td>
<td>12</td>
<td>13</td>
</tr>
<tr>
<td>DistResourceViews</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>13</td>
<td>8</td>
<td>11</td>
</tr>
<tr>
<td>BlogViews</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>18</td>
<td>16</td>
</tr>
<tr>
<td>BookViews</td>
<td>18</td>
<td>18</td>
<td>18</td>
<td>18</td>
<td>18</td>
<td>18</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IS323 Features</th>
<th>IG</th>
<th>IGR</th>
<th>SU</th>
<th>CS</th>
<th>RF</th>
<th>MV</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoginFreq</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>ForumViews</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>6</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>DistForumViews</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>ForumPosts</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>8</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>DistForumPosts</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>7</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>LabSubmission</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>ResourceViews</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>4</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>DistResourceViews</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>5</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>BlogViews</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>9</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>BookViews</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>10</td>
<td>6</td>
<td>6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CS111 Features</th>
<th>IG</th>
<th>IGR</th>
<th>SU</th>
<th>CS</th>
<th>RF</th>
<th>MV</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoginFreq</td>
<td>5</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>Assignment upload</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>DistAssign</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>ForumViews</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>9</td>
<td>5</td>
</tr>
<tr>
<td>DistForumViews</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>ForumPosts</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>DistForumPosts</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>LabSubmission</td>
<td>4</td>
<td>7</td>
<td>6</td>
<td>3</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>ResourceViews</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>DistResourceViews</td>
<td>7</td>
<td>4</td>
<td>5</td>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>BlogViews</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td>BookViews</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>12</td>
<td>10</td>
<td>10</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IS222 Features</th>
<th>IG</th>
<th>IGR</th>
<th>SU</th>
<th>CS</th>
<th>RF</th>
<th>MV</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoginFreq</td>
<td>5</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>ForumViews</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>13</td>
<td>8</td>
</tr>
<tr>
<td>DistForumViews</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>8</td>
<td>6</td>
</tr>
<tr>
<td>ForumPosts</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>10</td>
<td>7</td>
</tr>
<tr>
<td>DistForumPosts</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>Lab Work Submissions</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>TotAssign</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td>DistAssign</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>12</td>
<td>16</td>
</tr>
<tr>
<td>DistQuizAtt</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>TotalQuizComp</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>DistLessonAtt</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>DistLessonAtt</td>
<td>14</td>
<td>14</td>
<td>14</td>
<td>14</td>
<td>16</td>
<td>14</td>
</tr>
<tr>
<td>ResourceViews</td>
<td>13</td>
<td>13</td>
<td>13</td>
<td>11</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>DistResourceViews</td>
<td>2</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>BlogViews</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>5</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>BookViews</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>14</td>
<td>10</td>
<td>10</td>
</tr>
</tbody>
</table>
It can be clearly noticed that the ranks vary for the different ranking methods. This is largely due to the internal evaluation measures within the ranking algorithms. The final ranking adopted is the resolved ranking obtained from the MV scheme. The highlighted ranks are features that have been deemed insignificant or redundant after validation in Figure 4.4.

Another notable observation during the ranking process was that the ranking of attributes also changed from semester to semester. Table 4.5 shows this for consecutive offering for the UU100 course. Similar occurrences have been seen for the other five courses.

**Table 4.5: Results of MV Scheme showing changing ranks in UU100 across offerings**

<table>
<thead>
<tr>
<th>Course Features/ Rank</th>
<th>S1-2011</th>
<th>S2-2011</th>
<th>S1-2012</th>
<th>S2-2012</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoginFreq</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>AssignSubmitted</td>
<td>4</td>
<td>6</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>DistAssignSubmitted</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>ForumViews</td>
<td>13</td>
<td>14</td>
<td>13</td>
<td>14</td>
</tr>
<tr>
<td>DistForumViews</td>
<td>14</td>
<td>13</td>
<td>14</td>
<td>13</td>
</tr>
<tr>
<td>ForumPosts</td>
<td>16</td>
<td>16</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>DistForumPosts</td>
<td>15</td>
<td>15</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>QuizStarted</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>7</td>
</tr>
<tr>
<td>QuizCompleted</td>
<td>10</td>
<td>10</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>DistQuizAttempt</td>
<td>2</td>
<td>7</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>NumRQuizStarted</td>
<td>9</td>
<td>3</td>
<td>8</td>
<td>11</td>
</tr>
<tr>
<td>NumRQuizPassed</td>
<td>7</td>
<td>5</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>NumRQuizFailed</td>
<td>17</td>
<td>17</td>
<td>17</td>
<td>17</td>
</tr>
<tr>
<td>DistRQuizPassed</td>
<td>3</td>
<td>1</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>AvgRQuizScore</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>ResourceViews</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>9</td>
</tr>
<tr>
<td>DistResourceViews</td>
<td>6</td>
<td>8</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>BlogViews</td>
<td>18</td>
<td>18</td>
<td>18</td>
<td>18</td>
</tr>
<tr>
<td>BookViews</td>
<td>19</td>
<td>19</td>
<td>19</td>
<td>19</td>
</tr>
</tbody>
</table>

The elimination of the insignificant features was validated using the four ML algorithms, namely IB1, Naive Bayes, C4.5 decision tree and the radial basis function (RBF) network. The tests were conducted with 10 fold cross-validation for each offering and the average accuracy of the four (4) classifiers has been reported using $F_1$ score. The results are shown in Figure 4.4.
Figure 4.4: Average F1 Score and optimal elimination point for each course

It can be noted that each course has its own optimal elimination point. Eliminating the poor features significantly improves the F1 Score of the classifier. For instance, it is notable in the case of UU100 (Figure 4.4 a), that removing the bottom nine (9) lowly
ranked features contributed significantly towards improving the accuracy of the prediction model. It suggests that some features are better for predicting pass or fail but does not necessarily state that the courseware item is effective for learning.

Similar interpretations can be made for other courses. The comparative performances of implementing feature selection versus using the complete feature set is demonstrated in Figure 4.5 and highlighted in Table 4.6.
Figure 4.5: Performance impact of feature selection versus using all attributes
It is not surprising to see that the benefit of feature selection is not significant in the e-learning domain. This is because, the e-learning environments is continuously evolving. Alterations such as inclusion or removal of online activities, changes in assessment weightings, changes in learner behaviour, change in offering focus and extension of activity deadlines are regular occurrences. Having a rigid model with restricted feature set is bound to become invalid for such environments.

Feature selection is usually adopted to reduce the input dimensions to suit the classifier and to improve prediction performance (Hagen & Morris, 2005). However, since elimination of features has not brought about any noticeable improvement in the predictor performance, we propose to continue the rest of the experimentation with the entire feature set. This is to ensure consistency in future experimentations. Also, learning from all the features will ensure that the systems do not become specialised to a single offering thus eliminating frequent retraining.

### 4.5 Identifying the most suitable training profile for predicting 'at-risk' students

USP has mandated all the course coordinators within its three (3) faculties and the College of Foundation Studies (CFS) to prepare mid-semester reports on a semesterly
basis. This is to allow for identification of “at-risk” student performers so that remedial interventions can be undertaken. The due date of this report is Week 8 and is enforced by the Faculty Learning and Teaching (L&T) Office.

These individual course reports together with the school report are submitted by the L&T Office to Student Administrative Services (SAS), Faculty Academic Standards and Quality Committee (FASQC), Academic Standards and Quality Committee (ASQC), Deputy Vice-Chancellor – Learning Teaching & Student Services (DVC-LTSS), Campus Directors (CD) and to all the sponsors for their respective interests and decision making. Additional details about this report are provided in Appendix 2.

Some of the issues of the mid-semester report is that it is not regular, the preparation process is manual and the report is sometimes inaccurate. Therefore, it would be better to have an automated, real-time monitoring and reporting system.

4.5.1 Experiment Setup

In this section, we explore to proposition of whether Week 8 profile or Week 15 (complete) profile is more suited for training the ANN model for predicting 'at-risk' students around the faculty mid-semester reporting deadline. To test this hypothesis, we again consider all six (6) courses listed in Table 4.1. The ANN is trained using students profiles extracted at both Week 8 and Week 15 of the semester. The underperformers identified by the learning model is then compared with the summary prepared by the course coordinator in the mid-semester report.

4.5.2 Results and Discussion

Firstly, to identify the most suitable training profile, the model is trained with Week 8 data. A second classifier is trained with Week 15 data. The prediction accuracy for both the trained models is evaluated on the test data from the subsequent offering using the F1 score.

The performance comparison of using different training points for the ANN model is presented in Table 4.7.
Table 4.7: Average F1-Score of training with Week 8 versus Week 15 profile

<table>
<thead>
<tr>
<th>Course</th>
<th>Training Dataset</th>
<th>Test Dataset</th>
<th>Test result for model trained with Week 8 data</th>
<th>Test result for model trained with Week 15 data</th>
</tr>
</thead>
<tbody>
<tr>
<td>ST131</td>
<td>S1-2012</td>
<td>S1-2013</td>
<td>0.793</td>
<td>0.800</td>
</tr>
<tr>
<td>MA101</td>
<td>S1-2012</td>
<td>S1-2013</td>
<td>0.730</td>
<td>0.779</td>
</tr>
<tr>
<td>CS111</td>
<td>S1-2012</td>
<td>S1-2013</td>
<td>0.804</td>
<td>0.811</td>
</tr>
<tr>
<td>IS323</td>
<td>S1-2011</td>
<td>S1-2012</td>
<td>0.911</td>
<td>0.898</td>
</tr>
<tr>
<td>IS323</td>
<td>S1-2012</td>
<td>S1-2013</td>
<td>0.934</td>
<td>0.946</td>
</tr>
<tr>
<td>IS222</td>
<td>S1-2012</td>
<td>S2-2012</td>
<td>0.893</td>
<td>0.903</td>
</tr>
<tr>
<td>IS222</td>
<td>S2-2012</td>
<td>S1-2013</td>
<td>0.927</td>
<td>0.940</td>
</tr>
<tr>
<td>IS222</td>
<td>S1-2013</td>
<td>S2-2013</td>
<td>0.865</td>
<td>0.886</td>
</tr>
<tr>
<td>UU100</td>
<td>S1-2011</td>
<td>S2-2011</td>
<td>0.804</td>
<td>0.901</td>
</tr>
<tr>
<td>UU100</td>
<td>S2-2011</td>
<td>S1-2012</td>
<td>0.901</td>
<td>0.901</td>
</tr>
<tr>
<td>UU100</td>
<td>S1-2012</td>
<td>S2-2012</td>
<td>0.793</td>
<td>0.906</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td>0.850</td>
<td>0.879</td>
</tr>
</tbody>
</table>

From Table 4.7, it can be noted that using Week 15 training profile on average has a slight edge over Week 8 profile for most course in terms of F1-Score around the mid-semester reporting deadline. Week 15 profile also proves considerably better (Figure 4.6) than week 8 for progressive monitoring of student performance from as early as Week 3 of the semester. The results show that the graph exhibits less variability (much smoother curve) for the F1-Score performance measures when Week 15 profile is used.

![Figure 4.6: Performance of ANN model trained on Week 8 vs Week 15 profile and evaluated on UU100 test data](image)
We also choose in favour of Week 15 as it is a complete profile for training the ANN model. This will allow for a more learned model that can be used to more accurately predict student performances beyond Week 8 as highlighted in Figure 4.8.

This outcome holds similar resemblance to pattern recognition problems such as image recognition (Tapia & Perez, 2013) and gene-expression profiling (Stiekema, et al., 2013) where complete profiles are often used for training the models and perform better even when the actual prediction is done for partial or obscured profiles.

### 4.6 Performance of ANN-based model to identify at-risk students

Based on the findings in the previous section, at this stage in our experiment, we have trained our neural network model using memetic algorithm on week 15 dataset for all the six courses utilizing full feature set. A series of experiments with varying ANN architectures (number of hidden layers \{1,2\}), number hidden layer neurons \{5,10,15\} and classification threshold \{0.3,0.5,0.7\} was conducted. The parameter for MA was set as follows (refer to Appendix A.1 for full parameter details):

**Global Search parameters (GA)**
- Population Size = 50
- Selection Ratio = 10%
- Mutation Rate = 30%
- Crossover Probability = 0.5

**Local Search parameters (SA)**
- Start Temperature = 10
- Stop Temperature = 0.1
- Number of Cycles = 10
- Cooling Schedule = 0.9 (geometric)

A total of 11 offerings from six courses have been tested to validate our ANN model for predicting “at-risk” or underperforming students. The optimal ANN architecture consisted of a single hidden layer and comprised of 20 inputs, 5 hidden and 1 output layer neurons. The output threshold of the output layer neuron was identified as \( \beta = 0.7 \).
The result on the ANNs ability to distinguish between satisfactory and unsatisfactory (at-risk) learners compared to the course coordinator is presented in Table 4.8:

**Table 4.8: Average F1- Score of ANN model versus Course Coordinators Report**

<table>
<thead>
<tr>
<th>Course</th>
<th>Offering</th>
<th>Course Coordinator (Week 8)</th>
<th>ANN Approach (Week 8)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ST131</td>
<td>S1-2013</td>
<td>0.7937</td>
<td>0.8592</td>
</tr>
<tr>
<td>MA101</td>
<td>S1-2013</td>
<td>0.8222</td>
<td>0.7961</td>
</tr>
<tr>
<td>CS111</td>
<td>S1-2013</td>
<td>0.7758</td>
<td>0.8234</td>
</tr>
<tr>
<td>IS323</td>
<td>S1-2012</td>
<td>0.9126</td>
<td>0.9272</td>
</tr>
<tr>
<td>IS323</td>
<td>S1-2013</td>
<td>0.8896</td>
<td>0.9630</td>
</tr>
<tr>
<td>IS222</td>
<td>S2-2012</td>
<td>0.8511</td>
<td>0.8736</td>
</tr>
<tr>
<td>IS222</td>
<td>S1-2013</td>
<td>0.9161</td>
<td>0.9438</td>
</tr>
<tr>
<td>IS222</td>
<td>S2-2013</td>
<td>0.9060</td>
<td>0.8346</td>
</tr>
<tr>
<td>UU100</td>
<td>S2-2011</td>
<td>0.9115</td>
<td>0.9177</td>
</tr>
<tr>
<td>UU100</td>
<td>S1-2012</td>
<td>0.9063</td>
<td>0.9290</td>
</tr>
<tr>
<td>UU100</td>
<td>S2-2012</td>
<td>0.8682</td>
<td>0.8904</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td></td>
<td><strong>0.8594</strong></td>
<td><strong>0.8871</strong></td>
</tr>
</tbody>
</table>

The results show that the trained neural network provides comparable or better performance in nearly all the occasions for identifying “at-risk” students. The accuracy is more trustable as it considers the students complete commitment in the course and not just assessment scores. What is more notable though is that ML opens up a new pathway for continuous management of student progression (Figure 4.9). Mid-semester reports are just a single point of feedback for the stakeholders and does not account for the learners confidence and commitment in the course.

Using the course coordinators approach also has a few limitations. Some course coordinators have reported that their courses have just one or two assessments in the first half of the semester. The identification of “at-risk” students becomes even more challenging when these assessments are group-based. Some students may not have done well in assessment items due to their group members having performing well or unforeseen circumstances such as missing a short test or quiz due to sickness or natural disasters.
Another common concern is the lack of assessments or non-availability of the assessments marks by the reporting deadline. Thus a common practise is that most course coordinators resort to is to use the attendance records or the assignment submission status to identify at-risk students. However, those students identified in the true-sense may not be actual cases of underperformers as students have their own learning styles as well as reasons for non-participation in assessments.

The ANN-based approach, however, could be used for a wider cross-section of courses. This is clearly evident with the high accuracies attained for the good mix of courses in Figure 4.7. The model is able to capture the learning process more holistically (online commitment, completion, engagement and achievement) proving itself as an innovative and effective means of learner diagnosis.
4.7 Identifying an adaptive learning strategy for subsequent offering

4.7.1 Importance of Adaptation

Adaptability is a common problem faced by most intelligent systems. This involves dealing with non-static data from dynamic environments. When there is a shift in data, the classification or prediction models need to be adaptive to the changes. In data mining the phenomenon of change in data distribution over time is known as concept drift (Gama, et al., 2014). However, neural network is not new to such dynamical and non-stationary time series. Over the past few years, neural networks have been successfully used to model financial time series ranging from options prices (Jain & Kumar, 2007), corporate bond ratings (Moody & Utans, 1994) and stock index trading (Chen, et al., 2003) to currency exchange (Kamruzzaman & Sarker, 2003). This robust quality is one of the main reasons why ANN was one of the preferred algorithms chosen for the prediction task in this research.

Because data is expected to evolve over time, its underlying distribution can change dynamically over time. The phenomena of concept drift between time point \( t_0 \) and time point \( t_1 \) can be explained as:

\[
\exists X : p_{t_0}(X,y) \neq p_{t_1}(X,y)
\]  

(2.4)
where \( p_{t_0} \) denotes the joint distribution at time \( t_0 \) between the set of input variables \( X \) and the target variable \( y \). Changes in data can be characterized as changes in the components of this relation \((Gao, et al., 2007)\) such as:

- the prior probabilities of classes \( p(y) \) may change,
- the class conditional probabilities \( p(X|y) \) may change, and
- as a result, the posterior probabilities of classes \( p(y|X) \) may change affecting the prediction.

The two common types of drifts are:

1. Real concept drift – this refers to changes in \( p(y|X) \). Such changes can happen either with or without change in \( p(X) \). In other words, the class boundary changes. Real concept drift is sometimes also referred to as concept shifting or conditional change \((Gao, et al., 2007)\).

2. Virtual drift occurs if the distribution of the incoming data changes, that is \( p(X) \) changes without affecting \( p(y|X) \) \((Delany, et al., 2005)\) \((Tsymbal, 2004)\). In simpler terms, the distribution of inputs change but the class boundary stays the same.

Figure 4.8 illustrates the two types of drifts. The plot shows that in real concept drift, the class boundary changes making the previous decision model obsolete. In real world scenarios, virtual drift changing prior probabilities or novelties may appear in combination with the real drift and therefore also affects the class boundary.

![Figure 4.8: Types of drifts: circle represents instance, different colours represent different classes](Gama, et al., 2014)
4.7.2 Concept Drift in E-Learning Environment

Changes in data distribution in eLearning context may manifest in different forms. A drift may happen abruptly or it can be incremental, gradual or recurring, as illustrated in Figure 4.9.

![Figure 4.9: Concept drift: Patterns of change using one-dimensional data](image)

In such scenarios, predictive systems will need to adapt, otherwise its accuracy will degrade over time. Table 4.9 lists the types of drift commonly encountered in the educational context.

<table>
<thead>
<tr>
<th>Change in Data</th>
<th>Possible Causes</th>
</tr>
</thead>
</table>
| **Gradual**    | - change in students study habits overtime  
|                | - inclusion/removal of a few online activities  
|                | - minor changes in assessment weightings  
|                | - change in teaching approach or focus  
|                | - change in coordinator  
|                | - incentives given to students for online participation  |
| **Abrupt**     | - change of mode of offering (e.g. from F2F to Online)  
|                | - major course revisions (as per review cycle 3-5 years)  
|                | - change in Learning platform/tools  |

4.7.3 Experimental setup for determining suitable adaptive learning scheme

The experimentation in this section explores different approaches of improving the LMS and the cost associated (performance and efficiency) in maintaining the learned model as new data becomes available. It concludes with a practical and feasible learning strategy to sustain the ANN model over multiple semesters.

The *learning mode* refers to the technique of updating the predictive model using the evolving data. Figure 4.10 shows the learning mechanisms that can be adopted for our experiment.
There are two different learning modes: Retraining, in which the model is trained at the beginning with all the available data. Whenever new data arrives, the previous model is discarded, the new data is merged with the previous data, and a new model is learned on this data. Incremental adaptation updates the current model with only the newly available data. The model in operation is strengthened as new data becomes available over time.

In the instance-incremental method, the classifier learns from each example as it arrives. Batch-incremental on the other hand learns from consecutive batches of data. For the e-learning context, although the learner data trails become available in real-time, the complete dataset with the final grades only becomes available at the end of the semester making the batch-incremental learning more suitable for this scenario.

To explore the most feasible learning strategy, the experiments tests the following three cases:

**Scenario I:** Periodically retraining - This assumes that there is no need to retrain the model on a semesterly basis. The model has a binding lifetime, synonymous to the course review cycle of 3-5 years after which, the model is discarded and a newly trained model can implemented.

**Scenario II:** Complete retraining is done on a semester basis. Here, the initial model is built from the very initial offering. Whenever new data arrives, the previous model is discarded, the new data is merged with the previous data, and a new model is learned on this data.
Scenario III: Incremental online adaptation which updates the current model with only the newly arrived data. The starting point of retraining is the existing knowledge preserved in as the starting weight of the ANN classifier. For this scenario, we adopt the incremental batch adaptation approach as the new data arrives in batches.

We use three courses, namely UU100, IS222 and IS323 to explore the most feasible learning strategy. The reason for selecting these courses is because they have complete logs for 3 or more consecutive offerings. These courses also are offered in different modes. UU100 is offered in blended mode, IS222 is offered in fully online mode and IS323 is a face to face course enabling us to verify the strategy for different learning context. These courses also exist at different levels in the undergraduate programmes. The results are based on the average of 5 test runs for each scenario.

4.7.4 Results and Discussion

The learning strategy was evaluated in terms of performance and efficiency. The result for the respective courses using the three scenarios is outlined in Figure 4.11-4.13.

The objective was to enhance the classifier such that the old knowledge retained in the weights of the ANN is improved with new arriving samples and future predictions can be done more accurately. In doing so, the model should avoid the phenomenon of catastrophic forgetting (Polikar, et al., 2001), where it completely forgets the past patterns when exposed to a new set of patterns. Scenario II, thus, becomes the control experiment as it is presented with both the old as well as the new knowledge during the retraining process.

Empirical results, however, rightfully highlight that this approach (Scenario II) is computationally expensive compared to the other two approaches in terms of number of fitness evaluation during training. This is depicted for subsequent offerings of UU100, IS222 and IS323 courses in Figure 4.11 (b), Figure 4.12 (b), and Figure 4.13 (b) respectively. This was expected as the learning process completely discards the old model as soon as new data become available and learns on the merged dataset. This
required the system to keep the old training set. The performance of complete retraining 
(*Scenario II*) is better than the periodic retraining (*Scenario I*), but not as good as 
incremental online adaptation (*Scenario III*).

*Scenario III*, which adopts incremental online adaptation, delivers the best accuracy in 
all the three courses when it comes to predicting student status in future offerings as 
highlighted in Figure 4.11 (a), Figure 4.12 (a), and Figure 4.13 (a) respectively. This 
suggests that the design consideration and strategies within the hybrid memetic 
algorithm has coped well in its ability to continuously learn from the new data while 
preserving previously learned knowledge, known as the stability- plasticity dilemma 
(Carpenter, et al., 1991). The improvement in accuracy over complete retraining also 
advocates that some degree of forgetting has to be allowed in the prediction model due 
to the ever evolving nature of the e-learning environment. Incremental online learning 
also reduces the learning cost by approximately 20% (Figure 4.11(b), Figure 4.12(b) and 
Figure 4.13(b)) for each of the three courses and distinguishes itself as a promising 
strategy.

The periodic retraining approach (*Scenario I*) incurs the least learning cost as it assumes 
no future retaining. The accuracy of the results, however, is slightly poorer than the 
other two approaches. Additionally, it must be noted that as the Internet, mobile 
technologies, and open education further accelerates pedagogical transformation in 
higher education, the assumption of “no need to adapt” may not be valid.

Therefore, we conclude this section by recommending the incremental online adaptation 
strategy for sustaining the hybrid memetic trained ANN model. This learning strategy 
has suited the three courses experimented regardless of their mode on offering. This 
scheme also delivers greater solution accuracy and performance efficiency compared to 
the other approaches.

We conclude by suggesting that period training strategy may be taken advantage of by 
some special courses which have a very strict offering structure or have been introduced 
for limited offerings, and the educators do not prefer to undertake semesterly retraining 
exercise.
UU100: F1-Score for each Learning Approach

<table>
<thead>
<tr>
<th>Task</th>
<th>Periodic Retraining</th>
<th>Complete Retraining</th>
<th>Incremental Learning</th>
</tr>
</thead>
<tbody>
<tr>
<td>to predict S2-2011</td>
<td>13352</td>
<td>13352</td>
<td>13352</td>
</tr>
<tr>
<td>to predict S1-2012</td>
<td>-</td>
<td>9146</td>
<td>5744</td>
</tr>
<tr>
<td>to predict S2-2012</td>
<td>-</td>
<td>9478</td>
<td>6216</td>
</tr>
<tr>
<td>Total</td>
<td>13352</td>
<td>31976</td>
<td>25312</td>
</tr>
<tr>
<td>Percentage of Total</td>
<td>41.8%</td>
<td>100.0%</td>
<td>79.16%</td>
</tr>
</tbody>
</table>

(b) UU100: Num. of fitness evaluations for each retraining approach

(c) UU100: Histogram showing computational overhead for each retraining approach

Figure 4.11: UU100 results for adaptive learning Scenario I, II and III.

69
(a) IS222: F1-Score for each Learning Approach

<table>
<thead>
<tr>
<th>Task</th>
<th>Periodically Retrain</th>
<th>Complete Retraining</th>
<th>Incremental Learning</th>
</tr>
</thead>
<tbody>
<tr>
<td>to predict S2-2012</td>
<td>5676</td>
<td>5676</td>
<td>5676</td>
</tr>
<tr>
<td>to predict S1-2013</td>
<td>-</td>
<td>6010</td>
<td>4975</td>
</tr>
<tr>
<td>to predict S2-2013</td>
<td>-</td>
<td>6678</td>
<td>3773</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>5676</strong></td>
<td><strong>18364</strong></td>
<td><strong>14424</strong></td>
</tr>
<tr>
<td><strong>Percentage of Total</strong></td>
<td><strong>30.9%</strong></td>
<td><strong>100.0%</strong></td>
<td><strong>78.54%</strong></td>
</tr>
</tbody>
</table>

(b) IS222: Num. of fitness evaluations for each retraining approach

(c) IS222: Histogram showing computational overhead for each retraining approach

Figure 4.12: IS222 results for adaptive learning Scenario I, II and III.
(a) IS323: F1-Score for each Learning Approach

<table>
<thead>
<tr>
<th>Task</th>
<th>Number of Fitness Evaluations</th>
<th>Periodically Retrain</th>
<th>Complete Retraining</th>
<th>Incremental Learning</th>
</tr>
</thead>
<tbody>
<tr>
<td>to predict S1-2012</td>
<td>5810</td>
<td>5810</td>
<td>5810</td>
<td></td>
</tr>
<tr>
<td>to predict S1-2013</td>
<td>-</td>
<td>4274</td>
<td>2905</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>5810</td>
<td>10084</td>
<td>8715</td>
<td></td>
</tr>
<tr>
<td>Percentage of Total</td>
<td>57.6%</td>
<td>100.0%</td>
<td>86.42%</td>
<td></td>
</tr>
</tbody>
</table>

(b) IS222: Num. of fitness evaluations for each retraining approach

(c) IS323: Histogram showing computational overhead for each retraining approach

Figure 4.13: IS323 results for adaptive learning Scenario I, II and III.
4.8 Chapter Summary

In this chapter we have successfully demonstrated the ability of the ANN to predict “at-risk” students. The results not only show that the proposed approach has better accuracy over the course coordinators current reporting practice, but its sheer ability to detect the underperformers much sooner gives all stakeholders sufficient time to implement remedial actions.

Feature selection techniques were successfully employed and provided major insight into the evolving nature of e-learning landscape. It also suggested that some features are better for predicting student performance. Also, dominant learning activities and learner actions in the LMS could facilitate courseware maintenance.

We also identified using empirical investigation that the Week 15 is most suited for training the ANN model. Finally, the adaptive online learning strategy using incremental learning is suggested to update the ANN-based prediction model to sustain itself in subsequent offerings.
Chapter 5

Discussion: Challenges and Opportunities in E-Learning

In chapter 4, we successfully demonstrated the ability of the ANN to model student behaviour in the e-learning environment. A supervised learning approach was undertaken where the model used past behavioural traits of students to answer some common questions arising in the e-learning context. An adaptive learning strategy was also suggested.

This chapter focuses on some specific challenges and opportunities arising from this research and how it could be effectively capitalised upon.

5.1 Lack of sufficient prior knowledge

In our research, the ANN-based student prediction model employs a supervised learning approach. Since the model is data-driven, its performance is only as good as the quality and diversity of the training set. Situations may arise in the e-learning context when there will be insufficient prior knowledge available for the model to learn. This would usually occur when a new course is introduced or where the nature of an existing course (activities, assessment portfolio, and mode) may change significantly from one offering to the other, thus, prior knowledge would be unavailable or inappropriate.

In such a situation, we have proposed the application of Self-Organizing Map (SOM) (Singh, et al., 2012). SOM is one of the most popular neural network models based on unsupervised learning. It belongs to the category of competitive learning networks developed by Kohonen (Kohonen, 1995). It provides a topology preserving mapping
from the high dimensional space to map units. Figure 5.1 provides a graphical representation of the SOM architecture.

![SOM architecture with a 2D hexagonal grid.](image)

**Figure 5.1: SOM architecture with a 2D hexagonal grid.**

The mapping process usually works on the euclidean distance similarity measure and aims to partition the data in natural groups. It allows visualization of the cluster structure of the SOM using the unified distance matrix (U-Matrix) and the distance matrix (D-Matrix) techniques.

Inspired from this research, this paper demonstrated the potential of the SOM technique on the first online offering of MA101 course in Semester 2, 2010. This was the very first online course offered by USP and had 80 students enrolled from five different campuses. The k-means clustering algorithm with $k=2$ (the number of clusters in which the data is divided) was used to differentiate satisfactory and unsatisfactory learners based on their online interaction logs.

The data was extracted at five (5) different points of the semester. Code vectors in Figure 5.2 clearly show the formation of clusters as the course progress during the semester.
Figure 5.2: U-Matrix with class labels at 5 different points during the semester

It can be noted that the clusters were visible from the week 5 dataset. The measure of the cluster quality is presented in Figure 5.3. It is worth noting that although the accuracy is not as high as supervised learning; the model does provide an acceptable platform for identifying underperformers when there is no prior knowledge available at all. It achieves around 60% accuracy around Week 8 and reaches 80% by the end of the semester.

Figure 5.3: Graph showing the accuracy of the unsupervised learning algorithm on MA101 S2-2010

A lot has changed since Semester 2-2010. More effective learning tools and activities have been introduced into Moodle since. Accessibility to internet has improved in the regional campuses and course designers are now more informed of learner preferences.
and online development options. Therefore, it is highly likely that unsupervised learning approach would be able to deliver better accuracies when the supervised learning option is not available.

The clustering approach of SOM also shows great potential as well and can be deployed towards recognizing similar student learners (“study buddies”) to promote greater collaboration amongst them in studies.

5.2 Recommender System Development

Recommender system is a major deployment that can eventuate from the insights gained from the extensive learner generated trails in the LMS. Recommender systems can suggest activities to students based on the behaviour traits in order to achieve learning success.

Association rule mining (ARM) is one of the most well studied methods in data mining (Agrawal, et al., 1993) (Kotsiantis & Kanellopoulos, 2006). It has served as a useful tool for discovering correlated items in a large transactional database. It produces if-then statements concerning attribute-values. An association rule $X \Rightarrow Y$ expresses that in those transactions in the database where $X$ occurs; there is a high probability of having $Y$ as well. $X$ and $Y$ are called the antecedent and consequent of the rule. The strength of such a rule is measured by its support and confidence.

We applied ARM algorithm on top 10 ranked features of UU100 course dataset to discover the combination of prominent learning activities or actions undertaken by successful students (Singh & Lal, 2013).

These prominent interaction attributes were obtained via feature selection techniques discussed in the earlier section. The reason to limit the attribute set was to eliminate the common problem of ARM where the algorithm discovers a huge quantity of rules to comprehend and many of these rules are found to be irrelevant (Simon, et al., 2013).
All the continuous numeric attributes were transformed into discrete attributes for easier categorization. After employing the Apriori algorithm with a minimum support of 0.2 and a minimum confidence of 0.9, the following rules were ascertained:

Table 5.1: Rules extracted using Apriori-frequent algorithm

<table>
<thead>
<tr>
<th>Rule</th>
<th>Antecedent</th>
<th>Consequent</th>
<th>Conf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DistAssignSubmitted=MEDIUM, DistResourceViews=HIGH, DistQuizAttempt=HIGH, AvgRQuizScore=HIGH</td>
<td>Class =Pass</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>ResourceViews=HIGH, LoginFreq =MEDIUM</td>
<td>Class=Pass</td>
<td>0.92</td>
</tr>
<tr>
<td>3</td>
<td>DistAssignSubmitted=MEDIUM</td>
<td>Class=Pass</td>
<td>0.97</td>
</tr>
<tr>
<td>4</td>
<td>DistQuizAttempt=HIGH</td>
<td>Class=Pass</td>
<td>0.96</td>
</tr>
<tr>
<td>5</td>
<td>AssignSubmitted=LOW, DistQuizAttempt=HIGH, ResourceViews=LOW, QuizCompleted=LOW</td>
<td>Class=Pass</td>
<td>0.96</td>
</tr>
</tbody>
</table>

As can be noticed in Table 5.1, the rules are very precise and very easy to interpret. The confidences of the rules are also quite high. Rule 4, for example, states that having a high value for number of distinct quiz attempts is a behaviour that could significantly contribute towards achieving a pass in the course. Similar proposition is also demonstrated by Rule 5. It also unveils that not all the highly significant activities have to be completed in order to achieve success in the course. For instance, if a student is lacking in certain activities, being able to identify the correct supplementing activity can assist in achieving success.

Comparison of rules and their confidence reveals interesting knowledge. For instance, comparing Rule 3 and 4 suggests that distinct Assignment activities are more effective in the course compared to the Quiz activities.

Another recommender system opportunity possible with the current data is to provide adaptive sequence of content based on learner behaviour, sometimes referred to as “intelligent curriculum”. This would involve matching interaction profile of current
learners with similar profiles of past successful students. Based on this similarity analysis, the most prominent learning activities can be suggested. This is an area we hope to venture into in our future research.

5.3 Integration with the e-learning system

The true beneficiaries of the e-learning prediction models are the students, instructors and the course creators. Thus, it is important that the prediction tools are integrated into the e-learning environment. In this way, the tool will be more widely used by educators, and feedback and results obtained can be directly accessible within the e-learning platform.

The ANN-based model used in this research is implemented using C Sharp programming language and runs on an independent machine. Its input features are accepted in CSV (comma separated values) format and pre-processed locally. The model uses a single generalized architecture for all the courses. The training process runs without any human intervention and the output is exported to a text file.

Our future work will integrate the adaptive hybrid ANN-based classifier with USP’s e-learning platform (Moodle) to make it a more effective system. The data communication process can be automated with the Moodle database. The student warning dashboard can be implemented in Moodle as a block within the course. Figure 5.4 demonstrates the simplified overview of the proposed implementation.
All processing will be done on an independent back-end server to avoid any overhead on the e-learning platform. The training process will only be required to run on a semesterly basis. For prediction, the data tables can be synchronized in real-time or a *cron* job can send the input features to the back-end server at preset check points (e.g., weekly or on strategic dates depending on the course instructor). The pre-processing and the classification task will be performed on the local server and the result will communicated with the Moodle server as a response message to be displayed in the block for the respective student.
Chapter 6

Conclusion and Future work

Large amount of sophisticated user activity and interactions data is captured in real-time in the LMS of many higher education institutes. Failure to exploit this astonishing array of data to improve teaching and learning practices especially at USP, envisioned this research. The undertaking involved implementing a robust ANN-based classifier that possessed the ability to handle voluminous data in real-time and could operate in such a dynamic online environment. Through exploration and careful consideration, a hybrid memetic meta-heuristic with optimal design was identified and incorporated for training the ANN model to cope with difficult error manifolds and deliver trained neural networks that guaranteed better convergence and finer performance in the online context. An efficient adaptive online learning strategy is also suggested for sustaining the ANN-based classification model for life-long learning in e-learning environment.

The finalized ANN model was used to verify how effectively we could predict student performance based on the history of their interaction in the e-learning environment. The model trained itself from past behavioural data (student interactions) to form a prediction map that permitted rating a students’ ability to pass the course. Whilst the faculty mid-semester reports provided just a single point of feedback for the stakeholders the ML approach opened up a new pathway for continuous performance monitoring and provided an opportunity for much earlier intervention.

E-learning systems has recently seen some advancements in the implementation of student flagging system such as Course Signal at Purdue University, Early Warning
System at USP, PACE at Rio Salado College – Arizona and Check My Activity tool at UMBC. These systems are rule or criterion based discrete classification models operating on a limited set of pre-determined indicators such as marks, views and activity completion. Our approach distinguishes itself from these systems as it considers the behavioural aspects (learning styles) when evolving the model and predicts performance based on a student’s personalized learning commitment/profile. Additionally, our approach truly advocates the principle of online learning which is meant to be learner-centered, flexible and self-paced and not constrained to deadlines and pre-determined scores as employed in the aforementioned systems.

Our future research intends to extend the current work on behavioural modeling to explore the opportunity of providing incremental learner diagnosis on preparedness before assessments, further categorization of “at-risk” students into high risk and low risk students (to distinguish dropout traits and failure traits), producing learner-centered “on the fly” curriculum and other student support mechanisms such as distinguishing similar student learners (“study buddies”) to promote greater collaboration amongst them in studies.
References


Appendix A

A.1 Parameters for Memetic Algorithm

The Feed Forward Neural Network (FFNN) was trained using the hybrid Memetic Algorithm (MA) to attain its optimal weights. The ANN prediction model had an input layer, a single hidden layer and an output layer. The architecture of the ANN is given in Figure A.1

![Architecture of the ANN used to test the prediction problem](image)

**Figure A.1: Architecture of the ANN used to test the prediction problem**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Purpose</th>
<th>Assigned Value</th>
<th>Reason for Choice of Value</th>
</tr>
</thead>
</table>
| $N$       | Determine the number of neurons in the hidden layer | $N = 19$ for spam dataset  
$N = 5$ for student dataset | This parameter was determined empirically |
| $\beta$  | Moves the output threshold of transfer function toward inexpensive classes such that examples with higher costs become harder to be misclassified | $\beta = 0.7$ | This parameter was determined empirically and was chosen such that it was sufficient to classify the imbalanced datasets |

**Table A.1: ANN parameters**

The Memetic Algorithm (MA) was used to train the ANN and was controlled by a number of parameters to control the global and local search processes. These parameters
were kept constant during all experiments, and in any instant, if there is any change to one or more parameters, these would be stated clearly for the specific experiment.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Purpose</th>
<th>Assigned Value</th>
<th>Reason for Choice of Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Global Search Parameters (Genetic Algorithm)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Population Size</td>
<td>Lists how many chromosomes (possible solutions) are in population</td>
<td>50</td>
<td>This parameter was determined empirically via experimentations.</td>
</tr>
<tr>
<td>Selection Ratio</td>
<td>Probability for a chromosome to be selected for crossover</td>
<td>10%</td>
<td>This parameter was determined empirically via experimentations.</td>
</tr>
<tr>
<td>Mutation Rate</td>
<td>Measure of the likeness that random elements of the chromosome will be flipped into something else (improves diversity)</td>
<td>30%</td>
<td>This parameter was determined empirically via experimentations.</td>
</tr>
<tr>
<td>Crossover rate</td>
<td>Controls the frequency with which the crossover the crossover operator is applied</td>
<td>0.5</td>
<td>This parameter was determined empirically via experimentations.</td>
</tr>
<tr>
<td>Crossover Type</td>
<td>Approach of combining parent solutions to produce a child solution</td>
<td>Uniform Crossover</td>
<td>This parameter was determined empirically via experimentations.</td>
</tr>
<tr>
<td>Max number of training Epochs</td>
<td>Criterion to stop the training process</td>
<td>1500</td>
<td>This value has been determined empirically after multiple observations during initial experimentations</td>
</tr>
<tr>
<td>Error Rate ( err )</td>
<td>Criterion to stop the training process assuming sufficient learning has taken place</td>
<td>( err = 0.005 )</td>
<td>This parameter was determined empirically via experimentations.</td>
</tr>
</tbody>
</table>

| **Local Search Parameter (Simulated Annealing)** |                                                                        |                |                                               |
| Initial temperature \( T \) | Usually large starting value to allow SA to explore the entire solution space | \( T = 10 \)  | This parameter was determined empirically via experimentations. |
| Stop temperature \( t \)     | Suitably low temperature acting as the stopping criteria for the algorithm | \( t = 0.1 \) | This parameter was determined empirically via some experimentation. |
| Cycles \( c \)               | The number of trials allowed at each temperature level                  | 10             | This parameter was set to a low value to minimise overhead. |
| Cooling schedule              | Dictate the rate at which simulated annealing reaches its final solution | Geometric 0.9  | This scheme was chosen due to its success in most literature |

Table A.2: MA parameters that controlled the training process
A.2 Operating Platform for ANN

The ANN and the training algorithms were implemented using C Sharp (C#) programming language which runs on the Common Language Infrastructure (CLI) and Common Language Runtime (CLR) platform which is the virtual machine component of Microsoft’s .NET Framework. This programming language was chosen because of the candidates’ high competence in the language and the rich class libraries available within the language API’s provided a sound platform to support the research by providing a centralized framework for creating, evolving and testing different architectures of artificial neural network easily.

Experiments were conducted on two Windows® machines. The specifications of each computer are shown in Table A.3.

<table>
<thead>
<tr>
<th>Computer</th>
<th>Operating System</th>
<th>Processor</th>
<th>RAM</th>
<th>System Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computer 1</td>
<td>Windows® 7</td>
<td>Intel® Core™ 2</td>
<td>2 GB (DDR2)</td>
<td>32-bit</td>
</tr>
<tr>
<td>(Desktop)</td>
<td></td>
<td>Duo</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Computer 2</td>
<td>Windows® 7</td>
<td>Intel® Core™ i5</td>
<td>8 GB (DDR3)</td>
<td>64-bit</td>
</tr>
<tr>
<td>(Laptop)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table A.3: Specifications of computers used for conducting tests
Appendix B

B.1   Communiqué from FSTE L&T Office

(Next Page)
June 18, 2014

TO WHOM THIS MAY CONCERN

This serves to attest that the following information is released upon the request of Mr. Shaveen Singh (S02009725), a Master of Science candidate at the University of the South Pacific, for the purpose of his research.

The schedule below provides a summary of the underperforming students in the following courses as per the mid-semester reports submitted to the Learning and Teaching Office at FSTE.

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Offering</th>
<th>Coordinator</th>
<th>Class Count</th>
<th>“At-Risk” Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>ST131</td>
<td>Semester 1-2013</td>
<td>Dr. MGM Khan</td>
<td>103</td>
<td>38</td>
</tr>
<tr>
<td>MA101</td>
<td>Semester 1-2013</td>
<td>Dr. Bibhya Sharma</td>
<td>96</td>
<td>31</td>
</tr>
<tr>
<td>CS111</td>
<td>Semester 1-2013</td>
<td>Prof. Ansgar Fehner</td>
<td>277</td>
<td>52</td>
</tr>
<tr>
<td>IS323</td>
<td>Semester 1-2012</td>
<td>Mr. Dinesh Kumar</td>
<td>174</td>
<td>16</td>
</tr>
<tr>
<td>IS323</td>
<td>Semester 1-2013</td>
<td>Mr. Ronal Singh</td>
<td>176</td>
<td>39</td>
</tr>
<tr>
<td>IS222</td>
<td>Semester 2-2012</td>
<td>Mr. Ronal Singh</td>
<td>71</td>
<td>28</td>
</tr>
<tr>
<td>IS222</td>
<td>Semester 1-2013</td>
<td>Mr. Ronal Singh</td>
<td>190</td>
<td>42</td>
</tr>
<tr>
<td>IS222</td>
<td>Semester 2-2013</td>
<td>Mr. Vivnesh Prasad</td>
<td>90</td>
<td>24</td>
</tr>
<tr>
<td>UU100</td>
<td>Semester 2-2011</td>
<td>Mr. Shaveen Singh</td>
<td>1759</td>
<td>269</td>
</tr>
<tr>
<td>UU100</td>
<td>Semester 1-2012</td>
<td>Mr. Shaveen Singh</td>
<td>1691</td>
<td>247</td>
</tr>
<tr>
<td>UU100*</td>
<td>Semester 2-2012</td>
<td>Mr. Shaveen Singh</td>
<td>1953</td>
<td>380</td>
</tr>
</tbody>
</table>

*Excludes Tivatol and Emalus Campus

Schedule of courses with their summary statistics

I hope the above information would be held in confidence and not be divulged to any other party without prior consent from the faculty.

All the best

Dr. Bibhya Sharma
Associate Dean - Learning and Teaching
Faculty of Science, Technology and Environment
The University of the South Pacific
Private Mail Bag Suva, Fiji
Mobile: +(679) 998 9577
Phone: + (679) 323 2069
Fax: + (679) 323 1416
B.2 Semesterly Call for Mid-Semester Report (FSTE L&T Office)

(Next Page)
Dear Colleagues:

I hope and pray that everything is progressing well in your courses.

We are fast approaching the mid-semester break and that means that we will have to check and flag students who are in danger of failing (at-risk students) and organise extra help for them from Week 8 onwards. Some of us have already begun extra sessions & clinics for the late registrants and the students on probation, we merely invite the at-risk students to join these sessions or maybe include more sessions if the numbers are big.

As before, we are required to prepare mid-semester reports for all our courses. These reports should include the sections: course details; assessments carried out; the performance in general; tutorial attendance (of those who maybe in the danger zone); list of **ALL** the at-risk students; list of **ALL** sponsored students with comments on their performances; remedial actions planned for the at-risk students, etc. For this semester we can include a section on late registrants and students on probation.

**Note: the Mid-Semester Reports are Mandatory.**

From past semesters, remedial actions such as extra tutorial session per week, focused tutorial exercises, in-form tests, study clinics (weekly as well as in the study break), f2f consultations, and increased SLS support, mandatory peer mentoring or study buddies, etc have been quite effective. However, please note that if we include an item to the list of remedial work, we have to make sure that it actually takes place 😊

Please upload your reports (a sample attached) on the Faculty L&T portal under the mid-semester reports section. Also submit the report to your HoS who will then send a school report (a summary of assessments, performances and remedial actions from the school) to the L&T Office.

The individual and school reports will be submitted by the L&T Office to SAS, FASQC, ASQC, DVC (LTSS), CDs and to all the sponsors. The school reports will also be presented in the ASQC. So essentially, we will have only ONE report to produce unlike 3 or 4 separate ones required in the past. Less work for us 😊

The due date for the individual mid-semester reports to be uploaded on the L&T Portal is **11 APRIL.**

With kind regards

*Dr. Bibhya Sharma*

*Associate Professor of Mathematics*

*Associate Dean – Learning and Teaching*

*Faculty of Science, Technology and Environment*

*The University of the South Pacific*

*Private Mail Bag Suva, Fiji*

*Mobile: +(679) 998 9577*

*Phone: +(679) 323 2069*

*Fax: +(679) 323 1416*
B.3 Sample Mid-Semester Report

Summary of the report is presented to preserve the anonymity of student data:

| Total Enrolment: | 176 students |
| Assessments Completed by Week 8: | Assignment 1 (7%), Test 1 (10%) and Tutorial & Lab Participation (2.3%) |
| Total Number of Needy Students identified: | 39 Students |

Thanking you

Mr. Ronal Singh
IS323 Coordinator