



Faculty of Computer Science and Information Technology

Information Theoretic-based Feature Selection for Machine Learning

Muhammad Aliyu Sulaiman

**Doctor of Philosophy
2018**

Information Theoretic-based Feature Selection for Machine Learning

Muhammad Aliyu Sulaiman

A thesis submitted

In fulfilment of the requirements for the degree of Doctor of Philosophy

(Computer Science)

Faculty of Computer Science and Information Technology
UNIVERSITI MALAYSIA SARAWAK
2018

DECLARATION

The work described in this PhD thesis, entitled “**Information Theoretic-based Feature Selection for Machine Learning**” is to the best of the author’s knowledge that of the author except where due reference is made.

The thesis has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.



(Student’s signature)

ID: 14010100

Muhammad Aliyu Sulaiman

February 2018

Date

ACKNOWLEDGEMENT

First and foremost, I would like to give thanks to almighty Allah (He who taught man the use of the pen) for giving me the ability and strength to see to the completion of this thesis work.

Acknowledgment is due to Universiti Malaysia Sarawak for giving me the opportunity to study and making my dreams come true.

My special and sincere gratitude goes to my Ph.D. thesis supervisor, Assoc. Prof. Dr. Jane Labadin for her support, guidance and encouragement throughout this work. I remained grateful for her critical review and superior suggestions in the course of this thesis work.

I am thankful to the Dean, Centre for Graduate Studies, Prof. Dr. Fasihuddin Badruddin Ahmad, the Dean, Faculty of Computer Science and Information Technology, Assoc. Prof. Dr. Johari Abdullah, Head of the Department of Computational Science & Mathematics, Dr. Chiew Kang Leng and other academic & non-academic staffs of the faculty for their cooperation and supports.

My acknowledgment goes to the members of my extended family; my parent, my wife and kids, my uncles, my aunts, brothers, sisters, friends and family friends for their unshakable caring feelings towards my studies and life in general.

I remain indebted to friends I met in UNIMAS whom time and space will not allow me to enumerate all of them here. However, I am thankful to Mr. Adam Mani Yangora and Dr. Shakeel for their enormous support throughout my time of study in UNIMAS.

ABSTRACT

Three major factors that determine the performance of a machine learning are the choice of a representative set of features, choosing a suitable machine learning algorithm and the right selection of the training parameters for a specified machine learning algorithm. This thesis tackles the problem of feature selection for supervised machine learning prediction tasks through dependency information. The feature evaluation strategy is formulated based on mutual information (MI) to handles both classification and regression supervised learning tasks and the search strategy is a modified greedy forward strategy designed to manage redundancy between features and avoiding features that are irrelevant to the predicting output. The problem with many existing feature selections that evaluate features based on mutual information is that they are designed to handles classification tasks only. And the few existing ones that can work for regression tasks were recently found to underestimate mutual information between two strongly dependent variables. In addition to these problems, the search strategy which is usually a heuristic greedy method used with many existing feature selections, lacks scientifically sound stopping criterion and the forward greedy procedure despite its advantages over the backward procedure is found to reveal suboptimal. Thus, this thesis has developed and evaluated a filter based Information Theoretic-based Feature Selection (IFS) for machine learning. Various experiments were carried out to assess and test components of IFS algorithm. The first test was designed to evaluate the formulated IFS Selection Criterion Strategy (MI estimator) by comparing it with six different MI estimator benchmarks. The second test evaluates IFS in a controlled study using simulated datasets. Moreover, the third test used ten natural domain datasets obtained from UCI Repository, in about fifteen different experiments, using three to four different Machine Learning

Algorithms for performance evaluation. Also, additional experiments to compare the relative performance of the IFS with five related feature selection algorithms were carried out using natural domain datasets. Besides, this thesis developed a hybrid filter method to enhance the performance of the IFS. IFS served as filter together with an Ant Colony Optimization System (ACO) as a metaheuristic form the hybrid system. In these extended IFS method, feature selection method was defined and presented as a 0-1 Knapsack Problem (MKP). Thus, this thesis precisely developed and evaluated IFS_BACS (Binary Ant Colony System) hybrid method. Further experiments were carried out using the natural domain datasets and comparison were made between IFS and hybrid IFS_BACS methods. In most of the cases, experimental results of IFS and its extended IFS_BACS hybrid method significantly reduced features and produce competitive performance accuracy when compared to the results of the full feature set before applying the IFS or IFS_BACS method. And comparing the IFS with its extended version, the extended version (IFS_BACS) seems to be more promising in selecting optimal feature subset from large datasets.

Keywords: Feature Selection, Information Theory, Mutual Information, Entropy, Density Estimation, Optimization, Machine Learning Algorithms, Supervised Learning, Modeling, Ant Colony System.

Maklumat Berasaskan Teori Pemilihan Ciri untuk Pembelajaran Mesin

ABSTRAK

Tiga faktor utama yang menentukan prestasi pembelajaran mesin adalah pilihan set wakil ciri-ciri, memilih algoritma pembelajaran mesin yang sesuai dan pilihan yang tepat parameter latihan bagi algoritma pembelajaran mesin yang ditetapkan. Tesis ini menangani masalah pemilihan ciri untuk tugas-tugas ramalan pembelajaran mesin diselia melalui maklumat kebergantungan. Strategi penilaian ciri dirumuskan berdasarkan maklumat bersama (MI) untuk mengendalikan kedua-dua klasifikasi dan regresi di bawah seliaan tugas-tugas pembelajaran dan strategi carian adalah satu strategi ke hadapan tamak diubahsuai direka untuk menguruskan lebihan antara ciri-ciri dan ciri-ciri mengelakkan yang tidak relevan dengan output meramalkan. Oleh itu, maklumat penapis berdasarkan pemilihan ciri berdasarkan teori-(IFS) untuk pembelajaran mesin telah dibangunkan dan dinilai. Pelbagai eksperimen telah dijalankan untuk menilai dan komponen ujian algoritma IFS. Ujian pertama telah direka untuk menilai IFS pemilihan strategi kriteria yang digubal (MI penganggar) dengan membandingkannya dengan enam garis dasar penganggar MI berbeza. Ujian kedua yang telah dilakukan untuk menilai IFS dalam kajian yang dikawal menggunakan set data yang tiruan. Ujian ketiga menggunakan sepuluh dataset domain semula jadi yang diperolehi daripada UCI repositori, kira-kira lima belas eksperimen yang berbeza, menggunakan 3-4 algoritma pembelajaran mesin yang berbeza untuk penilaian prestasi. Dalam usaha untuk meningkatkan prestasi IFS, kaedah penapis-wrapper hibrid telah dibangunkan. IFS telah digunakan sebagai penapis bersama-sama dengan semut sistem koloni pengoptimuman metaheuristic (ACO) sebagai pembungkus. Dalam kaedah IFS lanjutan ini, kaedah pemilihan ciri ditakrifkan dan dibentangkan sebagai 0-1 masalah bunttil multidimensi (MKP). Oleh itu, kaedah hibrid IFS_BACS (binari sistem koloni semut) telah

dibangunkan dan dinilai. Eksperimen selanjutnya telah dijalankan dengan menggunakan set data domain semula jadi dan perbandingan antara IFS dan IFS_BACS kaedah hibrid telah dibuat. Dalam kebanyakan kes, keputusan eksperimen kaedah hibrid IFS dan IFS_BACS dilanjutkan berkurangan ciri-ciri dan menghasilkan ketepatan prestasi kompetitif berbanding keputusan ciri penuh ditetapkan sebelum menggunakan kaedah IFS atau IFS_BACS.

Kata kunci: *Pemilihan Ciri, Teori Maklumat, Maklumat Bersama, Entropi, Ketumpatan Anggaran, Pengoptimuman, Algoritma Pembelajaran Mesin, Pembelajaran Terselia, Pemodelan, Sistem Koloni Semut.*

TABLE OF CONTENTS

	Page
DECLARATION	i
ACKNOWLEDGEMENT.....	ii
ABSTRACT.....	iii
<i>ABSTRAK</i>.....	v
TABLE OF CONTENTS	vii
LIST OF TABLES	xii
LIST OF FIGURES	xiv
CHAPTER 1: INTRODUCTION.....	1
1.1 Background of the Study	2
1.2 Motivation.....	5
1.3 Thesis Statement	7
1.4 Objectives	9
1.5 Scope of the Thesis	9
1.6 Thesis Overview	10
CHAPTER 2: SUPERVISED MACHINE LEARNING: CONCEPT AND	
DEFINITION.....	11
2.1 Machine Learning Tasks.....	11
2.1.1 The Classification Task.....	12
2.1.2 The Regression Task.....	12
2.2 Data Representation	13
2.3 Preprocessing Steps	16
2.3.1 Data with Missing Values.....	17
2.3.2 Data Scaling	18

2.3.3	Data Transformation	20
2.3.4	Sampling	21
2.4	Machine Learning Models	21
2.4.1	Model Representation and Cost Function	22
2.5	Learning Optimization and Performance Evaluation	34
2.5.1	Gradient Descent Algorithm	34
2.5.2	Normal Equations	35
2.5.3	Performance Evaluation	37
2.6	Attribute Discretization	39
CHAPTER 3: LITERATURE REVIEW		43
3.1	Feature Selection Versus Feature Extraction	43
3.2	Feature Selection for Big Data	44
3.3	Characteristics of Feature Selection Algorithms	45
3.3.1	Search Strategy	45
3.3.2	Evaluation Strategy	46
3.3.3	Stopping Strategy	46
3.4	Heuristic and Metaheuristic Search	47
3.5	Feature Filters	50
3.5.1	Distance Measures Filter-based Feature Selection	51
3.5.2	Information Measures Filter-based Feature Selection	52
3.5.3	Consistency measures Filter-based Feature Selection	54
3.5.4	Dependency Measures Filter-based Feature Selection	54
3.6	Feature Wrappers	55
3.7	Feature Hybrids	57

3.8	Brief Description of some selected Filter-based Feature Selection Benchmark Selected for Evaluation of the Proposed IFS Method	60
3.9	Review of Ant Colony Optimization (ACO) and Multidimensional Knapsack Problem (MKP).....	64
3.9.1	Ant Colony Optimization.....	64
3.9.2	Multidimensional Knapsack Problem (MKP)	65
3.9.3	Formulation of ACO for the MKP	67
3.9.4	ACO Algorithm for MKP	70
3.10	Literature Review of the Related Information Theorem Feature Selection Methods.....	74
3.10.1	Review of Relevant, Irrelevant and Redundant Features.....	82
3.10.1	Research Gap Analysis	83
CHAPTER 4: INFORMATION THEORETIC-BASED FEATURE SELECTION		
	(IFS) METHOD	88
4.1	Rationale	88
4.2	Fundamental of Mutual Information (MI) Estimation.....	91
4.2.1	Ideas and Definition.....	91
4.2.2	Properties of Mutual Information	93
4.3	Formulation of Mutual Information Estimator	94
4.3.1	Basic Ideas	95
4.3.2	Conventional Kernel Based Density Estimation.....	96
4.4	Formulation of IFS Criterion Function Based on Mutual Information Estimation for Uncertain Data	97
4.5	IFS method.....	101

4.5.1	Algorithm for Stopping Criterion Function	105
4.5.2	Algorithm for Optimal Feature Subset Selection Function	107
CHAPTER 5: DATASETS USED IN EXPERIMENT		110
5.1	Domains	110
5.2	Experimental Methodology	114
CHAPTER 6: EVALUATION OF THE PROPOSED IFS METHOD		120
6.1	Evaluation of IFS Method with Artificial Datasets	121
6.1.1	Evaluation of IFS Selection Criterion Function (IFS_MI estimator)	121
6.1.2	Evaluation of IFS Method for Feature Selection using Artificial Dataset.....	123
6.2	Evaluation of IFS Method with Natural Domains	126
6.2.1	Proposed IFS Evaluation for Regression Tasks.....	127
6.2.2	Proposed IFS Evaluation for Classification Tasks.....	131
6.3	Comparing Proposed IFS Method with Some Other Feature Selection Methods.....	133
6.3.1	Experimental Setups for Evaluation of IFS Method with Five Other Feature Selection Methods.....	134
6.3.2	Comparison of IFS Method with Five Other Feature Selection Method.....	135
CHAPTER 7: HYBRID FEATURE SELECTION, IMPLEMENTATION AND EVALUATION		140
7.1	Binary Ant Colony System (BACS) for Multidimensional Knapsack Problem (MKP).....	141
7.1.1	Solution Representation and Construction.....	143
7.2	Hybrid Feature Selection Method.....	150
7.3	IFS_BACS Experimental Setups	154

7.3.1	Presentation and Discussion of IFS_BACS Results	157
7.4	Comparison of Filter Based IFS and Hybrid Based IFS_BACS Methods	158
CHAPTER 8: CONCLUSION		163
8.1	Summary	163
8.2	Conclusion	167
8.3	Future work.....	168
REFERENCES		169
APPENDICES		186

LIST OF TABLES

	Page
Table 2.1 Sample from Mammographic Mass Data	13
Table 2.2 Sample of Energy Efficiency Dataset.....	14
Table 2.3 Description of Energy Efficiency Dataset Sample	15
Table 2.4 Sample from NASDAC Composite Daily Closing Stock	19
Table 2.5 Gradient Descent Algorithm.....	35
Table 3.1 Hill-Climbing Algorithm	48
Table 3.2 Stochastic First-Ascent Hill-Climbing.....	49
Table 3.3 ACO Meta-heuristic Pseudocode.....	49
Table 3.4 Dynamic Programming Algorithm for the 0-1 Knapsack Problem.....	67
Table 4.1 The Proposed IFS Algorithm.....	104
Table 5.1 Details of the Datasets from UCI Repositories.....	110
Table 5.2 Some Details of the Dataset that Required Normalization and/or Discretization	116
Table 5.3 Details of Machine Learning Algorithm Parameters Settings.....	118
Table 6.1 % Appearance of each Feature in Selected Feature Subsets for 100 Runs.....	123
Table 6.2 Feature Set Before and After IFS Feature Selection Method is Applied For Regression Datasets.....	127
Table 6.3 The Machine Learning Models Performance Evaluation for Regression Tasks	129
Table 6.4 Performance Evaluation between Features before IFS Method and Features after IFS Method is Applied for Regression Task.....	130
Table 6.5 Feature Set Before and After IFS Feature Selection Method is Applied for Classification Datasets	131

Table 6.6	Performance Evaluation between Features Before IFS Method and Features After IFS Method is Applied for Classification Task	133
Table 6.7	Comparison of the Number of Features Selected in the Feature Subsets from the Datasets by the Feature Selection Algorithms	136
Table 6.8	Percentage Misclassification for the Six Feature Selection Algorithms.....	136
Table 6.9	Performance Comparison between IFS Method and Other Feature Selection Methods Based on the RMSE.	139
Table 6.10	Performance Comparison between IFS Method and Other Feature Selection Methods Based on the NMSE.	139
Table 7.1	BACS Algorithm for MKP	145
Table 7.2	Result of IFS_BACS for Classification Tasks.....	157
Table 7.3	Results of IFS_BACS for Regression Tasks	158
Table 7.4	Comparison of the Number of Features in Feature Subset for IFS and IFS_BACS Methods using Classification Datasets.....	159
Table 7.5	Comparison of the Number of Features in Feature Subsets for IFS and IFS_BACS Methods fusing Regression Datasets.....	160
Table 7.6	Comparison of the Performance Evaluation between IFS and IFS_BACS for Classification Tasks	160
Table 7.7	Comparison of the Performance Evaluation between IFS and IFS_BACS Methods for Regression Tasks	161

LIST OF FIGURES

	Page
Figure 2.1 The Big Picture: Workflow Diagram for Supervised Machine Learning Tasks.....	17
Figure 2.2 Linear Regression Model	23
Figure 2.3 Linear Regression Model Modified to Output 1/0	25
Figure 2.4 Logistic Regression Model.....	26
Figure 2.5 Shape of Cost Function for Logistic Regression.....	27
Figure 2.6 SVM Decision Boundary	29
Figure 2.7 Piecewise Cost Function of SVM: (a) is the Negative Case for $y = 0$ and (b) is Positive Case for $y = 1$	30
Figure 2.8 Simple Model of Neuron: (a) Without Bias, (b) With Bias Input x_0 and Weight θ_0 Explicitly Shown.....	32
Figure 2.9 Artificial Neural Network Architecture with Four Layers	33
Figure 3.1 Big Data Diagram.....	45
Figure 3.2 Filter Model Feature Selector.....	51
Figure 3.3 Wrapper Model Feature Selector.....	56
Figure 3.4 Hybrid (filter-wrapper) Model Feature Selector	58
Figure 3.5 Hybrid (filter-metaheuristic) Model Feature Selector	60
Figure 3.6 The Original Battiti's MIFS Algorithm.....	76
Figure 4.1 Learning System with Continuous Input Variable X and Continuous Output Variable Y	93
Figure 4.2 Algorithm for Stopping Criterion Function.....	106
Figure 4.3 Algorithm for Optimal Feature Subset Selection Function	109
Figure 5.1 Illustration of Coding Matrix of SVM Binary Learners	

for $k = 2$ up to $k = 5$	119
Figure 6.1 Comparison between IFS_MI Estimator and some of the Benchmarks using Ranges of Functional Relationships.....	122
Figure 6.2 Result of IFS Evaluation of Redundancy in Features	125
Figure 6.3 Classification Accuracy of the Six Feature Selection Methods Using: (A) Iris, (B) Breast Cancer, (C) Breast Tumor, (D) Glass, (E) Wine, (F) Energy Utilization (cooling load) and (G) Energy Utilization (heating load) Datasets	137
Figure 7.1 State Machine Representation of Ants Tour Construct for BACS.....	143
Figure 7.2 State Diagram Illustrating Tour Construction	144
Figure 7.3 Example Illustrating Binary Representation of a Tour Construction.....	144
Figure 7.4 Hybrid (IFS_BACS) Feature Selection Method Flowchart	151
Figure 7.5 Illustration of the Outputs from IFS and IFS_BACS Methods: (A) fcount array, (B) Processed Features and their Values, and (C) Optimal Feature Subset Selected by IFS_BACS.....	152
Figure 7.6 The Cost Curve for BACS.....	154
Figure 7.7 Comparison of the Percentage Prediction Accuracy between IFS and IFS_BACS Methods for Classification Tasks.....	161

CHAPTER 1

INTRODUCTION

As the quest for computers to be sufficiently intelligent like human (McCarthy & Hayes, 1987; Harnad, 2008) advances in the recent time, the exciting field of Machine Learning emerges (Langely, 2011) to provides a new capability for computers. Applications such as credit card fraud detection, spam detection, speech understanding, product recommendation, medical diagnosis, stock trading, handwritten recognition, natural language processing and other works in computer vision, all of which proved to be very difficult or nearly impossible to program by hands are now possible with the recent development of machine learning technologies. At the heart of the machine learning technologies is the process of learning from a feature set, selected from an available dataset. However, the mostly available dataset in various application areas contain features or attributes collected for some reasons other than mining the data, consequently, they contain replicate or irrelevant features to the predicting tasks (Fayyad et al., 1996), of great importance to the learning machine, is feature selection. Feature selection is an optimization method used to select optimal feature subset from the full or whole feature set. Selected optimal feature subset is expected to retain the relevant dependency information in the full feature set as a key focus of a learning algorithm to the predictable tasks. Thus, the hypothesis investigated in this thesis is that feature selection for supervised learning tasks can be achieved based on the dependency information between input features and the target attribute. Also, that such feature selection technique will be useful to wide range of common machine learning algorithms. A technique for Information Theoretic-based Feature Selection (IFS) is developed and evaluated using some popular machine learning algorithms and empirical methods. The selection criterion function is derived based on the popular information theory introduced by Shannon (Shannon, 1948). It is

simple to execute and can eliminate unnecessary features. In many cases, it improves the performance of learning algorithms and produces results that are comparable with state of the art feature selection methods from the available literature.

1.1 Background of the Study

Machine learning is an exciting field of study that evolved out of work in artificial intelligence (Langely, 2011). It brings together learning theories from computer science and statistics (Jordan, 2014) and primarily concerns with prediction tasks, based on known or learned features of a dataset. There are two types of machine learning prediction tasks. The first category is classification task which is a prediction task that classifies target object (Output Attribute) into one of the ordinal, nominal or categorical pre-labeled attribute known as class. For example, a machine learning task for medical diagnosis to predict if a patient has cancer or not. The second type of machine learning prediction tasks is the regression task which involves predicting target object as a quantitative or numerical approximation. For the same cancer example, for instance, the prediction task could be to predict the approximate size or spread of cancer in a patient.

Irrespective of the type of tasks, machine learning problems are divided into three broad categories. The first division is the supervised learning; this requires that the dataset should consist of input attributes or features and the corresponding target attribute. The goal here is that the learning system learns model parameters that maps inputs (unseen before) to outputs. The second category is the unsupervised learning, which involves unlabeled data (Dataset consist of only inputs features). The goal of unsupervised learning is to learn a hidden structure or pattern in a dataset. The third category of machine learning task is reinforcement learning, which requires a learning system to learn how to interact with its environment in a

dynamic way. The goal is that the learning system learns how to map situation to action like in the case of learning to play a game by playing against an opponent (Bishop, 2006).

State of the art machine learning algorithms, such as Artificial Neural Network (ANN), Support Vector Machine (SVM), Bayesian Network, Decision Tree Learning, Clustering, Evolutionary Algorithms, Reinforcement Learning and so forth have been explored and modeled to successfully solve a wider range of prediction problems in various areas of human endeavor. The performance of any learning algorithm to successfully predict an unseen data relies heavily on the nature of data attributes used for learning the model parameters. Various studies carried out in the literature indicated that using “Relevant” attributes or features to train a learning model results in compact models that generalized better (Kira & Rendell, 1992; Koller & Sahami, 1995; Guyon & Elisseeff, 2003; Pudil & Somol, 2005; Gadat & Younes, 2007). Feature selection algorithms are used to select feature subset in a preprocessing step. So, the chosen feature subset is then used for learning the model parameters.

Feature Selection techniques will continue to play an important role in machine learning and pattern recognition for many years to come (Guyon & Elisseeff, 2003; Collins & Liu, 2005; Hoi et al., 2012; Wang et al., 2013; Wu et al., 2014). Most especially, now that the world is currently facing challenges of big data analytics. Pending on the learning domain and processing, Feature Selection Algorithms consist of two broad categories namely the batch and online feature selection techniques. Batch feature selection methods fundamentally assume that all or fixed set of candidate features for learning are available in advance. Moreover, a feature selection method is to choose an m feature subset from a pool of n candidates, where m is expected to be less than the n . This choice is made by optimizing a criterion function over all subsets of the m (Collins & Liu, 2005). This category of feature

selection is suitable for standard or traditional learning process (Glocer et al., 2005; Wang et al., 2013). The online feature selection methods assume that features arrive in stages, and no new addition of instances to the problem (Glocer et al., 2005). That is, feature flows into a model one after the other dynamically and feature selection performs at the arrival time. This online feature selection differs from standard online training which assumes instances arrive dynamically (Hoi et al., 2012; Wang et al., 2013). This thesis focuses on batch feature selection for supervised learning tasks and has been studied for many years (Blum & Langley, 1997; Guyon & Elisseeff, 2003; Liu & Yu, 2005; Saeys et al., 2007).

Feature selection algorithms are believed to consist of two major components namely the Selection Criterion Function and the Search Strategy (Collins & Liu, 2005). The Selection Criterion Function is a quantitative measure which is used to compare one feature subset against another. The Search Strategy is a systematic method to enumerate candidate feature subsets and to decide when to stop (Collins & Liu, 2005). However, in this thesis, the author considered Stopping Criterion as a major component in addition to the two mentioned previously. While the Selection Criterion Function determines how to measure and quantitatively compare features and the Search Strategy provides the means to search for optimal feature subset through the feature subsets space, the Stopping Criterion determines if optimal or suboptimal feature subset is selected or attained. There are two categories of the Selection Criterion Functions which give rise to the three types of Feature Selection Models. Those that utilize a statistical and probability distribution of dataset attributes to measure relevance of feature subset and those that evaluate feature subset based on its performance with a learning machine model. The former referred to as Filter and the latter Wrapper. Hybrid model combines the strong points of both Filter and Wrapper.

1.2 Motivation

As the world is currently witnessing a sudden burst of data due to advances in technologies, the challenges of big data analytics become self-evident. As such the author of this thesis believes that effective feature selection methods will continue to play a role in uncovering most silent features in the data, that will make it easy for applying Machine Learning Algorithms to big data. Previous studies in (Kira & Rendell, 1992; Richeldi & Lanzi, 1996; Blum & Langley, 1997; Guyon & Elisseeff, 2003; Pudil & Somol, 2005; Chen & Lin, 2006; Rossi et al., 2006; Gadat & Younes, 2007; Doquire & Verleysen, 2011; Unler et al., 2011) highlighted that feature selection reduces the dimension of a dataset by selecting variables that are relevant to the predicting attribute; it helps to improve the predicting capability of machine learning models by building a compact model that often avoid over-fitting and generalized better, it can improve the accuracy of prediction because of reduction in estimation errors, and empirical results indicate that building good predictor model requires a decrease in feature subset. More so, feature subset selection reduces the burden of data collection as well as reducing computational complexity. The last two advantages of feature selection are of economic importance most especially in some domains that are capital-intensive and time-consuming to organize facilities and collect, or derived domain variables enough for a traditional empirical method of analysis. Feature selection will ensure the use of concise dataset to develop predictive model for a specific domain.

The two main types of feature selection criterion functions introduced in Section 1.1, each has its strong and weak points. Filter models are fast and select feature subset independent of any machine learning algorithm, as such it generalizes the use of selected feature subset by many machine learning algorithms. However, wrapper models are more efficient in choosing feature subset that gives better prediction model with a specific machine learning algorithm. Because

of wrapper mode of operation, requires searching through feature subset space and using a specific machine learning algorithm as more or less a Blackbox to select a feature subset with better predictive performance with the machine learning model. By searching for the feature subset in subset space, train and test a machine learning, wrappers incur additional computational overhead. Also, a selected feature subset using wrapper may have higher prediction accuracy with the intended machine learning algorithm, but may perform woefully with a different machine learning algorithm. That is, each machine learning algorithm requires researching through the feature subset space to find its optimal feature subset.

In this thesis, the author is of the opinion that the advantages of filter outpace its disadvantages. Also, combining the strength of filter (based on its greedy heuristic approach to search fast through the feature subsets space) and the power of wrapper (based on its use of a meta-heuristic algorithm) result in efficient feature selection model that is relatively fast and have less computational overhead. In this regard, this thesis explored both Filter and Hybrid (filter-metaheuristic) approaches.

Previous works in Machine Learning Feature Selection, focus more on classification tasks rather than regression, see Guyon and Elisseeff (2003). However, many domains require predicting quantitative variables; such problems are found in various fields including the oil & gas, pharmaceutical, food and textile industries (Rossi et al., 2006; Sulaiman & Labadin, 2015a). The reason for focusing more on classification task can be connected to the fact that it is not clear or rather very difficult to estimate selection criterion function such as estimation of Mutual Information between dataset with both continuous input variables and continuous target variable (Guyon & Elisseeff, 2003; Unler et al., 2011). The usual practice is to discretize continuous variable into discrete or apply other methods of projection (Unler et al, 2011), such transformation will preclude one to consider redundancy between features

(Doquire & Verleysen, 2011) since redundancy in features are subject to feature instances/values. This thesis focuses on investigating Information theoretic based feature selection methods for both classification and regression supervised learning tasks with more emphasis on regression problems.

Previously, various stopping criteria used in the literature do not have a scientific justification. For an instance, one might stop adding or removing attributes when none of the alternatives improves the estimate of classification accuracy. One might continue to revise the feature set as long as accuracy does not degrade. Alternatively, one might continue generating candidate sets until reaching the other end of the search space and then select the best (Blum & Langley, 1997). In another form, features are ordered according to relevance score then use a parameter to determine the breakpoint (Blum and Langley, 1997). This thesis investigated a stopping criterion function based on a scientifically sound idea.

1.3 Thesis Statement

This thesis claims that an efficient Feature Selection Algorithm for a supervised learning task can be achieved based on a dependency information between features and a target attribute. To be specific, this thesis investigates the following hypothesis:

A relevant feature subset is one in which the dependency information between each feature in the subset and a predictive output is high. Yet with minimal redundancy among features in the subset.

To investigate this hypothesis, the author formulates Feature Selection Criterion Function based on Mutual Information and create a Feature Selection Algorithm that evaluates the relevant of feature sets and eliminate redundancy in feature subset to minimal level. Chapter 4 detailed the Implementation of Information Theoretic-based Feature Selection (IFS). IFS used

Mutual Information (MI) as its evaluation strategy, MI which is an information theoretical concept that quantitatively measures the amount of dependency information two random variables have about each other. Unlike correlation coefficient that measures linear dependency only. MI measures both linear and nonlinear functional dependencies between variables. A property that made it a popular choice for feature selection (Battiti, 1994; Peng et al., 2005; Rossi et al., 2006; François et al., 2007). Since from the past two decades, majority of previous formulation of Mutual Information Estimator assumed classification task, that is, a target variable is assumed to contain class labels only. Thus, any Regression task requires a transformation of the output variable into classes before the application of a Feature Selection Algorithm (Kwak & Choi, 2002b; Verleysen et al., 2009; Ulner et al., 2011). However, even though the proposed IFS require that a target variable first discretized/partitioned into partition set in an unsupervised manner, in principle, it does not need any form of data transformation.

The proposed IFS method checked for redundancy between features and can be used for both supervised classification and regression tasks. The proposed IFS has both filter and hybrid IFS versions, both versions are straightforward and fast to execute, though the proposed hybrid IFS version incurred additional computational overhead, it is less when compared to additional overhead incurred by wrapper only. Various experiments in chapter 6 show that the proposed IFS can identify relevant, redundant and irrelevant features when strong and moderate information dependency exist. However, the new IFS method will fail to select all relevant features when there are redundancies among features.

1.4 Objectives

- i. To propose a methodology to estimate mutual information (MI) for both discrete and continuous variables, suitable for feature selection for both classification and regression tasks.
- ii. To create or design feature selection algorithm based on information theorem concepts that can eliminate redundancy within feature subset to minimal level without additional overhead.
- iii. To propose hybrid filter based feature selection that will be relatively fast with less computational overhead compared to Wrapper.

1.5 Scope of the Thesis

The main focus of this thesis is to create information theoretic based feature selection suitable for both classification and regression supervised machine learning tasks. The feature selection will be using a proposed methodology that will allow direct estimation of mutual information from regression and classification datasets. The greedy search procedure for the feature selection will be an improved forward procedure with scientifically sound stopping criteria which will be based on the idea of null hypothesis and resampling method. Also, the improved forward procedure will be designed to minimizing redundancy in feature subset in an effective way. This thesis will not involve wrapper, and will be suitable for dataset with few features. But to improve on this limitation, the study will investigate a hybrid of filter method, with metaheuristic ant colony optimization system as part of the thesis. This is to investigate any improvement in terms of further reduction of feature subset and increase performance in the trained machine learning model.

1.6 Thesis Overview

Chapter 2 presents concepts and definition of terms for supervised machine learning. Also, it provides the detailed review of discretization methods.

Chapter 3 presents a literature survey of feature selection for machine learning. It discusses the distinction between feature selection and feature extraction, reviewed online feature selection for big data and reviewed both filter and wrapper feature selection. Finally, conducted a related research literature review and research gap analysis.

Chapter 4 presents Information theoretic-based feature selection (IFS) method, starting with the rationale for this approach, reviewing mutual information concept and properties, formulation of IFS selection criterion function and finally, presentation of IFS algorithm.

Chapter 5 describes the datasets and experimental methodology used in the various experiments as discussed in Chapter 6 and Chapter 7.

Chapter 6 evaluates IFS and its components using both empirical method and machine learning algorithms performance as a metric. Finally, presents the evaluation of IFS relative performance with some feature selection algorithm benchmarks.

Chapter 7 presents and evaluate IFS hybrid feature selection method and compare results obtained with the IFS filter based method.

Chapter 8 presents conclusions and suggest future work.

CHAPTER 2

SUPERVISED MACHINE LEARNING: CONCEPT AND DEFINITION

Machine learning is a scientific discipline that learns to recognize complex patterns and make intelligent decisions based on data (Kappen & Kok, 2011). When a learning process involves both input variables (features) and the corresponding target variable, and a learning algorithm learns model parameters by comparing the predicted output value (by the algorithm) with corresponding desired output variable (given), this is referring to as *Supervised learning*. This process of learning which requires the algorithm to learn based on the given target variable is known as the *training process*. The training process aimed at minimizing modeling error—which is the difference between the predicted value and the actual training value (given). This process is often repeated on each pair of a training instance (a pair of input example and desired output value) until it found the optimal model parameters. This training process allows the resulted machine learning model to predict output value when given a new input instance. However, the successes of finding optimal parameters of a machine learning model depends on the nature of the dataset provided for training (Richeldi & Lanzi, 1996; Doquire & Verleysen, 2011; Unler et al., 2011). When the training examples are adequate and relevant, this will have effect in finding optimal parameters of the machine learning model and accurate prediction of a new input data.

2.1 Machine Learning Tasks

The main task of a supervised machine learning is to predict. In this regard, there are two categories of tasks for supervised machine learning. When prediction task involves the desired output, which is label into classes, it referred to as classification task. However, when

prediction task involves the output attribute which is approximate numeric values, this is known as regression task.

2.1.1 The Classification Task

Supervised Machine Learning Classification Task usually involves dividing the input instances into two or more classes based on the desired output. The targeted output values depend on the nature of the domain in which the data comes. It is ordinal values/numbers that tend to qualitatively characterizes an object. For example, in a breast tumor dataset that includes a binary output of an ordinal value that infer if an instance of any input variables from the given breast tumor dataset is malignant (dangerous/harmful) or benign (not malignant) tumor. The described example is known as *classification task* or *binary classification problem*. Also, it is a supervised learning task since it includes an ordinal desired output values that divide input instances into two classes – malignant or benign. However, if the given breast tumor dataset contains only instances of input variables, but no desired output. So instead, the learning algorithm is expected to find patterns (after training process has attained optimality) that will divide the data into subgroups or classes based on input variables only. This learning task is known as *unsupervised learning*.

2.1.2 The Regression Task

Supervised Machine Learning Regression Task involves a quantitative numeric or continuous variable as the desired output. For the same breast tumor data, instead of including ordinal values that indicate malignant or benign, if the approximate size or spread of tumor cells is the desired output variable. Then the learning task becomes a *regression problem*.

Each of the two types of tasks has its characteristic information and processing approach by different learning algorithms (Slovic & Lichtenstein, 1971).

2.2 Data Representation

In a supervised machine learning task, data representation is in a tabular or matrix form. In this arrangement, each row is an instance or example which may contain output value for the input instance. Each column is a feature (also known as attribute or variable), and the last column may contain output variable. Each input features could be ordinal, nominal or numerical data. The numerical data could be discrete or continuous real values.

Table 2.1: Sample from Mammographic Mass Data

BI-RADS assessment	Age	Shape	Margin	Density	Severity
5	67	3	5	3	1
4	43	1	1	?	1
5	58	4	5	3	1
4	28	1	1	3	0
5	74	1	5	?	1
5	57	1	5	3	1
5	76	1	4	3	1
3	42	2	1	3	1
4	36	3	1	2	0
4	60	2	1	2	0
4	54	1	1	3	0
3	52	3	4	3	0
4	59	2	1	3	1
4	54	1	1	3	1

Table 2.1 is a sample of Breast cancer screening dataset obtained from the UCI repository. It had 961 Instances and associated with Classification task, details of which can be found in Elter et al. (2007). The first and fifth features (BI-RADS Assessment and Density) are ordinal data, the third, fourth and last attribute (Shape, Margin, and Severity) are nominal and the second attribute (Age) is numerical data. Severity is the desired output variable, and it is a binomial data with 0 – indicating Benign and 1 – indicating Malignant.

Table 2.2: Sample of Energy Efficiency Dataset

X1	X2	X3	X4	X5	X6	X7	X8	y1	y2
0.98	514.50	294.00	110.25	7.00	2	0.00	0	15.55	21.33
0.98	514.50	294.00	110.25	7.00	3	0.00	0	15.55	21.33
0.98	514.50	294.00	110.25	7.00	4	0.00	0	15.55	21.33
0.98	514.50	294.00	110.25	7.00	5	0.00	0	15.55	21.33
0.90	563.50	318.50	122.50	7.00	2	0.00	0	20.84	28.28
0.98	514.50	294.00	110.25	7.00	2	0.10	1	24.58	26.47
0.98	514.50	294.00	110.25	7.00	3	0.10	1	24.63	26.37
0.98	514.50	294.00	110.25	7.00	4	0.10	1	24.63	26.44
0.98	514.50	294.00	110.25	7.00	5	0.10	1	24.59	26.29
0.90	563.50	318.50	122.50	7.00	2	0.10	1	29.03	32.92
0.98	514.50	294.00	110.25	7.00	2	0.10	2	24.29	25.95
0.98	514.50	294.00	110.25	7.00	3	0.10	2	24.31	25.63
0.98	514.50	294.00	110.25	7.00	4	0.10	2	24.13	26.13
0.98	514.50	294.00	110.25	7.00	2	0.10	3	24.28	25.68
0.98	514.50	294.00	110.25	7.00	3	0.10	3	24.40	26.02
0.98	514.50	294.00	110.25	7.00	4	0.10	3	24.11	25.84
0.98	514.50	294.00	110.25	7.00	5	0.10	3	24.35	26.14
0.98	514.50	294.00	110.25	7.00	2	0.10	4	24.38	25.91
0.98	514.50	294.00	110.25	7.00	3	0.10	4	24.23	25.72
0.98	514.50	294.00	110.25	7.00	4	0.10	4	24.04	26.18
0.98	514.50	294.00	110.25	7.00	5	0.10	4	24.32	25.87
0.98	514.50	294.00	110.25	7.00	2	0.10	5	24.35	25.64
0.98	514.50	294.00	110.25	7.00	3	0.10	5	24.33	25.98
0.98	514.50	294.00	110.25	7.00	4	0.10	5	24.03	25.88
0.98	514.50	294.00	110.25	7.00	5	0.10	5	24.26	26.18

Table 2.2 is a sample of Energy Efficiency Dataset from UCI Dataset Repository. It contains eight input features $X1$ to $X8$ and two distinct output attributes $y1$ and $y2$, both of which are regression values. Table 2.3 is the description of each attribute, and the details of the dataset are available in Tsanas and Xifara (2012). However, in Tsanas and Xifara (2012) $X1$ to $X8$ are the input features used to train two predicting models each for $y1$ and $y2$ separately. The dataset had 768 instances and associated with both classification and regression tasks. When an output attribute is a continuous variable, the common way is to first discretize the variable before applying a classification method. The reason for doing so could be either due to the convenience of a classification learning algorithm or due to the needs of the domain

application. For example, in Table 2.2 a range of values of y_1 , y_2 or both may have a meaningful nominal output description such as high, medium and low as defined by a domain expert.

Table 2.3: Description of Energy Efficiency Dataset Sample

X1	Relative Compactness
X2	Surface Area
X3	Wall Area
X4	Roof Area
X5	Overall Height
X6	Orientation
X7	Glazing Area
X8	Glazing Area Distribution
y1	Heating Load
y2	Cooling Load

While some domain will have as many as millions of features as input (e.g. in computer vision) to a learning algorithm, time domain (e.g. in time series prediction) may require a single feature input to a learning algorithm. That is, if X of length n is a time series, with $x_1, x_2, \dots, x_n$ as instances of X feature. Table 2.4 presents a sample from NASDAC (2011). The dataset is the daily closing stock between 5th February 1971 to 27th October 2011. Assuming there are data recorded daily over several months and the interest is to predict one day (value) in advance. In this case, presentation of data instances to a learning algorithm will follow certain patterns. Assuming there are 30 data point values, $x_1, x_2, \dots, x_{30}$ and using some visualized data analysis, some of the recorded daily values correlated vaguely with the past four day's data. Each training pattern will be five values as follows:

Pattern 01: x_1, x_2, x_3, x_4, x_5

Pattern 02: x_2, x_3, x_4, x_5, x_6

...

Pattern 26: $x_{26}, x_{27}, x_{28}, x_{29}, x_{30}$

Where the first four correspond to input nodes and the fifth value is the desired value for the output node.

Finally, feature selection and feature scaling are common practice in machine learning and are considered as important preprocessing steps. Before the training process commences, feature normalization into similar scale takes place. Moreover, it is useful for training performance.

2.3 Preprocessing Steps

Data preprocessing is an essential step to consider before applying machine learning algorithms to train a predictive model. Raw data collected for machine learning modeling are usually not in good shape and format ready to be used by machine learning algorithms. That is why data mining experts believe that data preprocessing is about 80% of the time spent in data mining activities (Paul et al., 1999). Raw data may have missing data, variation in scale, presence of a very large data set and inappropriate format. Each of these issues affects machine learning algorithms in different ways and different preprocessing steps are used to correct the data into good shape ready to be used. Below is the brief discussion of preprocessing steps to correct the above mentioned issues. Figure 2.1 presents the workflow diagram for supervised machine learning tasks.

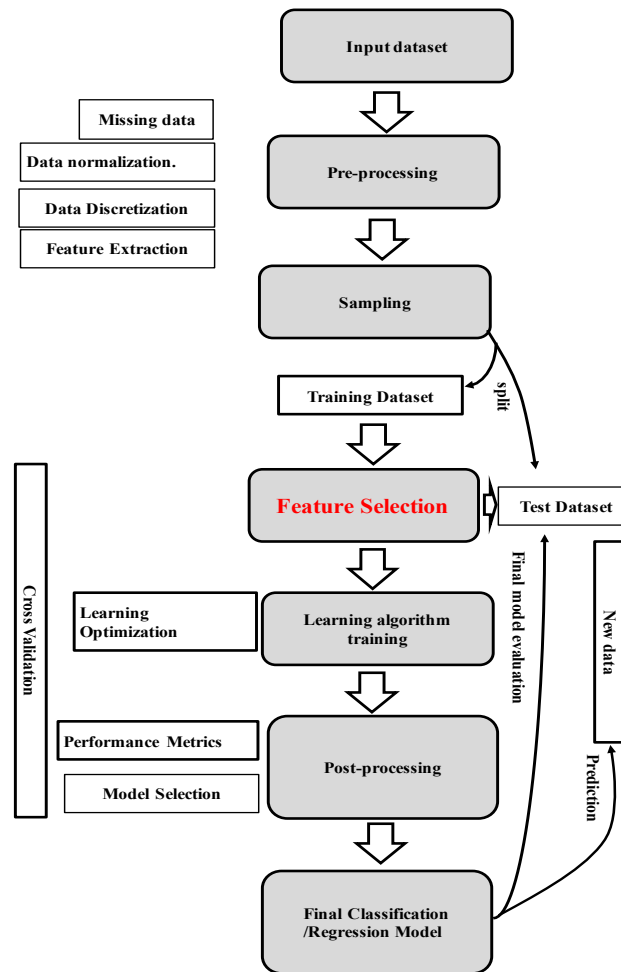


Figure 2.1: The Big Picture: Workflow Diagram for Supervised Machine Learning Tasks¹

2.3.1 Data with Missing Values

Machine Learning Algorithms such as Neural Networks and Support Vector Machine will not work properly if there are missing data. Data with missing values needs **cleaning**. Cleaning data involves removing or fixing values and this method works only when the missing values are sparse (fewer in numbers). The first strategy, removes instances (rows) with missing values from the data samples, or occasionally, removes attribute (column) with missing values from the data set if the attribute is not critical to predicting problem. Alternative strategy for handling missing data involve fixing values, either by the use of descriptive statistics such as

¹ The workflow diagram is a modification of the original idea by Sebastian Raschka 2014, licensed under a Creative Common Attribution 4.0 International license. URL: http://sebastianraschka.com/Articles/2014_intro_supervised_learning.html

mean, median, minimum, maximum or zero to replace missing values or a better option involves imputation using k-nearest neighbor. To impute a missing value, attribute (column) with missing values is considered as an intermediate target attribute and then using instances (rows) for which values are not missing to predict the missing values.

2.3.2 Data Scaling

Data set may contain varying scales for different attributes. Many machine learning algorithms are sensitive to the scale of data. Attribute whose range of quantitative values spans millions will influence the outcomes of a machine learning algorithm when compare with another attribute that is in a few tens.

Table 2.4: Sample from NASDAC Composite Daily Closing Stock

Close
100
100.84
100.76
100.69
101.45
102.05
102.19
101.74
101.42
100.7
99.68
99.72
100.64
101.23
101.34
101.78
101.84
102.07
102.78
103
104.23
104.41
104.14
104.26
104.5
105.13
105.56
105.5
105.88
105.77
105.37
105.11
104.7
104.6
105.07
105.2

Scaling techniques such as normalization (min-max scaling) and z-score normalization (unit-variance scaling) are used to eliminate or minimize bias likely to be introduced by variation in scales. Scaling techniques transform all attributes into a common range.

2.3.2.1 Normalization (Min-Max Scaling)

Min-Max scaling is done by dividing each attribute by its largest value:

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (2.1)$$

This technique scale attribute in a certain range, example 0 to 1 or 0 to 3.

2.3.2.2 Z-score Normalization (Unit-Variance Scaling)

Z-score normalization is also called standardization. It is another way of scaling attribute in such a way that each attribute will have the properties of a standard normal distribution (zero mean ($\mu = 0$) and a unit variance ($\sigma = 1$)).

Unit-Variance scaling calculates the difference between each attribute value and mean value of the attribute and divided by the standard deviation value of the attribute:

$$z = \frac{x - \mu}{\sigma} \quad (2.2)$$

2.3.3 Data Transformation

Machine Learning Algorithms differ in terms of their manner of working, while some algorithms work on certain value types of some attribute, others do not. For example, data set with categorical values as the desired output, are not suitable for regression predictive models. Likewise, if the desired output involved quantitative numeric values, it cannot be used with a classification algorithm like logistic regression which works with binomial type attribute. These conditions require the use of data transformation. Common data transformations are scaling, data discretization (see Section 2.7 for detail discussion), attribute decomposition (splitting complex attribute into constituent parts) and attribute aggregation (derivation of attribute from two or more variables).

There are some other situations where the actual attributes or features are not readily available from raw data (example in image processing), such situation requires data transformation known as feature extraction. Section 3.1 highlights more on feature extraction.

2.3.4 Sampling

There may be a situation where you have a very large dataset to work with. This means that there is a need for more running times for algorithms and in-memory computational requirements. Even though the area of big data analytics addresses this issue, the big data framework may not be readily available. Sampling method has been the traditional way of reducing the computational overhead. Sampling method involves random selection of a portion of data set to represent the data space. The randomly selected portion of the data set is further randomly split into training and testing sets. The training data is then used with a machine learning algorithm to train a predictive model and subsequently, the test data set is used for testing the prediction error and avoid overfitting.

2.4 Machine Learning Models

Machine learning models are predictive models that utilize learning theories from computer science and statistics to predict outcomes, Seymour (1993). Each learning algorithm conceptually describes each learning model. In supervised learning, learning algorithm builds prediction hypothesis or model (known as a predictor) by learning optimal set of parameters for the corresponding inputs to the hypothesis. Until recently most of the learning models are known to handle only one type of learning task effectively. For example, Support Vector Machine (SVM) and decision trees are classifier models while Linear regression and its variation are estimator/predictor models. However, with kernel trick and modification of learning hypothesis, decision tree – with a categorical variable in its final leaf nodes is classification tree, while regression tree is designed to have continuous or numerical

approximate values at terminal leaf nodes. The same trend is seen in Support Vector Machine (SVM) versus Support Vector Regression (SVR) and Linear Regression versus Logistic Regression.

To test the effect of applying feature selection to a dataset, this thesis used three machine learning algorithms to train two categories of predictive models. Predictive models before applying feature selection to dataset and predictive models on subset datasets. The three machine learning algorithms include Linear Regression (multi-linear), Support Vector Machine (SVM) and Artificial Neural Networks (ANN). All the three are widely used in machine learning applications and have proven to be effective in practice.

2.4.1 Model Representation and Cost Function

This section presents model representation for linear least squares as a regression model because of its strong basis in statistical data analysis, its generic nature in the representation of many machine learning models and it is mathematically tractable. There are twofold to primary characteristics of knowledge representation – what information is made explicit and how the information is encoded physically for subsequent use (Haykin, 2009). In the real-world application of machine learning, a good solution depends on a good representation of models (Woods, 1986).

All supervised machine learning models consist of the following properties represented using notations:

- a. m – as a total number of training examples (observations).
- b. X 's – “input” variables / features
- c. y 's – “output” variables/ “target” variables.

- d. ' h ' – hypothesis expressed as a function of input variables X 's and unknown parameters θ . ' h ' can also be referred to as machine learning model or function.

The overall concept of training in supervised machine learning modeling is how to represent h as a function of X 's and θ ?

Without loss of generality, consider a Linear Regression with one input variable X :

Hypothesis: $h_{\theta}(x) = \theta_0 + \theta_1 x$ (2.3)

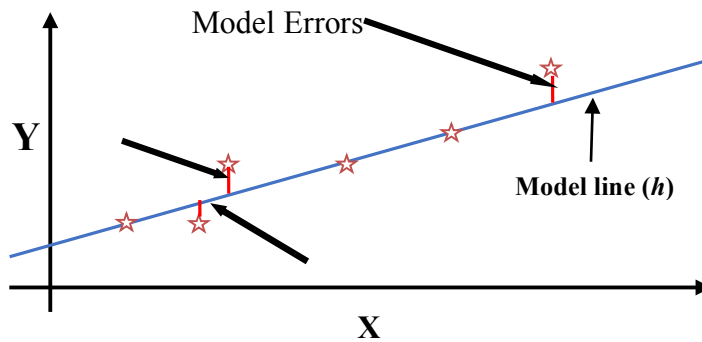


Figure 2.2: Linear Regression Model

Figure 2.2 represents a machine learning linear regression model expressed in Equation (2.3). Linear regression tried to find the line of the best fit (the model line h) in a training set. Though it may not be a perfect fit, however, the accuracy of the fit depends on the model parameters. The blue line in Figure 2.2 is the model line plotted against the actual data (stars in red), the red lines from the actual data points to the model line indicate the difference between the model predicted value and the actual training value and that is the model error. The goal of a machine learning algorithm is to find model parameters that will minimize the model sum of squared errors known as the **cost function**. Looking at the Equation (2.3), it is clear that the model parameters of h function is the slope (θ_1) and intercept at y-axis (θ_0).

Equation (2.4) expresses the cost function for linear regression:

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^i)^2 \quad (2.4)$$

Where “ T ” is the training instance or example, “ m ” the total number of training instances, $h_{\theta}(x^{(i)})$ is the model predicted value (approximate value of y), and y^i is the training desired value (the actual value from the training data). Since J is a function of the model parameters, each time the model parameters (slope and intercept at y-axis) change, it gives a new model (different regression line), different from the previous one. Subsequent section will explain how to optimize the process of finding model parameters that minimize the cost function.

To adopt the Linear Regression Model to Classification task, as in the case of Breast tumor example, the hypothesis is modified to the *Logistic Regression Model*. The output of the linear regression from Equation (2.3) (*i.e.* h_{θ}) is a continuous value. However, for the tumor example it is binomial type, either the output is 1 (for malignant) or 0 (for benign). So, using $h = \theta \cdot X$, the line of fit will look like in Figure 2.3.

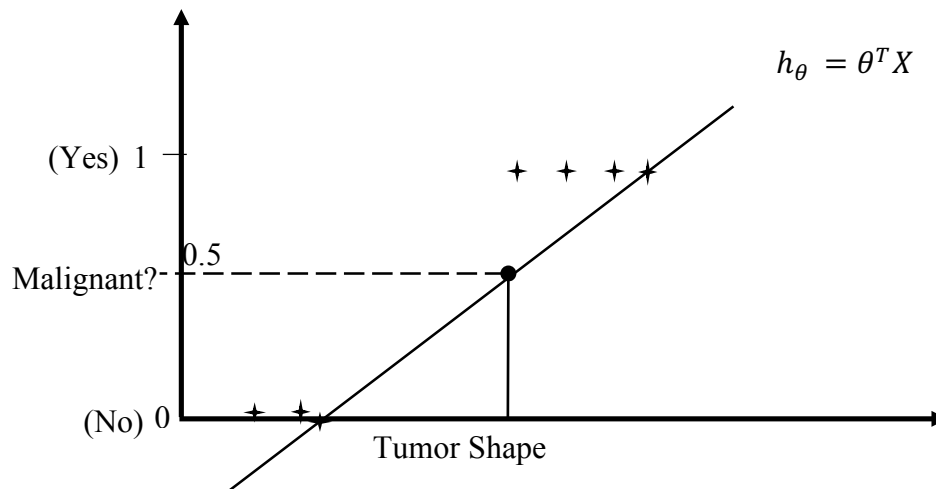


Figure 2.3: Linear Regression Model Modified to Output 1/0

With a threshold value of 0.5 and assuming that: if $h_{\theta} \geq 0.5$, *predict 1* and if $h_{\theta} < 0.5$, *predict 0*. The model is adjusted to output 1/0 values. However, a closer look at Figure 2.3 it is evident that with a further observation of say large tumor data, surely the regression line will change slightly and this will make the Linear Regression have values which are greater than one as well as a value that is less than zero. This scenario will affect the scope of the problem which is defined by 1/0 values.

However, using a Logistic Regression Model with hypothesis between 0 and 1 (as in Figure 2.4) will address the above problem.

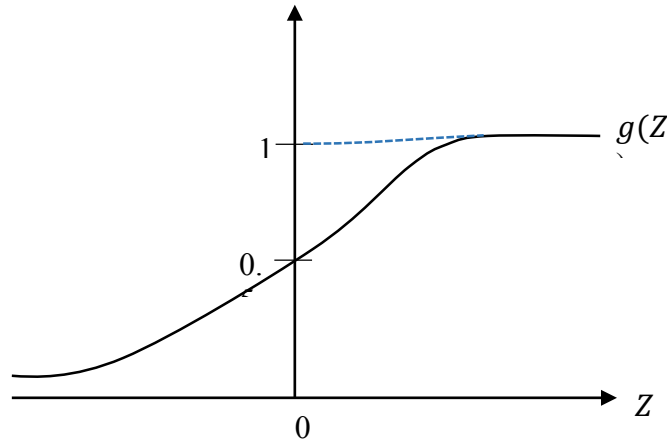


Figure 2.4: Logistic Regression Model

Logistic Regression Model output is $0 \leq h_{\theta} \leq 1$, given that the hypothesis is:

$$h_{\theta}(x) = g(\theta^T X) \quad (2.5)$$

Where both θ and X are vectors, and 'g' is a Logistic Function defined as:

$$g(Z) = \frac{1}{1+e^{-Z}} \quad (2.6)$$

Combining Equation (2.5) and Equation (2.6), the hypothesis of Logistic Regression Model becomes:

$$h_{\theta}(x) = \frac{1}{1+e^{-\theta^T X}} \quad (2.7)$$

From Equation (2.7), $h_{\theta}(x)$ for Logistic Regression is the *Estimated Probability* that $y = 1$ on X input and parameterized by theta as:

$$h_{\theta}(x) = P(y = 1|x; \theta) \quad (2.8)$$

The decision boundary is restricted to 0 from below and 1 from up. Thus $y = 1$ is predicted if $h_{\theta}(x) \geq 0.5$, and $y = 0$ is predicted if $h_{\theta}(x) < 0.5$.

The Cost Function for Logistic Regression Model is different from the Linear Regression Cost Function expressed in Equation (2.4). In practice when Equation (2.7) is substituted into Equation (2.4), it gives a non-convex function (function with many local minimum). This will prevent the optimization algorithm from finding the global minimum. Equation (2.9) is the expression of Cost Function for Logistic Regression Function:

$$J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^m \text{Cost}(h_{\theta}(x^{(i)}), y^i) \quad (2.9)$$

where:

$$\text{Cost}(h_{\theta}(x), y) = \begin{cases} -\log(h_{\theta}(x)), & \text{if } y=1 \\ -\log(1-h_{\theta}(x)), & \text{if } y=0 \end{cases} \quad (2.10)$$

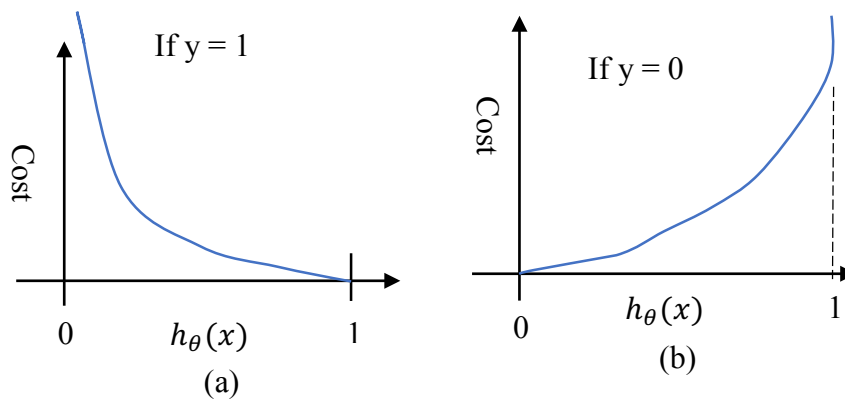


Figure 2.5: Shape of Cost Function for Logistic Regression

Figure 2.5 provides the shape of the cost function for Logistic Regression for each case. Figure 2.5 (a) is when $h(x) = 1$ and $y = 1$, in this case, the cost is 0 – the model made an accurate prediction. However, as the model prediction started to approach 0 while $y = 1$, the cost will increase to infinity when the model had a prediction of 0 ($h(x) = 0$). The opposite is the case in Figure 2.5 (b). When $h(x) = 0$ and $y = 0$ the cost is 0 – the model made an accurate

prediction, but when model prediction approaches one while $y = 0$, the cost becomes bigger till infinity when the model had a prediction of 1 ($h(x) = 1$).

With these explanations, the cost function has logarithmic behavior and can be simplified as in Equation (2.11):

$$Cost(h_{\theta}(x), y) = -y * \log(h_{\theta}(x)) - (1 - y) * \log(1 - h_{\theta}(x)) \quad (2.11)$$

Since y takes only 1 or 0 values, Equation (2.11) takes care of the two cases expressed in Equation (2.10). If $y = 1$, the 2nd term (right-hand side of Equation (2.11)) will be 0 equivalent to case 1 in Equation (2.10) and when $y = 0$, the 1st term is 0. Thus, Equation (2.12) is the complete Cost Function for Logistic Regression Model:

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^m y^i * \log(h_{\theta}(x^i)) + (1 - y^i) * \log(1 - h_{\theta}(x^i)) \quad (2.12)$$

2.4.1.1 Support Vector Machine (SVM)

Support Vector Machine (SVM) is a feed forward learning algorithm which sometimes gives a cleaner solution to complex nonlinear problems than logistic regression and ANN. Central to the construction of SVM is a small subset of data points extracted during the learning process from the training sample. These small subset of data points are the support vector. So, SVM learning algorithm is an inner-product kernel between a “Support vector” x_i and a vector x drawn from the input data space (Haykin, 2009). The Kernel trick basic to the design of a support vector machine is always optimal. SVM is sometimes called Large Margin Classifier, and this is because it always looks for a decision boundary with the widest margin from the training dataset.

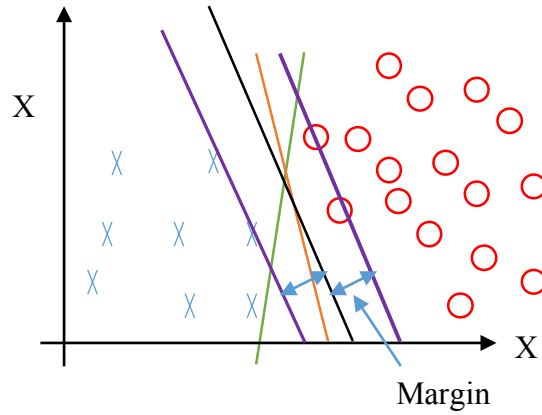


Figure 2.6: SVM Decision Boundary

Figure 2.6 illustrates some decision boundaries, both green and orange lines are decision boundaries representing some logistic regression, but the black line is a decision boundary of an SVM which gives wide margin compared to any other decision boundaries.

Modifying the logistic regression cost function in Equation (2.12) to include regularization parameter, gives regularized cost function as in Equation (2.13):

$$\min_{\theta} J(\theta) = \frac{1}{m} [\sum_{i=1}^m y^i \text{cost}_1(z) + (1 - y^i) \text{cost}_0(z)] + \frac{\lambda}{2m} \sum_{j=1}^n \theta_j^2 \quad (2.13)$$

Where $Z = \theta^T X$, $\text{cost}_1(z) = -\log \frac{1}{1+e^{-z}}$, for $y=1$ and $\text{cost}_0(z) = -\log (1 - \frac{1}{1+e^{-z}})$, for $y=0$. λ is the regularization parameter. Regularization of Cost Function tends to decrease the magnitude of model parameters without reducing the features. It is a useful measure of avoiding over-fitting.

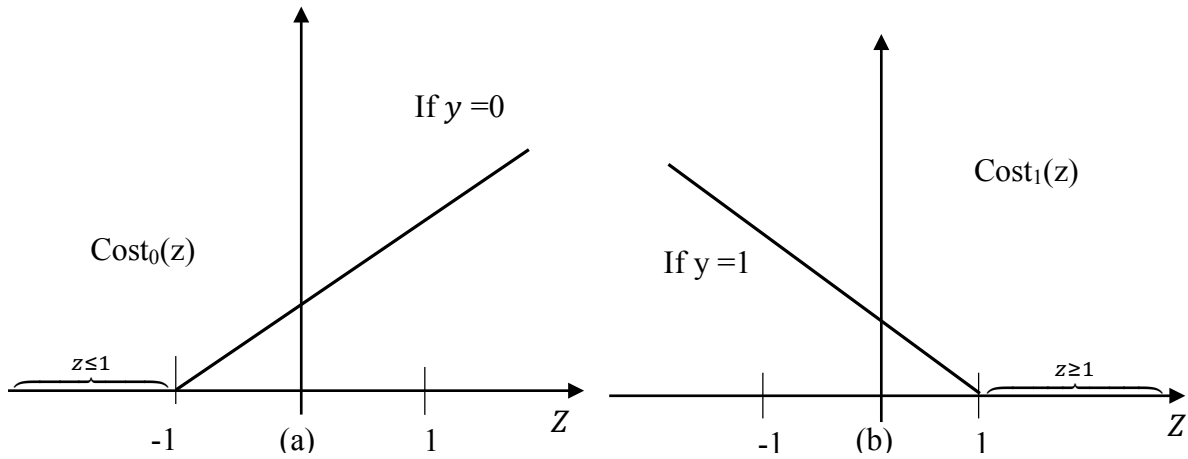


Figure 2.7: Piecewise Cost Function of SVM: (a) is the Negative Case for $y = 0$ and (b) is Positive Case for $y = 1$

With some modification of logistic regression Cost Function in Equation (2.13), we have the cost function for SVM. Since $1/m$ is a constant, ignoring it and introducing C as $1/\lambda$ will not affect the cost function, rather it gives a convex minimization cost function for SVM as:

$$\min_{\theta} J(\theta) = C \sum_{i=1}^m [y^{(i)} \text{cost}_1(z) + (1 - y^{(i)}) \text{cost}_0(z)] + \frac{1}{2} \sum_{j=1}^n \theta_j^2 \quad (2.14)$$

And the hypothesis of SVM is:

$$h(x) = \begin{cases} 1, & \text{if } Z \geq 0 \\ 0, & \text{if } Z < 0 \end{cases} \quad (2.15)$$

The Cost Function looks like Figure 2.7, with perfect prediction (cost = 0) when Z is greater than or equal to 1 (in the case of $y = 1$) or Z is less than or equal to -1 (in the case of $y = 0$). So, it gives extra safety margin factor for the SVM. With the minimization problem in Equation (2.14), when C is big value, then the chances are that θ will be choosing so that to minimize the first term of the cost function close to zero. Thus $\text{cost}_1 = 0$, whenever $y = 1$ and $\text{cost}_0 = 0$ whenever $y = 0$. It implies that cost (Z) will approach zero with the value of θ which

will make $\theta^T X \geq 1$ or $\theta^T X \leq -1$. So, if the entire first term of the minimization problem becomes zero, the new minimization problem will be:

$$\min_{\theta} J(\theta) = \frac{1}{2} \sum_{j=1}^n \theta_j^2 \quad (2.16)$$

Such that $\theta^T X \geq 1$, if $y^{(i)} = 1$ or $\theta^T X \leq -1$ if $y^{(i)} = 0$. Solution to this minimization problem gives decision boundary of the SVM with a large margin.

2.4.1.2 Artificial Neural Networks (ANN)

ANN algorithm developed through simulating human neurons or network of neurons in the brain. Neurons are cells with cell body and dendrites which are the input wires connected to the cell body and receive signals from body receptors. In addition to these two, neurons have output wire known as axon which sends signals or informative messages to other neurons. Thus, at a fundamental level, a neuron is a computational unit that receives some inputs (electric pulses) through its input wires (dendrites), does some computation in the cell body and sends output through its axon to other neurons in the brain.

The artificial neural network (ANN) derives its computing ability through its massively parallel distributed structured and through its power to learn and therefore generalize. These two capabilities make it possible for ANN to find good approximation solution for intractable complex problems, Haykin (2009). Among the features of ANN that made it attractive classification or prediction model in many fields of research are:

- (1) **Nonlinearity**: It can solve complex nonlinear problems.
- (2) **Input-Output Mapping**: The network learns from examples based on a popular paradigm known as supervised learning by constructing an input-output mapping of the problem at hand.

(3) **Adaptively**: this is a built-in power of neural networks to adapt their weights to changes in the surrounding environment. ANN trained to operate in a specific environment can be easily retrained to deal with minor modifications in the operating environmental conditions (Haykin, 2009).

Figure 2.8 depicted ANN implementation of a simple model of a neuron. The empty dashed circle plays a role analogous to the body of a neuron. Each arrow linking an input x_i to the neuron has attached a parameter or Weight θ_i , the diagram represents the computation of a Sigmoid fed (logistic regression hypothesis) with the dot product of x and θ as in Equation (2.7), which is rewritten as Equation (2.17) for convenience.

$$h_{\theta}(x) = \frac{1}{1+e^{-\theta^T x}} \quad (2.17)$$

Where x and θ are parameter vectors $X = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_m \end{bmatrix}$ and $\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_m \end{bmatrix}$

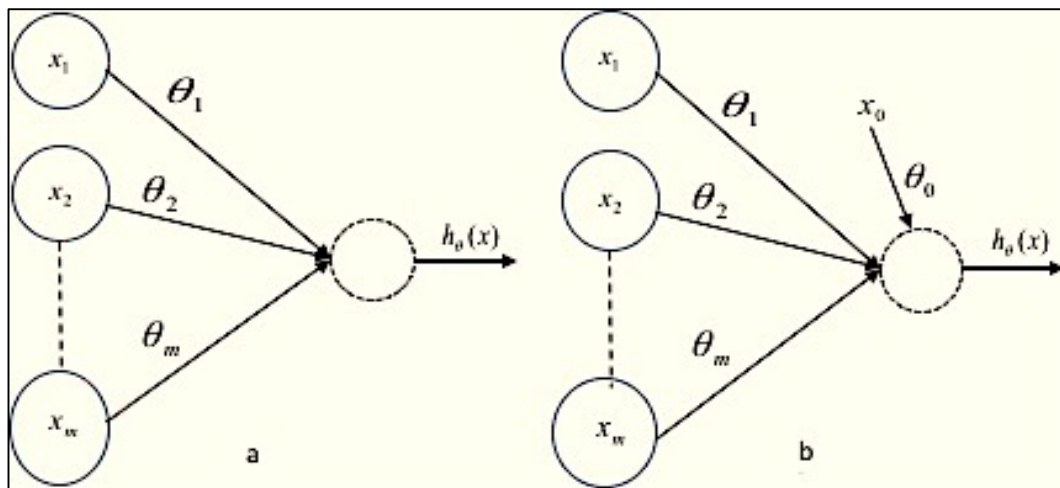


Figure 2.8: Simple Model of Neuron: (a) Without Bias, (b) With Bias Input x_0 and Weight θ_0 Explicitly Shown

Mathematically, Equation (2.18) represented an ANN implementation of a simple model of a neuron in Figure 2.8 as:

$$U_k = \sum_{j=1}^m \theta_{kj} x_j \quad (2.18)$$

Which refers to the weighted sum of input signals without bias. V_k is the weighted sum of input signals plus bias as expressed in Equation (2.19) and Equation (2.21), and y_k is the output function from the Logistic Regression Hypothesis.

$$V_k = (U_k + b_k) \quad (2.19)$$

$$y_k = h(U_k + b_k) \quad (2.20)$$

$$V_k = \sum_{j=0}^m \theta_{kj} x_j \quad (2.21) \quad \text{where}$$

$$b_k = \theta_0 x_0 \quad (2.22)$$

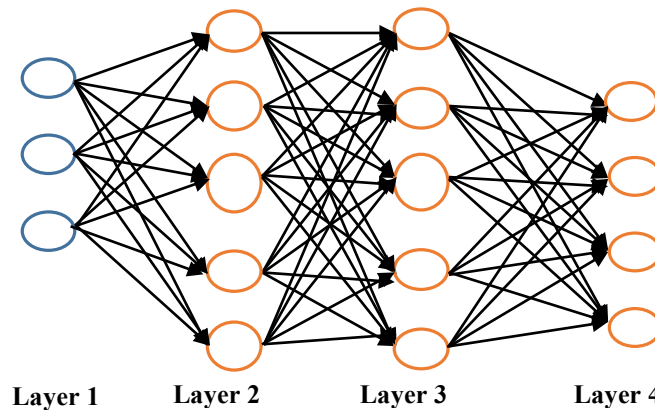


Figure 2.9: Artificial Neural Network Architecture with Four Layers

Figure 2.9 illustrates a structure of neural networks with four layers – Inputs, two middle/hidden and output layers. To express the cost function of a neural network, let explain the following notations:

L – total number of layers in networks

S_l – number of units in layer l (this exclude bias unit).

S_L - is the number of units in the last layer (the output layer).

K – the number of desired classes or approximate value (same as S_L).

Equation (2.21) expressed the cost function for ANN learning algorithm as:

$$\min_{\theta} J(\theta) = -\frac{1}{m} \left[\sum_{i=1}^m \sum_{k=1}^K y_k^{(i)} \log(h_{\theta}(x^{(i)})_k) + (1 - y_k^{(i)}) \log(1 - h_{\theta}(x^{(i)})_k) \right] + \frac{\lambda}{2m} \sum_{l=1}^{L-1} \sum_{i=1}^{S_l} \sum_{j=1}^{S_{l+1}} (\theta_{ji}^{(l)})^2 \quad (2.23)$$

ANN cost function is a modification of Logistic Regression but much more complex. Because there is only one output variable (from the model) in the case of Logistic Regression and one dependent variable y . However, in Neural Networks there may be many output variables, the hypothesis is a vector with a dimension of K and training set dependent variable is a vector.

2.5 Learning Optimization and Performance Evaluation

Optimization algorithms are important part of machine learning training process. They are critical in ensuring that learning process selects parameters that minimized the cost function during training. This section discusses two machine learning optimization algorithms and enumerates other learning optimization use in the machine learning training process. The two learning optimization algorithms are both convex and first order methods and require less computation per step when compared to those that uses second order information to train data. Also, the performance evaluation of the models will also be discussed in this section.

2.5.1 Gradient Descent Algorithm

Gradient Descent is the algorithm used to minimize different functions. In this study, it is used to minimize the cost function for both Regression and Artificial Neural Network Models. The general idea is to start by picking a random combination of the parameters $(\theta_0, \dots, \theta_n)$ and

see what the cost function is, then search for the very next combination that will minimize the cost function until it reaches the local minimal.

Table 2.5 presents the Gradient Descent Algorithm, where α is the learning rate, and it controls the step taking by the algorithm when looking for a local minimum.

Table 2.5: Gradient Descent Algorithm

Repeat until convergence {
$\theta_j := \theta_j - \alpha \frac{\delta}{\delta \theta_j} J(\theta_0, \theta_1), \text{ for } (j=0 \text{ and } j=1)$
}

Moreover, both θ_0 and θ_1 must be updated simultaneously. Despite that the gradient-based learning algorithm has been effective in the literature most especially with the right choice of alpha, the use of Jacobian for operation (first order derivative) has been criticized in the literature for inefficient in using the information content of the training data.

Optimization techniques such as quasi-Newton, Levenberg-Marquardt, Conjugate gradient, Broyden-Fletcher-Goldfarb-Shannon (BFGS) and Limited-Memory BFGS (L-BFGS) algorithms can be used to minimize a cost function. They utilize second-order information of the training data, require no learning rate alpha and often faster than the gradient descent. However, they are more sophisticated to implement and in the case of quasi-Newton, BFGS etc., the algorithms depend on the use of approximate hessian matrix which requires large memory space especially when dealing with some large-scale problems.

2.5.2 Normal Equations

Normal Equation minimized the cost function for theta (θ) by solving the Equation (2.24).

$$\frac{\delta}{\delta \theta_j} J(\theta_j) = 0 \quad (2.24)$$

When the derivative of J is equal to zero, it minimized the cost function. Thus, Normal Equation Model unlike Gradient Descent works for only a linear model and it requires that Matrix X be invertible. However, it does not require a learning rate or a feature scaling and solve in ‘one shot’. Equation (2.25) expressed Normal Equation as:

$$\theta = (X^T * X)^{-1} * X^T * y \quad (2.25)$$

To work through the derivate of the Normal Equation, the linear regression Cost Function in Equation (2.4) is re-written in a vectorized format as:

$$J(\theta) = \frac{1}{2m} (X\theta - y)^T (X\theta - y) \quad (2.26)$$

Since the interest here is to find the derivative of J and equate to zero as in Equation (2.24), dropping $1/2m$ will not affect the derivation. So, Equation (2.26) can further be simplified by expansion and rearrangement as:

$$\begin{aligned} J(\theta) &= (X\theta)^T X\theta - (X\theta)^T y - y^T (X\theta) + y^T y \\ &= \theta^T X^T X\theta - 2(X\theta)^T y + y^T y \end{aligned} \quad (2.27)$$

With Equation (2.27) the derivative of J with respect to theta (θ) equated to zero is expressed in Equation (2.28):

$$\frac{\delta}{\delta \theta} J(\theta) = 2X^T X\theta - 2X^T y = 0 \quad (2.28)$$

This implies that:

$$X^T X \theta = X^T y \quad (2.29)$$

Assuming that Matrix X is invertible, multiplying both sides of Equation (2.29) with $(X^T X)^{-1}$ will give the Normal Equation expressed in Equation (2.25).

2.5.3 Performance Evaluation

Performance Evaluation is a process of finding out that things are working fine. It is an integral part of machine learning. It includes various methods used to test each aspect of learning process starting from preprocessing stages to implementation and testing stages.

This thesis used performance evaluation methods in two aspects, one in finding some suitable parameters in both feature selection methods and machine learning implementation to avoid bias/variance problems. Secondly, performance evaluation is used to test the effectiveness of the proposed methods based on global assessment metrics.

2.5.3.1 Cross-Validation Method

Cross-Validation method is an important technique used to avoid bias or variance problems, also known as type III error (Mosteller, 1948). It is a powerful method to properly choosing the model parameter that tends to influence bias/variance problems (Seni & Elder, 2010).

To apply cross-validation technique, divide each dataset into three - training set, validation set, and testing set. This study used cross-validation technique in both Feature Selection (FS) methods and implementation of Machine Learning. For example, in one of the proposed FS method, k -fold cross-validation, random permutation operation and student t-test were employed to choose the smoothing parameter K , (Sulaiman & Labadin, 2016). The K has influence in bias/variance problems- Small value of K leads the FS estimator to have small

bias and high variance, while a large value of K , makes the estimator have low variance and high bias.

Also in a machine learning model implementation, cross-validation method is used to avoid underfitting or overfitting problems. For example, in choosing regularization parameter (λ) to prevent the overfitting problem. A range of λ values are tested during training; say ten models of the same machine learning algorithm, and then used cross-validation set to compute cost (J) for each model. Model with lowest on cross-validation set is then run with a test set for reporting generalization error.

2.5.3.2 Performance Evaluation Metrics

As previously discussed in chapter 1, feature selection helps in improving the predicting capabilities of machine learning models since it reduces the dimension of a dataset by selecting variable that are relevant to the predicting attributes. It builds a concise model that avoid overfitting and generalized better, improves an accuracy of prediction due to a reduction in estimation errors and reduces burdens on data collection & computational complexities. So, the proper way to test the effectiveness of any feature selection is to see the relative performance of a choosing learning model with selected feature subsets compare with the full feature subset.

Two global metrics widely used in the model evaluation are Mean Squared Error (MSE) and the Mean Absolute Error (MAE). Although there is much argument in the literature on whether MAE should be the actual measure of average error or the Root Mean Squared Error (RMSE), see Willmot and Matsuura (2005) and Chai and Draxler (2014) for detail discussion. In this study, two performance evaluation used more often are RMSE and Normalized Mean

Squared Error (NMSE). The reason for their choice is sensitivity to big errors. Equation (2.30), and Equation (2.31) are expression of MAE and RMSE respectively:

$$MAE = \frac{1}{n} \sum_{i=1}^n |e_i| \quad (2.30)$$

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n e_i^2} \quad (2.31)$$

Where e_i is the Model Error for example i , that is, $e_i = \text{Actual value} - (\hat{y}_i) = \text{Predicted value} - (\hat{y}_i)$. The value of both i and n relies on the test set.

2.6 Attribute Discretization

Discretization is the process of re-encoding each continuous attribute into a discrete attribute using a set of intervals. It is an essential task of data preprocessing because it transformed data into a set of intervals. Discretization is a requirement of many machines learning classification models, which require every continuous input feature converted into Nominal data and (Dougherty et al., 1995) emphasized that variable discretization, increases the speed of learning methods. In some domain, discretizing data into a set of intervals make it more relevant for human interpretation and such discretization may be based on knowledge of the domain expert.

This section is intended to be a brief review of various discretization techniques in machine learning. There are three different categorizations of discretization techniques in the literature and each technique fall into two or three of the categories:

- Supervised versus Unsupervised
- Static versus Dynamic

- Global versus local.

Binning is the simplest discretization method which divides the target attribute into k bin with equal interval width. Where 'k' is user defined (Dougherty et al., 1995). It is static, global and unsupervised discretization method (Catlett, 1992; Fayyad & Irani, 1993; Dougherty et al., 1995; Chmielewski & Grzymala-Busse, 1996). It is static in the sense that it involves a single discretization on the dataset to determine the value of k , and it is done on each feature independent of others and produces cluster over n-dimensional continuous instance space. It is unsupervised because it does not depend on the class level if any. Also, referred to equal interval width. (Dougherty et al., 1995) Highlighted that this discretization method is prone to outliers and may skew the range. Also, binning of data for learning model is performed without careful on how discretization affects the performance of the learning method (Weiss & Kulikowski, 1991).

Equal Frequency Interval is a related method in which each of the K bin has $\frac{k}{m}$ width where m is the number of instances. As unsupervised methods, this method does not utilize instance classes in partition boundaries. So, it may likely lead to loss of classification information by putting values with a strong affinity with different classes into the same bin (Kerber, 1992). Moreover, if viewing discretization method as a form of discretizing a continuous output attribute into classes or partitions, this method will produce a large number of classes that may be inappropriate for practical application.

1R Algorithm discretizes continuous values of each feature into bins, with each bin contains the majority of one particular class. This method is a constraint with the condition that each bin containing at least some pre-specified number of instance. Dougherty et al. (1995)

Highlight that this method reasonably works well when used together with the 1R induction algorithm.

Entropy-based Discretization has been explored by Catlett (1992) as a way of achieving a remarkable speed of induction on a large dataset with large continuous attributes. It is a static, local and supervised method that creates partitions for each attribute recursively. Conditions such as a minimum of samples in one partition, a maximum number of partitions and a minimum information gain serve as stopping criteria.

A Recursive Minimum Entropy Partitioning Method was proposed by Fayyad and Irani (1993). It is heuristic based discretization with the following two main ideas- use of entropy to split features into intervals that maximize the information and partitioning should be less-fine-grained to avoid over-fitting. To prevent the formation of small intervals over continuous space the method uses Minimum Description Length (Rissanen, 1997) as control criteria. Despite that this approach was implemented locally by Fayyad and Irani (1993). Ting (1994) highlighted that the method is quite promising as a global discretization method.

In an attempt to compare Binning to Entropy-based Methods, Dougherty et al. (1995) Performed a test with Naïve-Bayes classifier, the result indicated that feature discretization with the Entropy-based Method gives a significant improvement in the naïve-Bayes model.

In another development, Pfahringer (1995) proposed MDL-DISC algorithm, which is a global discretization method. Also based on Minimum Description Length principle, it uses entropy as a selection criterion to scale up the number of candidate split-points. MDL is a modern method for the inductive inference that provides a generic solution to the model selection problem, Grunwald (2004). Basically, MDL principle says that use a fewer symbol to describe data than the number of symbols needed to describe the data.

Chiu et al. (1990) have proposed a Hierarchical Discretization Method based on optimizing the Shannon entropy over the discretized space. The initial partition of the continuous space involved searching in k bins along each axis based on hill-climbing method. Moreover, the search method is repeated to a particular interval to obtain final intervals. Dougherty et al. (1995) noted that the method though applied primarily to an information synthesis task, it bears strong similarities to discretization method for machine learning research.

The Chimerge System is a supervised method based on the heuristic method and statistically justified method. The system uses chi-square test to know when to merge adjacent intervals. Conceptually, the method tests the hypothesis that two adjacent intervals are statistically independent.

Richeldi and Rossotto (1995) also proposed StatDisc Method for interval merging similar to ChiMerge. It is a bottom-up method that creates a Hierarchy of Discretization Intervals using a threshold criterion for interval merging. Unlike ChiMerge System that considered two adjacent intervals, StatDisc considers N adjacent intervals for merging, where N is user defined. Merging continue until a set threshold reached before the procedure stop.

In this thesis, simple Binning Discretization Methods was used in different studies after careful visualization of the spread of the values and in combination with the expert domain knowledge found in the literature. In particular, Information Theoretic-based Feature Selection (IFS) method utilizes discretization as a means of petitioning a continuous output value, rather than transforming the values into classes or labels. That is, the proposed IFS always use the actual output from a dataset for both regression and classification tasks. details of the partitioning of output attribute can be found in Section 5.2.

CHAPTER 3

LITERATURE REVIEW

A common observation indicated that many application areas that used machine learning prediction generate a significant number of data attributes or variables. Also, that these datasets are meant for other purposes such as stock keeping and traditional empirical analysis in those application areas, rather than machine learning prediction. “The enormity of some datasets used in recent practical applications of machine learning prompts an investigation of whether such huge training sets are necessary and how they might be reduced without sacrificing accuracy” (Catlett, 1991). Thus, feature selection is an optimization problem of choosing a small subset of features, ideally necessary and sufficient to describe the target concept. It aims at reducing the dimension of a dataset by selecting variables that are relevant to the predicting attribute(s) while eliminating the irrelevant ones.

3.1 Feature Selection Versus Feature Extraction

Feature Extraction takes place in the initial preprocessing stage of Machine Learning Modeling tasks (as shown in Figure 2.1). When the original input dataset is raw data and not in a proper shape to be understood by machine learning algorithms, such data first required the use of feature extraction. Feature Extraction is a problem of finding an alternative representation of the underlying data. Feature extraction methods may involve the use of transformation such as Scaling, Rotation, Distortion, Translation, and so on. Feature Extraction is commonly an initial preprocessing step in applications such as image processing, text mining/analytics, and document understanding (Nixon & Aguado, 2002; Sumathy & Chidambaram, 2013).

Feature Selection takes place as a final preprocessing step for a machine learning task (as shown in Figure 2.1). The idea is to reduce the original dimension of the input features further, by eliminating features that are irrelevant or redundant to the predicting output towards learning model construction. The importance of feature selection is to speed up learning and improve model quality.

3.2 Feature Selection for Big Data

“In the era of big data, many novel applications, such as social media services, high-resolution images, genomic data analysis, and document data analysis, utilize data of extremely high dimensionality, in order of millions. For example, the Web Spam Corpus 2011 (Wang et al., 2012) collected approximately sixteen million features (attributes) for web spam page detection, and the data set from KDD CUP 2010 use for educational data mining to predict student performance accurately includes more than twenty-nine million features. To handle millions of features, the scalability of feature selection methods becomes critical” (Yu et al., 2014, p. 660, para. 1).

Big data is characterized with four Vs namely, Volume, Velocity, Variety, and Veracity (as shown in Figure 3.1). Volume referred to the enormous amount of data generated in terabytes or petabytes in size, at a very high speed of processing (Velocity) per seconds or microseconds. Coming with large feature (Variety) variables from various sources which are highly uncertain data (Veracity).

The batch feature selection techniques assumed the present of all the training examples a priori. While the current online feature selection techniques assumed that feature arrives dynamically (Perkins & Theiler, 2003; Glocer et al., 2005; Carvalho & Cohen, 2006; Wang et

al., 2013; Wu et al., 2014). In both cases, there is a need for scalability of the current Feature Selection Techniques to meet up with the challenges of the Big Data Analytics.

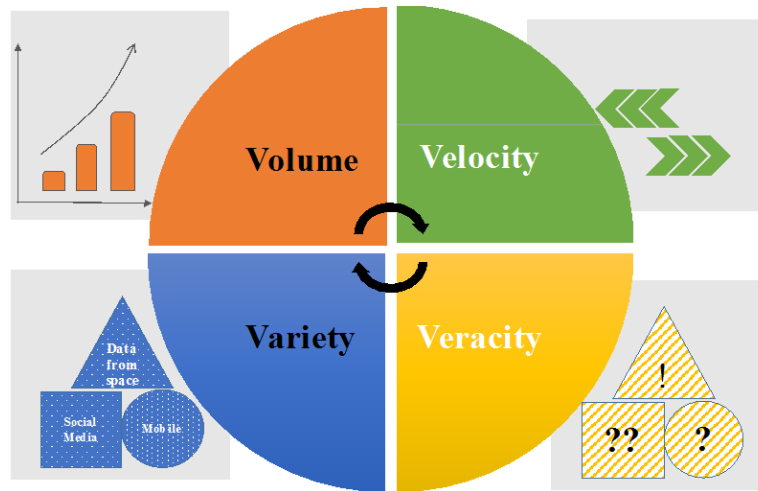


Figure 3.1: Big Data Diagram²

3.3 Characteristics of Feature Selection Algorithms

Feature Selection Algorithm is greedy heuristic optimization algorithm, like any other greedy heuristic techniques (NIST, 1999), it solves a feature selection problem by making the locally optimal choice stepwise, while the overall aim is to arrive at the global optimum (Black, 2005). Feature selection algorithm consists of three main components namely, Search Strategy, Evaluation Strategy, and Stopping Strategy.

3.3.1 Search Strategy

The Search Strategy is a systematic enumeration of candidate feature subset space. It includes the selection of the starting point from which the algorithm commences and the direction of the search. The first option is, to begin with, no features and subsequently adding feature one at a time. Also, refers to searching forward through the feature space. The second option is, to begin with, all features and consequently removing one feature at a time. This strategy is the

² This diagram is drawn by the Author of this thesis based on the initial idea from InfoDiagram.
<https://infodiagram.com/s/big%20data.html>

backward search through the feature space. The alternative strategy is the concept borrowed from dynamic programming. In this case, the search begins from no feature and subsequently add one feature at a time. Also, occasionally remove features. This strategy is forward-backward feature search. However, unlike dynamic programming, an exhaustive search through all subspaces is forbidding. However, it is possible to search for small initial feature subset from feature space.

3.3.2 Evaluation Strategy

The Evaluation Strategy is the most critical core component of Feature Selection Algorithm. One evaluation strategy utilizes statistical and a probability distribution properties of a feature to measure the relevant of the feature in features space. This assessment strategy is referring to Filter based. So, evaluation is made independent of any learning algorithm. The second evaluation strategy utilizes the performance of a feature subset space with a learning algorithm to evaluate a feature subset. This strategy uses *n-folds* cross-validation method with a learning algorithm to evaluate features, and it is known as a Wrapper.

Both search and evaluation strategies decide the final optimal feature subset from feature space in a heuristic way. Figure 3.2 and Figure 3.3 present a filter and the wrapper models, respectively.

3.3.3 Stopping Strategy

Stopping Criterion is the third component of the Feature Selection Algorithm. It is the component of the algorithm that supposed to decides when an optimal subset is reached and halt the algorithm. In one strategy, pending on the evaluation strategy, a feature is continuously added to or remove from feature subset based on the search and assessment strategies and come to halt only when further addition or removal of a new candidate feature

does not lead to a further evaluation gain in a current feature subset. In another strategy, a feature is continuously added to or remove from feature subset until reaching the other end of search space and then select the best (Blum & Langley, 1997). Alternatively, search and evaluation stopped when evaluation value begins to decrease (Verleysen et al., 2009). A question was then asked whether the reduction in evaluation value is only due to bias or noise in an evaluator, or has a sound link to the no or low relevance of a feature? (Verleysen et al., 2009). In other words, one is then forced to ask a question whether the traditional stopping strategy is scientifically sound? In a recent time, stopping strategy based on statistical hypothesis is introduced. A threshold value is set (a priori or dynamically) based on an empirical rule or statistical inference. Search and Evaluation stopped when a test value is less than or equal to a threshold value.

3.4 Heuristic and Metaheuristic Search

A dataset with M features initially available will have $2^M - 1$ non-empty feature subsets space. It is practically not feasible to test all the $2^M - 1$ non-empty feature subsets to select the best feature subset, particularly when M is large. Thus, is imperative to apply a greedy heuristic search algorithm to reduce the search space. Greedy hill-climbing strategy and its variant are common and simple heuristic local search applied to feature selection problem (Hall, 2000). Table 3.1 and Table 3.2 present Greedy Hill-Climbing (Kohavi & John, 1997) and Stochastic First-Ascent Hill-Climbing (Juels & Wattenberg, 1994) respectively.

Greedy Hill-Climbing has three alternative states to start with, whichever is the starting state the strategy considers local changes to the current subsets. If the focus of the algorithm is an addition to the feature subset, it is referring to *Forward Selection*. However, if it is deletion only, it is known as *Backward Elimination*. While when it involves both addition and deletion

to the current feature subset, is known as *Forward-Backward Elimination*. See Figure 3.2 for Filter Model Feature Selector.

Table 3.1: Hill-Climbing Algorithm

Simplest heuristic local search
• Start with a given state • Empty state • Best Tree • A random state • At each iteration • Evaluate all possible changes • Apply change that leads to best improvement in score • Reiterate • Stop when no modification improves score

One of the drawbacks of heuristic search is that it does fall into local maximum or local minimum. To avoid this limitation, TABU-search (De Werra & Hertz, 1989) or Random Restarts are introduced (Meehan, n.d.). TABU-search involves keeping K - most recently selected subsets, apply the best move that avoids the subsets in the K lists. In the case of Random restarts, when stuck in local maximum or minimum, apply some changes to some fixed number of random edges between nodes or states and restart search.

The variant approach is the Stochastic First-Ascent Hill-Climbing (Juels & Wattenberg, 1994). In this case, the search algorithm evaluates possible changes to the current feature subset at random and select the best or apply the first one that brings gain. Moreover, stop no gain after a fixed amount of unsuccessful attempts.

Table 3.2: Stochastic First-Ascent Hill-Climbing

Stochastic First-Ascent Hill-Climbing
• Evaluate possible changes at random • Apply the first one that leads "uphill." • Stop when a fixed amount of "unsuccessful" attempts to change the current candidate

Other heuristic search algorithm use for the feature selection problem are population or evolutionary based approach for solving the combinatorial optimization problem. It includes Genetic Algorithm (GA) (Ulner et al., 2011), Ant Colony Optimization (ACO) (Fidanova, 2007), and so on. A basic approach to ACO is an iterative process in which a population of a simple agent repeatedly construct candidate solutions. The construction of candidate solution is based on probabilistic process, guided by heuristic information and pheromone trail. Pheromone trail encodes a long-term memory about the entire ant search process. Moreover, Ants themselves are responsible for updating the trail. So, heuristic information or value represents a priori information about the problem example or run-time information provided by a source different from ants. Detail study of Ant Algorithm and its application to feature selection is in Chapter 7. However, Table 3.3 presents general ACO meta-heuristic pseudo code.

Table 3.3: ACO Meta-Heuristic Pseudocode

Procedure ACO Meta-heuristic	
Schedule Activities	
ConstructAntsSolutions	
UpdatePheromone	
DaemonAction	% optional
end-Schedule Activities	
end-procedure	

An ACO Algorithm consists of three procedures namely ConstructAntsSolutions, UpdatePheromone and DaemonAction processes. ConstructAntsSolution takes care of ants

that visit adjacent states. As they move, ants apply a stochastic local decision policy based on pheromone trail and heuristic information. Through this means, ants incrementally build solutions to the optimization problem. Pheromone trails are constantly modified using UpdatePheromone process, a deposit of pheromone on the node or connection increase the trails value. While pheromone evaporation decreases the trails value, Accumulation of pheromone deposit on nodes increase the probability that many ants either used those nodes or that were used by at least one ant and will lead to the good solution when it is used again by future ants. Pheromone evaporation provides a means of avoiding the path that will result in suboptimal solution and favoring the exploration of a new area of the search space. DaemonAction process introduces centralization of actions which cannot be performed by individual ant due to the lack of global knowledge.

3.5 Feature Filters

Filter Feature Selection Methods are the most widely used techniques for feature selection in Machine learning. Very many of which focus on classification tasks. Filters operate independent of the specific learning algorithm, this free mode of operation made a filter fast and relatively generalized better with all learning algorithms. Huan and Lei (2005), categorized filters into four categories namely; Distance Measures, Information Measures, Dependency Measures and Consistency Measures. The basis of this categorization is the Selection Criterion Function used to evaluate how relevant is a feature or feature subset. Figure 3.2 is the filter model feature selector, it takes in a training data with full features and apply heuristic search procedure guided by the evaluation criterion which quantitatively and qualitatively determine the goodness of a selected feature subset. The heuristic search often defined the quantity (in terms of number) of feature to be selected from a feature space and evaluation criterion determined the quality of each feature to be selected from a feature space.

Finally, an optimal feature is selected, and to test the optimality of the selected feature subset for machine learning prediction task, the selected optimal feature subset is passed to the machine learning algorithm for training and performance evaluation.

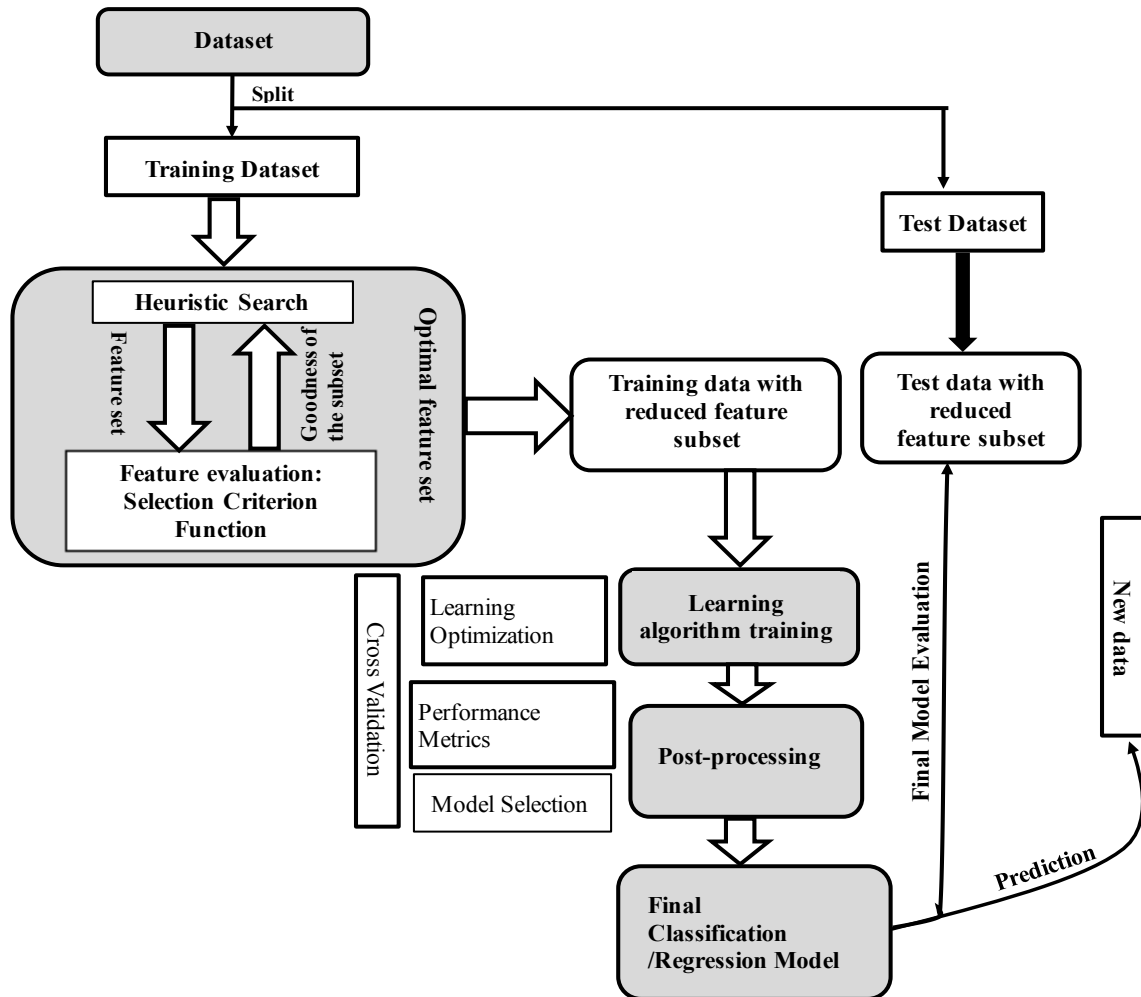


Figure 3.2: Filter Model Feature Selector

3.5.1 Distance Measures Filter-based Feature Selection

This category of Feature Selection Filters uses distance measure to discriminate between feature instances. For example, in a two-class problem where w_1 and w_2 represents the two classes and X and Y are feature instances. If $P(\omega_1/X = x)$ is separable from $P(\omega_2/X = x)$ as far as possible and $P(\omega_1/Y = y)$ is inseparable or poorly separable from $P(\omega_2/Y = y)$, then feature X is preferred to feature Y .

An example of Distance measure feature selection is RELIEF by Kira and Rendell (1992) and its variants RELIEF-F by Kononenko (1994) and RELIEF-S by Liu et al. (2004). Relief Algorithm randomly selects X instance from S - training dataset and use Euclidean distance to select its Near-hit and Near-miss instances. Near-hit of X is any instance that belongs to the close neighborhood of X and belong to the same category or class as X . While the Near-miss to X is any instance that does not belong to the same category or class as X , but belong to the close neighborhood of X . So, once the Near-hit and Near-miss of X instances are selected, a weight updated method as defined in Equation 3.1 is executed.

Subsequently, X feature is relevant if its relevancy measure is greater than or equal to the threshold value τ . Equation 3.2 computes relevancy measure in Relief.

$$W_i = W_i - \text{diff}(x_i, \text{near} - \text{hit}_i)^2 + \text{diff}(x_i, \text{near} - \text{miss}_i)^2 \quad (3.1)$$

where diff is a function that provides the differences between two instances of feature values. That is:

$\text{diff}(x_k, y_k) = \begin{cases} 0 & \text{if } x_k \text{ and } y_k \text{ are the same} \\ 1 & \text{if } x_k \text{ and } y_k \text{ are different} \end{cases}$ for the nominal feature. However, when x_k and y_k are numeric $\text{diff}(x_k, y_k) = (x_k - y_k)/nu_k$, where nu_k is the normalizing factor that makes the values of diff into $[0, 1]$ interval.

$$\text{Relevance} = \left(\frac{1}{m}\right) W \quad (3.2)$$

3.5.2 Information Measures Filter-based Feature Selection

The concept of Information measures as related to feature selection rule refers to the concept of information gain. This is conceptualized in Liu et al. (2000) based on classification task as follows:

Given an uncertainty function μ and the apriori class probability $P(C_i)$ for $i = 1, 2, \dots, d$. The information gain from a feature X , $IG(X)$ defined as the difference between the prior uncertainty $\sum_i \mu(P(C_i))$ and the expected posterior uncertainty using X , i.e.,

$$IG(X) = \sum_i \mu(P(C_i)) - E[\sum_i \mu(P(C_i|X))] \quad (3.3)$$

It means that feature X is chosen over feature Y if $IG(X) > IG(Y)$. Feature with less uncertainty has higher IG value than the feature with a much higher degree of uncertainty over the same feature space. For filter feature selection based on information measures, see the report in Huan and Lei (2005). For example, Koller and Sahami (1995) provides a suboptimal feature selection based on information gain known as Approximate Markov Blanket (AMB). AMB is capable of eliminating unnecessary feature and redundancy in features. Also, it can resist inconsistencies in training data. It relates to a backward elimination heuristic method where G is instantiated to contain all the features set. So, for each feature $F_i \in G$, Markov blanket M_i is selected as a criterion to remove the unnecessary feature. Which built on the idea that Markov blanket subset M subsume all information that F_i has about a class C as well as all other features in the same feature space. In this regard, expected cross-entropy is computed for F_i given M_i , in each iteration and it is used to eliminated a certain number of features pre-specified. Expected cross-entropy is computed as:

$$\delta_G(F_i|M_i) = \sum_{f_{M_i}, f_i} \Pr(M_i = f_{M_i}, F_i = f_i) \cdot D(\Pr(C|M = f_M F_i = f_i), \Pr(C|M = f_M)).$$

M_i is set to contain K features F_j in $G - \{F_i\}$ based on correlation factor (ρ_{ij}) with the largest magnitude. Correlation factor is computed at the beginning for every pair of F_i and F_j as:

$$\rho_{ij} = \frac{Cov(F_i, F_j)}{StdDev(F_i)StdDev(F_j)} \quad (3.4)$$

So, each iteration contains the steps above, and i with a minimal value of $\delta_G(F_i|M_i)$ is eliminated in each iteration.

3.5.3 Consistency measures Filter-based Feature Selection

The consistency measures filters rely on a selection criterion that utilizes the discriminating power of the data examples to describe its classes, rather than relying on class information as a basis for feature selection. Feature selection via set cover by Dash (1997) is one example of consistency measures filter. Dash (1997) formalized feature selection problem as that of finding the smallest set of feature that is "**consistent**" in describing classes. It equates to 'covering' each pair of examples that have different class labels (Oliveira & Vinceritell, 1992; Dash, 1997). Two instances of two features are inconsistent if they match all but differ in their class label. For example, (0 1 1) and (0 1 0) are two instances with matches in their first two attributes but differ in the last attribute. So, Dash (1997) consistency selection function is derived from inconsistency definition. That is, a feature S is consistent if, for any pair of examples that are inconsistent, there exist another feature in S for which their value differs (Oliveira & Vinceritell, 1992). So, including feature f in S will "cover for" all those examples pairs that differ in class labels. It implies that S is consistent once all pairs are fully "taken care of". Thus, Dash (1997) applied Johnson's greedy approximation algorithm for Set cover to Feature Selection Problem.

3.5.4 Dependency Measures Filter-based Feature Selection

Dependency measures filters are correlation measure feature selection. Features with higher dependency measure to the predicting attribute are considered to be more relevant. Correlation-based feature selection (CFS) by Hall (2000) is an example of dependency measures filter. CFS has selection criterion function that is derived from Pearson's correlation as follows:

$$Merit_s = \frac{k\overline{r_{cf}}}{\sqrt{k+k(k-1)\overline{r_{ff}}}} \quad (3.5)$$

The $Merit_s$ is a heuristic merit for feature subset S with k features, the $k\overline{r_{cf}}$ is feature class correlation average, and $\overline{r_{ff}}$ is the feature to feature inter-correlation average.

CFS uses best first search heuristic as feature selection algorithm. The algorithm begins by constructing a matrix of the feature class and feature-feature correlations, and with an empty set of features, all possible single feature expansions are generated. Feature subset with highest $Merit_s$ evaluation is chosen, and expansions continue in the same manner. CFS come to halts based on no further improvement in the search stopping criterion. With five consecutive non-improving feature subsets as a yardstick.

3.6 Feature Wrappers

Wrappers are machine learning algorithm dependent feature selectors. Wrapper uses a specific learning algorithm to evaluate a selected feature subset in an iterative step, in each iteration, a new feature is generated and tested for performance with a particular learning algorithm. The algorithm comes to stop if no further improvements are achieved. Moreover, a feature with highest performance value is considered the selected optimal feature subset. So, wrappers use heuristic algorithms to search through feature subsets space. Moreover, in the subsequent steps, test each subset with a specific learning algorithm in an iterative way. Wrappers use cross-validation approach in the evaluation of feature subset with a learning algorithm. For this reason, wrappers are found to be more accurate with a specific learning algorithm that is used for the search. However, in doing so, it incurs much higher computational overheads when to compare with filters (Guyon & Elisseeff, 2003; Yu & Liu,

2003; Unler et al., 2011, Brown et al., 2012). Figure 3.3 presents wrapper model feature selector.

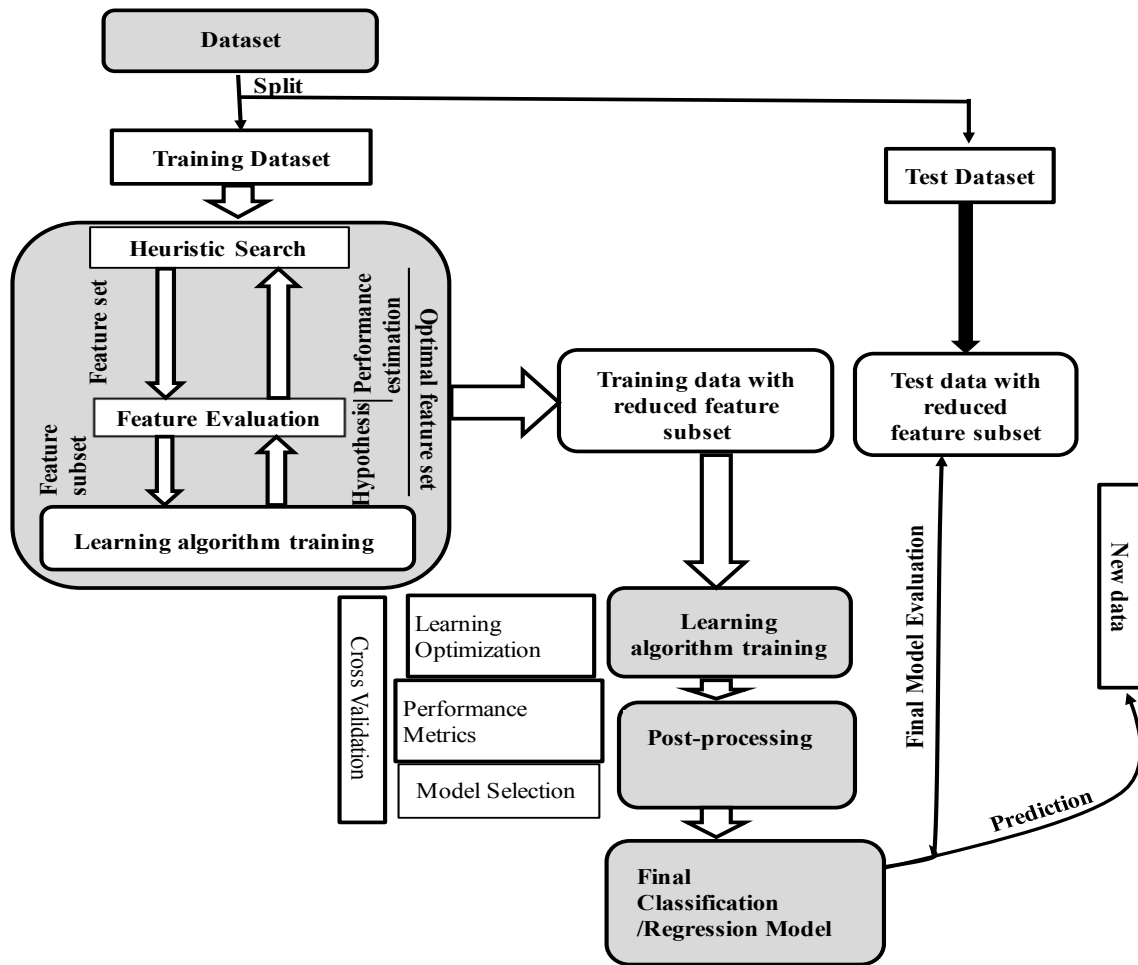


Figure 3.3: Wrapper Model Feature Selector

Despite that wrappers strengths are based on bias from a learning algorithm, this makes feature subset selected using wrapper only to be less generic. Such feature subset may perform badly with other learning algorithms different from the one used for evaluation. Thus, apart from the comparative computational efficiency of the filter, its generic selection of features independent of learning algorithm is a compelling argument to use a filter (Unler et al., 2011).

3.7 Feature Hybrids

Hybrid models are feature selection models that tend to bring the benefit of both filters and wrappers together while doing away with the disadvantage of the individual selector. In one hand filter generalized better due to the independency of any learning algorithm, very fast to process, but less accurate compared to the wrapper. On the other hand, wrappers are more precise, but takes a long time to process due to high computational overhead.

In most studies, a hybrid of filters and wrappers uses filters to rank features or to reduce the number of candidate features (Unler et al., 2011). This way, a number of feature subsets are generated by filter. Then wrappers in a cross-validation procedure find the optimal subset among the feature subset space. Unlike wrapper only that exhaustively test several feature subsets from feature space, filter-wrapper hybrid method selects optimal feature subset from few feature subset space, this way it significantly reduced computational overhead. Figure 3.4 is hybrid (filter-wrapper) model feature selector. It is mostly sequential two steps approach, starting with filter reducing the number of feature subset space by selecting a specified number of feature subsets from feature space in the first step. Moreover, the second step, a machine learning algorithm is used in a cross-validation method to trained and test each of the feature subset. The optimal feature subset is the one with the best performance with machine learning algorithm.

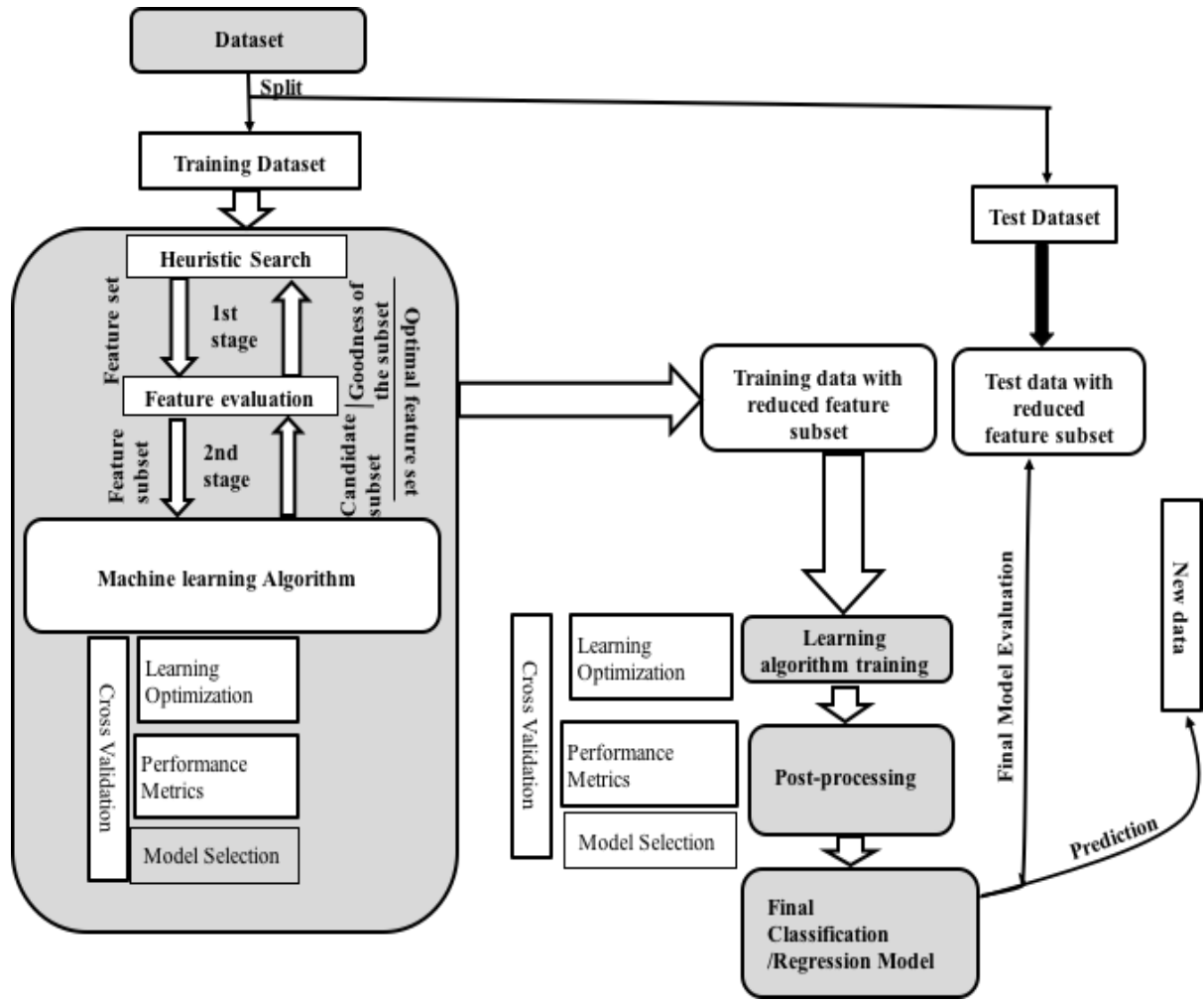


Figure 3.4: Hybrid (filter-wrapper) Model Feature Selector

In the recent time, meta-heuristic optimization algorithms such as Genetic Algorithm (GA), Particle swarm optimization (PSO) and Ant colony optimization (ACO) are employed based on a wrapper scheme in the second stage (Huang et al., 2007; Sivagaminathan & Ramakrishnan, 2007; Unler et al., 2011). So, hybrid models have been proposed to make use of both filters and wrappers, particularly for very high dimensional data (Das, 2001; Xing et al., 2001; Unler et al., 2011).

Figure 3.5 presents a hybrid model that is based on filter and metaheuristic in a filter-wrapper scheme. The idea is to assume some form of independence among feature subset selected.

One of the limitation of Wrapper based feature selection method is that its strength is biased by machine learning algorithm used as evaluation criterion and for that reason any selected feature subset based on wrapper may lacks generic optimality with all machine learning algorithms. That is, it requires that each machine learning algorithm to re-search through a feature space to find its optimal feature subset. Unfortunately, the same limitation may be present in the filter-wrapper hybrid method.

However, unlike wrapper, metaheuristic optimization algorithms such as genetic, particle swarm and ant colony optimization algorithms have been effectively used to solved a combinatorial problem in polynomial time t . Like filter, metaheuristic procedure does not require machine learning algorithm to solve a combinatorial optimization problem. Thus, when used to solve the feature selection problem, will produce a generic optimal feature subset that is independent to any machine learning algorithms. This is a compelling argument to use a hybrid of filter-metaheuristic method with a comparative computational efficiency of the filter and generic selection of features independent of learning algorithm. In the framework, filter will be used to select a feature subset in the first stage and metaheuristic algorithm is used to further reduce the feature subset while still maximizing the relevance of the final optimal feature subset.

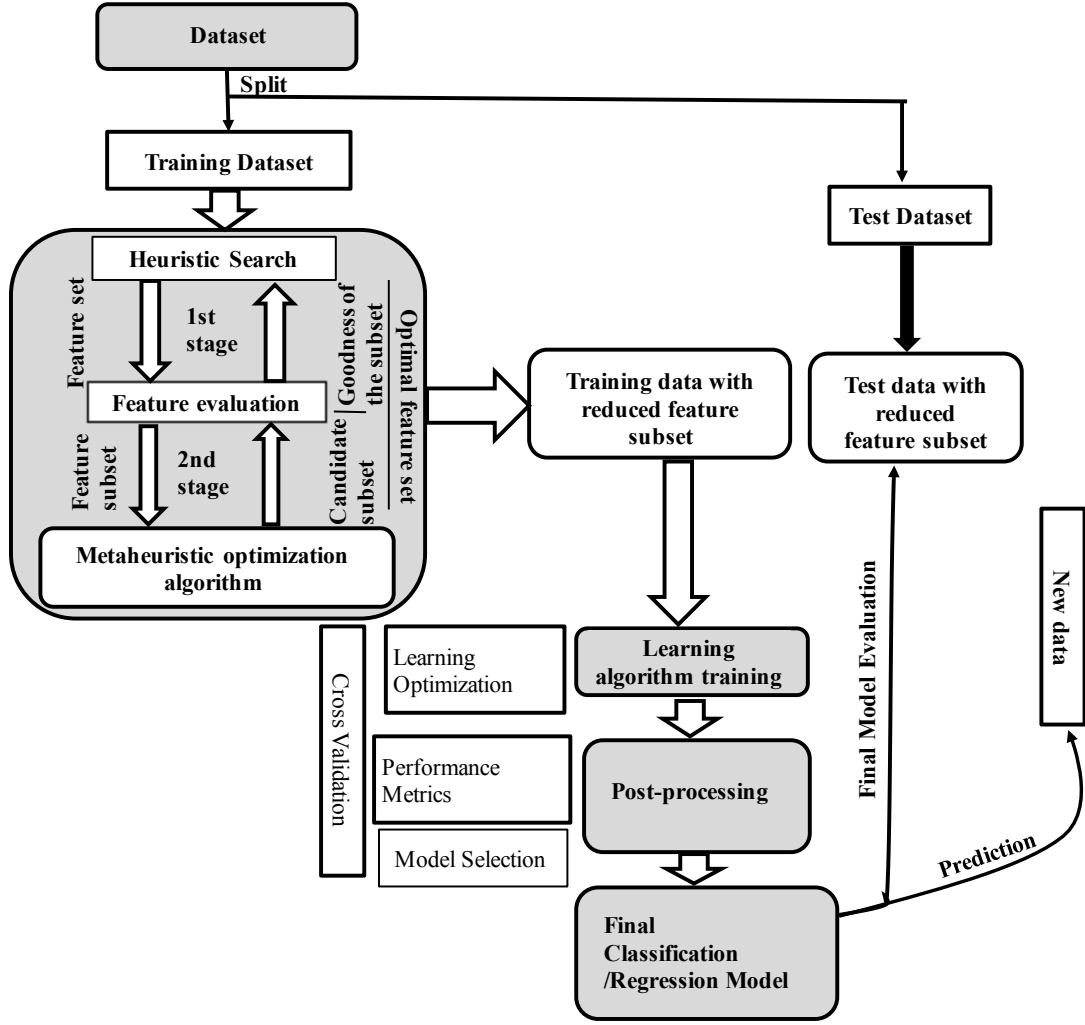


Figure 3.5: Hybrid (filter-metaheuristic) Model Feature Selector

3.8 Brief Description of some selected Filter-based Feature Selection Benchmark Selected for Evaluation of the Proposed IFS Method

Kruskal Wallis: is a filter based feature weighting method. It is a non-parametric method that does not consider a normal distribution of the data, it uses population medians among group to rank all data points across all group together by ignoring group membership. Kruskal Wallis is expressed as:

$$K = (N - 1) \frac{\sum_{i=1}^g n_i (\bar{r}_i - \bar{r})^2}{\sum_{i=1}^g \sum_{j=1}^{n_i} (r_{ij} - \bar{r})^2}. \quad (3.6)$$

Where:

- n_i is the number of observations in a group i
- r_{ij} is the rank of observation j from a group i
- N is the total number of observation across all groups
- r_i is the average rank of all observations in a group i , expressed as:

$$r_i = \frac{\sum_{j=1}^{n_i} r_{ij}}{n_i} \quad (3.7)$$

- $\bar{r}$ is the average of all the observations; that is average of all the r_{ij} , i.e.

$$\bar{r} = \frac{1}{2}(N + 1) \quad (3.8)$$

Kruskal Wallis estimate can further be simplified based on the simplification of its denominator as in Equation (3.9). see Zheng, et al. (2010) and Kruskal-Wallis. (n.d.) for detail of the simplification.

$$\hat{K} = \frac{12}{N(N+1)} \sum_{i=1}^g n_i (\bar{r}_i)^2 - 3(N + 1) \quad (3.9)$$

Information Gain: is a filter based feature weighing method. It is one of the most popular and simple to interpret. It is fundamentally misnomer to consider Information Gain as a feature selection algorithm in its entirety, because information gain is also known as mutual information. However, as a feature evaluator together with a machine learning algorithm like decision tree algorithm or heuristic search algorithm like greedy algorithm it satisfied the criterion to be considered as feature selection method. In ASU feature selection packages, Information Gain uses Weka's feature selection algorithm.

Information Gain (IG) of a feature X and the class labels Y is computed as:

$$IG(X, Y) = H(X) - H(X|Y). \quad (3.10)$$

where $H(X)$ is the entropy of X and $H(X|Y)$ a conditional of X after observing Y . Entropy is a measure of the uncertainty associated with a random variable. Both entropy and conditional entropy of variables X and Y for information gain are defined as:

$$H(X) = -\sum_i P(x_i) \log_2(P(x_i)). \quad (3.11)$$

$$H(X|Y) = -\sum_j P(y_j) \sum_i P(x_i|y_j) \log_2(P(x_i|y_j)). \quad (3.12)$$

Theoretically, the maximum value of IG is 1. Although, in practice, this may not be the case. However, feature with high information gain is relevant and features with top k values are selected as relevant features.

Fast Correlation Based Filter (FCBF): is filter based feature selection algorithm that evaluates feature-class and feature-feature correlation. Details of the Algorithm can be found in (Lei & Huan, 2003). However, it uses symmetrical uncertainty (SU) as goodness measure which is defined as:

$$SU(X, Y) = 2 \left[\frac{IG(X|Y)}{H(X) + H(Y)} \right] \quad (3.13)$$

Where Equation (3.10), Equation (3.11) and Equation (3.12) defined $IG(X|Y)$, $H(x)$ and $H(X|Y)$.

Given a dataset S with N features and a class C , a subset S' contains all relevant features is selected based on a threshold SU value δ , such that $\forall F_i \in S', 1 \leq i \leq N, SU_{i,c} \geq \delta$. $SU_{i,c}$ denotes the SU value that measures the correlation between a feature F_i and a class C .

FCBF hypothesized that a feature is good if it is *predominant* in predicting the class concept, and feature selection for classification identifies all predominant features to the class concept and removes the rest. That is, the correlation between a feature $F_i \in S$ and a class C is

predominant *iff* $SU_{i,c} \geq \delta$ and $\forall F_j \in S' (j \neq i)$, there exists no $F_j: SU_{j,i} \geq SU_{i,c}$. $F_j \in S_{p_i}$ is call redundant peer to F_i if exist. S_{p_i} denotes the set of all redundant peers for F_i .

Correlation Based Feature Selection (CFS): CFS is a correlation measure feature selection that derives its feature evaluator from Pearson's correlation as defined in Equation (3.5). Refers to Section 3.5.4 for brief description of CFS and for detail description see Hall (2000).

Minimum-Redundancy-Maximum-Relevance (mRMR): mRmR uses mutual information to evaluate features. See Peng, et al. (2005) for details of mRMR algorithm. However, mRMR maximize dependency between the joint distribution of the selected features and the class variable while minimizing redundancy in selected features. The combination of the two criteria Max-Relevance and Min-Redundancy (i.e. mRMR criterion) is equivalent to the Max-Dependency criterion if one feature is selected at one time.

Min-Redundancy

$$\text{For Discrete variables: } \min W_I, W_I = \frac{1}{|S|^2} \sum_{i,j \in S} I(i, j) \quad (3.14)$$

$$\text{For Continuous variables: } \min W_C, W_C = \frac{1}{|S|^2} \sum_{i,j} |C(i, j)| \quad (3.15)$$

Max-Relevance

$$\text{For Discrete variables: } \max V_I, V_I = \frac{1}{|S|} \sum_{i \in S} I(h, i) \quad (3.16)$$

$$\text{For Continuous variables: } \max V_F, V_F = \frac{1}{|S|} \sum_{i \in S} F(i, h) \quad (3.17)$$

Where S is the feature set, $I(i, j)$ the mutual information between feature i and j , $C(i, j)$ is the correlation between feature i and j , $h = \text{target classes}$ and $F(i, h)$ is the $F - \text{Statistic}$.

Max-Dependency criterion is defined as $I(S, h)$, which is a mutual information between the selected variables S and the target variable h .

Finally, all the five filter-based supervised feature selection algorithms discussed above are designed for classification tasks only.

3.9 Review of Ant Colony Optimization (ACO) and Multidimensional Knapsack Problem (MKP)

This section briefly reviews the Ant Colony Optimization System and its application to solving subset selection problems based on multidimensional knapsack problem formulation. Finally, the section review feature selection as a combinatorial problem, and established its representation as a multidimensional knapsack problem.

3.9.1 Ant Colony Optimization

Ant colony optimization (ACO) is a population-based optimization approach, inspired by the ways in which some ant species search for food from the source to their nest. Also, their inherent ability to find the shortest path from the source of their food to their nest. This foraging behavior of ant was adopted in ACO to solve hard combinatorial problems (Bonabeau et al., 1999; Dorigo & Stutzle, 2004). Ants deposit pheromone on the ground to serves as a footprint of some favorable path that should be followed by members of the colony as they move from nest to the source and back to the nest in search of the food. So, to solve optimization problems, ACO exploits a similar mechanism (Dorigo et al., 2006). ACO process is an iterative process in which some artificial ants repeatedly construct candidate solutions. The construction process is probabilistically biased towards heuristic information on the given problem instance and pheromone trails that formed a shared memory containing experience gathered by the ants in the previous iterations.

ACO has been applied to a broad range of NP-hard problems, among which are travelling salesman problem (Dorigo & Gambardella, 1997; Dorigo et al., 1999). Quadratic assignment problem (Gambardella et al., 1999a ; Maniezzo & Colomi, 1999). Vehicle routing problem (Gambardella et al., 1999b) and the multidimensional knapsack problem (Leguizamon & Michalewicz, 1999; Fidanova, 2003; Alaya et al., 2004; Fidanova, 2005; Fidanova, 2006; Fidanova, 2007; Boryczka, 2007; Kong et al., 2008; Lee & Bau, 2012).

3.9.2 Multidimensional Knapsack Problem (MKP)

The knapsack problem is a well-studied combinatorial optimization problem (Dantzig, 1957; Grotschel & Nemhauser, 2008). With wide ranges of applications involving resource allocation such as allocating processors & database in a distributed computer system, the capital budgeting, project selection, cutting stock and cargo loading, Kong et al. (2008). It is a case of having a knapsack with limited capacity (weight) W and a set of items, each with weight and value. The problem involves deciding the number of items to be included into the knapsack, in such a way that maximum total values is derived from items in the knapsack, while considering the weight of each item and the capacity W of the knapsack.

There are two types of knapsack problem namely:

- “0-1 knapsack problem” – here items are indivisible; is either taken or not taken. Some individual instances are solved with dynamic programming in pseudo-polynomial times.
- “Fractional Knapsack problem” – items are divisible; any fraction of an item can be taking.

Knapsack problems are *NP*-hard since there are no known efficient techniques other than possibly brute force (exhaustive enumeration of the solution space). However, knapsack

problem belongs to the class of *NP*-complete problems with a polynomial-time nondeterministic (*NP*) solutions.

3.9.2.1 Dynamic Programming Solution for the 0-1 Knapsack Problem

Dynamic programming is a recursive approach to solving a complex problem, by way of dividing the problem into a collection of smaller subproblems of the same type, and subsequently use the solutions of those subproblems to construct a solution of the whole problem in a pseudo-polynomial time. Dynamic programming algorithms are often used for optimization. It has been used to solve wide ranges of discrete optimization problems such as scheduling, string-editing, packaging and inventory managements, and have proven to be effective. (Potts & Kovalyov, 2000; Woeginger, 2000; Chowdhury et al., 2010; Doerr et al., 2011).

Table 3.4 presents a dynamic programming algorithm for the 0-1 knapsack problem. This algorithm has been effective in solving instances of 0-1 knapsack problem in a pseudo-polynomial time with time complexity of $\theta(n.W)$. However, dynamic programming is an exhaustive search procedure, hence completely prohibited in feature selection. Thus, thesis focus is to adopt the same methodology to design a population optimization algorithm for the 0-1 knapsack problem. In essence, this study focuses on the use of binary ant colony optimization system to solve a multidimensional knapsack feature selection problem. Various studies have shown that in many cases a problem that can be solved by dynamic programming, can also be solved by an evolutionary algorithm. In other words, a class of problems that is easy for the dynamic programming algorithm is also easy for a suitable evolutionary algorithm for most of the reasonable meanings of term "easy" (Doerr et al., 2011).

Doerr et al. (2009) proposed a framework based on dynamic programming approach for developing evolutionary algorithms with an additional ability to carrying out dynamic programming steps, and analyzed the optimization time. For example, expected optimization time for Knapsack problem using the proposed framework is $\Theta(n^2 \cdot W \cdot \log(n \cdot W))$.

Table 3.4: Dynamic Programming Algorithm for the 0-1 Knapsack Problem

Input:
1. v : an array to store Values of items
2. w : an array to store weights of items
3. n : number of distinct items
4. W : Knapsack capacity

5. j represents the instance of knapsack capacity from zero to the maximum capacity W
6.
7. <i>for</i> $j = 0$ <i>to</i> W
8.
9. $m[0][j] = 0$; $m[i][j]$ defines the maximum value that can be achieved with weight
10. j less than or equal to j .
11. <i>end</i>
12.
13. <i>for</i> $i = 1$ <i>to</i> n $// i$ refers to item i
14. (w_i, v_i) represents a pair of weight and value of an item i .
15. <i>for</i> $j = 0$ <i>to</i> W
16.
17. <i>if</i> $w_i \leq j$ $//$ item i can be part of the solution.
18.
19. $v_i + m[i-1][j-w_i] > m[i-1][j]$
20.
21. $m[i][j] = v_i + m[i-1][j-w_i]$;
22.
23. <i>else</i>
24.
25. $m[i][j] = m[i-1][j]$;
26.
27. <i>end</i>
28.
29. <i>else</i>
30.
31. $m[i][j] = m[i-1][j]$; $// w_i > j$
32.
33. <i>end</i>
34.
35. <i>end</i>
36.
37. <i>end</i>

3.9.3 Formulation of ACO for the MKP

Leguizamon and Michalevitz (1999) and Fidanova (2003, 2005, and 2007) were among those that first provides the general framework of ant colony system for knapsack problem. ACO for the MKP is most suitable for subset problems which are quite different from general ACO for ordering problems (Leguizamon & Michalevitz, 1999). Subset problem involve selecting

the best subset of s items out of a set S of n items, while possibly satisfying additional constraints. This is contrary to the concept of a path for which the general ACO is applied. Thus, it becomes very difficult to directly apply the general ACO formulation to the problem of subset selection (Leguizamon & Michalevitz, 1999, Fidanova, 2007).

Ant Colony System for Multidimensional Knapsack Problem (MKP) can be viewed as an allocation problem where there are m resources (the knapsacks) and n items and item i has a value of v_i and each resource has a budget C_j (knapsack capacity) and weight w_{ij} of resource j by item i . The core interest here is to *maximizing the sum of values*, while working with *limited budget*. Thus, MKP is formulated as:

$$\mathcal{J}_{mkp} = \max_{x_i} \sum_{i=1}^n v_i x_i \quad (3.18)$$

$$\text{subject to } \sum_{i=1}^n w_{ij} x_i \leq C_j \quad j = 1, 2, \dots, m$$

$$x_i \in \{0, 1\} \quad i = 1, 2, \dots, n.$$

That is, $x_i = 1$ if an item i is chosen and 0 otherwise.

This problem consists of m constraints; hence it is referred to as m -dimensional knapsack problem. Assuming that $J = \{1, 2, \dots, m\}$ and $I = \{1, 2, \dots, n\}$, with $C_j \geq 0 \forall j \in J$. A well declared MKP assumes that $v_i > 0$ and $w_{ij} \leq C_j \leq \sum_{i=1}^n w_{ij}$ (Leguizamon & Michalevitz, 1999; Fidanova, 2007). Note that, x_i will be fixed to zero or some constraints being eliminated if there is any violation of these conditions (Leguizamon & Michalevitz, 1999). Also, both w_{ij} and C_j ($\forall i = 1, 2, \dots, n; j = 1, 2, \dots, m$) are non-negative. Hence, distinguishes this problem from the general 0-1 linear programming problem (Leguizamon & Michalevitz, 1999).

3.9.3.1 Feature Selection as a Combinatorial Problem

Song et al. (2012) presents the problem of supervised feature selection as a combinatorial optimization problem. Prior to that, Charikar et al. (2000) provides a unified theoretic framework for studying combinatorial feature selection problems in general, which captured both dimension reduction and clustering problems.

Assuming that X is an input dataset and let F be the full set of features, such that each $f_j \in F$ corresponds to $X_j \in X$. Feature selection aimed at selecting a subset $f \subseteq F$ such that f retains the relevant information contained in X . Assuming that the relevance of a feature subset (to the outcome) is quantified by $Q(f)$, and is estimated by restricting the dataset to the dimensions in f . The objective function of feature selection problem is formulated as (Song et al., 2012):

$$\mathcal{J}_{fsp} = \underset{f \subseteq F}{\operatorname{argmax}} Q(f) \quad (3.19)$$

$$\text{subject to } |f| \leq t,$$

where $|\cdot|$ computes the cardinality of a set and t is an upper bound on the number of selected features.

It is easy to see that the objective function of the feature selection problem ($\mathcal{J}_{fsp}$) can conveniently be represented as a special case of the objective function of MKP ($\mathcal{J}_{mkp}$) as in Equation (3.18) because, both are designed to maximize the value or relevance of a subset of items or features. And since a well declared MKP assumes that $v_i > 0$ and $w_{ij} \leq C_j \leq \sum_{i=1}^n w_{ij}$ (Leguizamon & Michalevitz, 1999; Fidanova, 2007), by comparing Equation (3.18)

with Equation (3.19), the following settings are valid: $|f| = \sum_{i=1}^n w_{ij}x_i$ and $t = m.C_j$, if $\forall i \& j: w_{ij} = 1, C_j = 1$ and $x_i \in \{0,1\}$.

3.9.4 ACO Algorithm for MKP

In order to solve the problem of subset selection using MKP, in every iteration, the ants look for a subset of n items (see Equation (3.18)) such that the total profit or value is maximized and all resource constraints are satisfied. Ants search for subset of n items is guided by pheromone trails and heuristic information. In search of subset of n items, Ants transverses through nodes from nest to the source in a graph or network scheme. This process is referred to as Tour construct and it represents the solution vector. Ants construct tour biased by the pheromone trails and heuristic information. Pheromone trails and heuristic information are the two components of transition probability with which Ants decide movement during tour construct.

3.9.4.1 Pheromone Model

Deciding which component of the tour construct should be rewarded and how to exploit these rewards during the construct of a new solutions is the key point for solving ACO problem.

- I. In solving MKP with ACO algorithm, Leguizamon and Michalevitz (1999) uses model that involves laying pheromone trails on each selected node of the graph. This is to increase the visibility or desirability of some nodes so that they are more likely to be selected in constructing a new tour or solution vector. In this model, the ordering of the selected items in a subset solution is of no interest, hence the most recent components incorporated to the solution vector need not to be involved in the process for selecting the next element (Leguizamon & Michalevitz, 1999).

- II. However, Fidanova (2003) uses the alternative model of laying pheromone trails. In this model pheromone trails are laid on path, link or arcs that connect current node or component to the next node or component. In this scheme, the ordering of component is needed and the idea is to increase the visibility or desirability of chosen next node $j + 1$ while in a current node j .
- III. Alaya et al. (2004) proposed and used a third model in which pheromone trail is lay on a pair of objects or items (o_i, o_j) . The idea is to increase the visibility and desirability of chosen more components of from currently constructed solution in the next tour constructs. Like the first model, in the third model the ordering of the selected items in a subset solution is of no interest. Their intuition is to encourage selection of more of objects from the previous partial solution under construction.

3.9.4.2 Heuristic Information

Heuristic information is the second component of the transition probability. MKP been a constraints problem, its heuristic information is constructed using the constraints in a different way. The profit or value of an object or item is the most important information for objective function, and this will be included in heuristic (Fidanova, 2007). There are two models of heuristic information; static and dynamic.

- Static heuristic information is set at the beginning of the run to a fixed value.
- Dynamic heuristic information is dynamically set based on the partial solution at time t .

Other relevant ACO for MKP include Kong et al. (2008) who proposed a new method of ACO for MKP referred to as Binary Ant System (BAS). BAS differ from Leguizamon and Michalevitz (1999), Fidanova (2003) and Alaya et al. (2004) in the following ways:

BAS relaxed the MKP constraints to allow repair of infeasible solutions after their constructions while others approaches uses constraints information to guide ants finding feasible solution. In addition to that, BAS completely ignored computation of heuristic information both static and dynamic models, but rely on pheromone trails to generate transition probability. This approach has not only violated the original principle of ACO, but also ignore profit or value of items (which is the most important information in MKP objective function) in deciding Ants movement across nodes. In addition to that, BAS introduced a Pseudo-utility that relaxes MKP constraints condition by introducing a set of surrogate multipliers (weights) to constraint parameters (i.e. summation of consumption (w_{ij}) and budget (knapsack capacity) (C_j) as in Equation (3.18)), this is to allow construction of all possible solutions. Then introduces a local search based on greedy method to check if a potential solution is infeasible, the search randomly selects 4 components from a potential solution and finally introduces repair operation to fix the new solution if possible. The repair operation flip-flop bits of potential solution vector from 1 to 0 if feasibility is violated and from 0 to 1 as long as feasibility is not violated. The new repaired solution replaces the original solution if it is better. The local search is performed a 1000 times for each solution generated by ants (Kong et al., 2008). In terms of pheromone update, BAS applied evaporation rate to each node in a usual way and subsequently used convergence factor (cf) to update a set of solutions based on original approach introduced by Blum and Dorigo (2004). The set of solution include global best (best solution since the start of the algorithm), current iteration best solution and restart best solution (the best solution generated since the last re-initialization of the pheromone). The set of solution are updated based on intensification strategy. Kong et al. (2008) claimed that its proposed approach is much faster comparing to other ACO-based algorithm because it bypasses the time-consuming calculation

of the dynamic heuristic value and constraint violation judgment. Although the paper was silent over the case when the heuristic information is static which is actually computed once at the start of an ACO based MKP algorithm. Lastly, BAS can only provide better solutions if the maximum iteration number is set to be larger (e.g. 6000 iterations) (Kong et al., 2008). However, in a similar approach but different transition probability, Lee and Bau (2012) proposed Preference-list ACO (PACOM). PACOM is similar to BAS in the sense that both relaxes MKP constraints condition to allow construction of all possible solutions but applied repair operation on potential solution that are not feasible to generate a feasible solution and update pheromone trail to a set of solutions known as preference list in PACOM. But the two methods differ in transition or selection probability in which PACOM uses model similar to ACO, which depends on both pheromone trail and heuristic information. PACOM uses static heuristic information computed as Profits v_i over summation of consumption (w_{ij}) times a set of shadow price of i th constraint (μ_i) (i.e. surrogate multipliers):

$$\eta_j = \frac{v_i}{\sum_{i=1}^m \mu_i w_{ij}} \quad (3.20)$$

PACOM performed competitively with Alaya et al. (2004) in only 100 iterations (Lee & Bau, 2012).

However, Fidanova (2007) compare Ant Colony System (ACS) – whose main idea is to enforce the performance of the best-found solution at the same time to diversify search, Max-Min Ant System (MMAS) – whose main features are to exploit the best-found solution after each iteration with only one ant add pheromone and avoid stagnation of the search by limiting the pheromone values to fixed interval $[\tau_{min}, \tau_{max}]$. Ant Colony Optimization with Additional Reinforcement (ACO-AR) – the main idea is to additional pheromone to unused elements after pheromone update has taken place. This is to forced ants to exploit a search

space which has not been exploited and to avoid bad experience (Fidanova, 2007). With each of this Ant algorithm the study implements MKP and tested different model of laying pheromones and heuristic information models. Best results were achieved with pheromone laid on arcs that links nodes, static heuristic information whose parameters includes profit, expenses and budget, and ACO-AR algorithm.

Finally, the ACO for MKP proposed in this thesis uses strictly adhered to the MKP constraint conditions, but allowed construction of all possible potential solutions and check for violation of the constraints, however instead of repairing the infeasible solution, any infeasible solution is simply ignored, doing so reduces the overhead needed to perform repair operations. In addition, the proposed method uses static heuristic information with profit or value and consumptions (weights) as parameters. Chapter 7 of this thesis presents ACO for MKP specifically as hybrid feature selection method.

3.10 Literature Review of the Related Information Theorem Feature Selection Methods

Information theoretic feature selection methods are usually based on the mutual information because, mutual information provides a quantitative measure of amount of dependency information (linear and nonlinear) two or more variables contain about each other (Song et al.; 2012). Ditzler (2015) provides the general form of the objective function of information-theoretic methods as:

$$J(X, \mathcal{F}) = \text{Relevance}(X, Y) - \text{Redundancy}(X, \mathcal{F}) \quad (3.21)$$

where $\text{Relevance}(X, Y)$ measures the relevancy between a feature X and the output attribute Y , and $\text{Redundancy}(X, \mathcal{F})$ measures the redundancy of a variable X with a selected feature subset $\mathcal{F}$ from the feature space $\mathcal{X}$. Thus, relevancy index is the defining component of all filter method (Brown et al., 2012) and very many hand-designed heuristics were proposed

with the aim at maximizing feature ‘relevancy’, while minimizing selected feature subset ‘redundancy’ (Duch, 2006; Brown et al., 2012).

Battiti in (1994) proposed the use of mutual information to evaluate feature from the feature space, he defined feature selection problem as “a process of selecting k most relevant variables from a feature space with m variables, $k \ll m$ ” (Battiti, 1994). Vergara and Estévez (2014) highlighted the four main assumptions of Battiti’s work as follows:

- I. features are classified as relevant and redundant.
- II. a heuristic function is used to select features, which allows controlling the trade-off between relevancy and redundancy.
- III. a greedy search strategy is used.
- IV. the selected feature subset is assumed optimal.

Battiti Mutual Information for Feature Selection (MIFS) algorithm is based on greedy selection of features that takes both the mutual information with respect to the output class and with respect to the already selected features into account (Battiti, 1994). Figure 3.6 is the original MIFS as proposed by Battiti in 1994 and presented in Battiti (2008).

Step1: the algorithms start with the initialization of set of n features F , and empty feature selected list S .

Step 2: for each feature f belonging to the F , estimate mutual information with respect to the class C (i.e. $I(C; f)$).

Step 3: select the first feature f from F with the maximum estimated mutual information value with respect to the class C . Remove the f from the set of n features and add it to S .

- 1) (Initialization) Set $F \leftarrow$ “initial set of n features;” $S \leftarrow$ “empty set.”
- 2) (Computation of the MI with the output class) for each feature $f \in F$ compute $I(C; f)$.
- 3) (Choice of the first feature) find the feature f that maximizes $I(C; f)$; set $F \leftarrow F \setminus \{f\}$; set $S \leftarrow \{f\}$
- 4) (Greedy selection) repeat until $|S| = k$:
 - a) (Computation of the MI between variables) for all couples of variables (f, s) with $f \in F$, $s \in S$ compute $I(f; s)$, if it is not already available.
 - b) (Selection of the next feature) choose feature f as the one that maximizes $I(C; f) - \beta \sum_{s \in S} I(f; s)$; set $F \leftarrow F \setminus \{f\}$; set $S \leftarrow S \cup \{f\}$
- 5) Output the set S containing the selected features.

Figure 3.6: The Original Battiti’s MIFS Algorithm

Step 4: use greedy process to compute mutual information estimation for all pairs of (f, s) with $f \in F$, $s \in S$ if it is not already available. Select the next feature $f \in F$ that maximizes $I(C; f) - \beta \sum_{s \in S} I(f; s)$. Update F and S by removing f from F and add it to S .

Step5: output the selected feature set S if the number of features in S are equal to k .

The objective function of MIFS is expressed as:

$$\mathcal{J}_{mifs}(X, \mathcal{F}) = I(X; Y) - \beta \sum_{s \in S} I(f; s) \quad (3.22)$$

The objective function does not only maximize dependency criterion, but includes a term that reduced inter-feature correlation (Brown, 2009; Vergara & Estévez, 2014). β is empirically determined, when $\beta = 0$, it gives the simplest objective function $I(X, Y)$ known as mutual information maximization (Brown, 2009) and eventually dropped the redundancy term.

However, setting β to large value, will place more emphasis on reducing inter-feature dependency.

Following the proposal of MIFS by Battiti in (1994), many other mutual information based feature selection with similar idea were proposed.

Yang and Moody (1999) proposed a Joint Mutual Information (JMI) as a feature selection criterion function.

$$\mathcal{J}_{jmi} = \sum_{k=1}^{n-1} I(X_n, X_k; Y) \quad (3.23)$$

Essentially, JMI find the joint mutual information between pairs of feature variables and output variables and effectively captures redundancy term. The approach is based on the second order interaction and thereby increases the cost of computation. Evaluating joint mutual information $(n - 1) -$ times may lead to exhaustive search which is prohibitive due to the cost involved.

Kwak and Choi (2002) proposed MIFS-U as an improvement over MIFS to handle nonlinear problems since MIFS works well in a linearly dependent dataset (Kwak & Choi, 2002).

MIFS-U is based on the criterion function:

$$\mathcal{J}_{mifsu} = I(X_n; Y) - \beta \sum_{k=1}^{n-1} \frac{I(X_k; Y)}{H(X_k)} I(X_n; X_k) \quad (3.24)$$

by default, $\beta = 1$, and the algorithm is effective when information is distributed uniformly (Kwak & Choi, 2002).

Fleuret (2004) proposed Conditional Mutual Information Maximization (CMIM) as an intermediate solution that captured redundancy term. The approach deals with the tradeoff between relevance of a feature and redundancy in feature subset. Its objective function is defined as:

$$\mathcal{J}_{cmim} = \min_k [I(X_n; Y | X_k)] \quad (3.25)$$

$\mathcal{J}_{cmim}$ scores ($s(n, k)$) low when X_n is independent of Y or provides redundant information to already selected X_k . The objective function is re-written for clarity by (Brown, 2009) as:

$$\mathcal{J}_{cmim} = I(X_n; Y) - \max_k [I(X_n; X_k) - I(X_n; X_k | Y)] \quad (3.26)$$

Peng et al. (2005) propose the Maximum-Relevance Minimum-Redundancy (MRMR) criterion:

$$\mathcal{J}_{mrmr} = I(X_n; Y) - \frac{1}{n-1} \sum_{k=1}^{n-1} I(X_n; X_k) \quad (3.27)$$

MRMR is a modification of MIFS when $\beta = \frac{1}{n-1}$, this means it takes the mean of the redundancy term (Brown, 2009).

Verleysen et al. (2009) proposed a non-parametric relevancy criterion measure that is based on Kozachenko-Leonenko k -nearest neighbor (KNN) entropy estimator (Kozachenko & Leonenko, 1987). Verleysen et al. (2009) defined $\hat{I}(X, Y)$ as:

$$\hat{I}(X, Y) = \psi(K) + \psi(N) - \frac{1}{N} \sum_{n=1}^N [\psi(\tau_x(n)) + \psi(\tau_y(n))] \quad (3.28)$$

Where $\psi(\cdot)$ is the digamma function, τ_x and τ_y are point in X and Y respectively strictly less than ϵ_n , N is the total number of data points and K is the parameter of the algorithm that needs to be set with care. Details can be found in Verleysen et al. (2009).

The KNN mutual information maximization was proposed to address the empty space problem associated with estimating *pdf* of variables X using histogram or Parzen window methods, especially when X is large. However, the KNN has its own drawback as the parameter (K) must be chosen with care and the accuracy of the estimator decrease as the dimension of X space increases (Verleysen et al., 2009).

However, Verleysen et al. (2009) proposed greedy search strategy that formed the basis of the improved greedy forward method proposed in this thesis. The paper presents forward-backward search as follows:

Forward search comprises of selecting the first feature that maximizes the mutual information with the output Y :

$$X_{s1} = \underset{X_j, 1 \leq j \leq M}{\operatorname{argmax}} \hat{I}(X_j, Y) \quad (3.29)$$

Subsequently and sequentially, $t - th$ features is selected as:

$$X_{st} = \underset{X_j, 1 \leq j \leq M, j \notin \{X_{s1}, X_{s2}, \dots, X_{st-1}\}}{\operatorname{argmax}} \hat{I}(\{X_{s1}, X_{s2}, \dots, X_{st-1}, X_j\}, Y) \quad (3.30)$$

Where $t \geq 2$ and X_{st} is the next candidate feature to be selected.

Verleysen et al. (2009) presented backward procedure as it is used in many feature selections as means of preventing the forward procedure from being stuck in a local minimum. A feature is remove based on the Equation (3.31) and Equation (3.32) on the basis that its removal will increase the estimated mutual information value.

$$X_{sd} = \underset{X_j, 1 \leq j \leq t-1}{\operatorname{argmax}} \hat{I}(\{X_{s1}, X_{s2}, \dots, X_{sj-1}, X_{sj+1}, \dots, X_{st}\}, Y) \quad (3.31)$$

X_{sd} from Equation 2.11 is removed if:

$$\hat{I}(\{X_{s1}, X_{s2}, \dots, X_{sd-1}, X_{sd+1}, \dots, X_{st}\}, Y) > \hat{I}(\{X_{s1}, X_{s2}, \dots, X_{st}\}, Y) \quad (3.32)$$

Even though theoretically, Equation (3.32) looks trivial, since mutual information value is always positive value or zero, in practice it holds since Equation (3.30) and Equation (3.31) are estimated values of mutual information (Verleysen et al., 2009). Both forward and backward procedures need additional constraints to limit the number of feature selected, otherwise the procedure will lead to full set of M . Traditional stopping strategy relies on continuously adding to or removing a feature from feature subset and come to stop only when

addition or removal of a new candidate feature does not lead to improvement in a current feature subset (Blum & Langley, 1997; Dash & Liu, 2003). In another stopping strategy, a feature is continuously added to or removed from feature subset until reaching the other end of search space and then select the best (Blum & Langley, 1997). Alternatively, search and evaluation stopped when evaluation value begins to decrease (Blum & Langley, 1997; Verleysen et al., 2009). A question was then asked whether the reduction in evaluation value is only due to bias or noise in an evaluator, or it has a sound link to the no or low relevance of a feature? (Verleysen et al., 2009). In other words, one is then forced to ask a question whether the traditional stopping strategy is scientifically sound? Verleysen et al. (2009) proposed a resampling stopping criterion function that is based on null hypothesis. The same idea is used in this thesis.

Doquire and Verleysen in (2011) proposed an effective feature selection method based on mutual information for uncertain data. The proposed method modeled uncertainty of the data by presenting any value in the data set as an uncertainty region and defined probability density function (PDF) over it (Doquire & Verleysen, 2011). Choosing a Gaussian PDF centered in the value and with a well-chosen variance resulted in a good result (Doquire & Verleysen, 2011).

Previous method from MIFS and its variants have avoided estimation of mutual information between two continuous variables because, the estimation of PDF of variables and integration of those functions is difficult (Kwak & Choi, 2002b) instead, histograms is used in approximating the PDFs so as to avoid these complexities. This practice, lead to degrading performance due to the large errors in estimating the mutual information (Kwak & Choi, 2002b).

However, Doquire and Verleysen (2011) proposed method, estimate mutual information based on entropy defined as:

$$I(X; Y) = h(Y) - h(Y/X) \quad (3.33)$$

Where both $h(Y)$ and $h(Y/X)$ are estimated as:

$$\hat{h}(Y) = -\sum_{i=1}^k \hat{p}(y_i) \log \hat{p}(y_i) \quad (3.34)$$

$$\hat{h}(Y/X_j) = -\int_{X_j} \hat{f}_{X_j}(x) \sum_{i=1}^k \hat{f}_{Y_i|X_j}(y_i|x) \log \hat{f}_{Y_i|X_j}(y_i|x) dx \quad (3.35)$$

Equation (3.34) is a discrete version of $h(X)$ for continuous variable, which is defined as:

$$h(X) = -\int f_x(X) \log f_x(X) dx \quad (3.36)$$

Equation (3.34) indicates that Doquire and Verleysen (2011) proposed method like the previously review methods, is limited to classification problems and $\hat{h}(Y/X_j)$ is the estimated density of the i^{th} class conditional to the j^{th} feature. The authors reduced the estimated value of $\hat{I}(X; Y)$ to the conditional term only, because the author believes that $\hat{h}(Y)$ is constant for all $X_j: 1 \leq j \leq m$ features (Doquire & Verleysen, 2011). While this is a good idea since MI serves as feature selection criterion and meant to be used to compare variables, in a practical sense it will lead to underestimation of estimated MI value. Meanwhile, $\hat{f}_{Y_i|X_j}(y_i|x)$ in Equation (3.35) is rewritten based on Bayes theorem as:

$$\hat{f}_{Y_i|X_j}(y_i|x) = \frac{\hat{f}_{X_j|Y_i}(x|y_i)\hat{p}(y_i)}{\hat{f}_{X_j}(x)}. \quad (3.37)$$

then Equation 2.15 becomes:

$$\hat{h}(Y/X_j) = -\int_{X_j} \hat{f}_{X_j}(x) \sum_{i=1}^k \frac{\hat{f}_{X_j|Y_i}(x|y_i)\hat{p}(y_i)}{\hat{f}_{X_j}(x)} \log \frac{\hat{f}_{X_j|Y_i}(x|y_i)\hat{p}(y_i)}{\hat{f}_{X_j}(x)} dx. \quad (3.38)$$

the Authors used Equation (3.38) to estimate MI value using PDF of variable X_j alone and limited to the point with class label y_i . This method is only limited to classification tasks. However, this thesis adopts the same approach as proposed by Doquire and Verleysen (2011)

and extended the mutual information formulation to regression tasks, and claim that the extended MI formulation is sufficient for both types of problems since discrete variables are natural subset of continuous variables.

3.10.1 Review of Relevant, Irrelevant and Redundant Features

Blum and Langley (1997) highlighted that different definitions of relevant feature exist in the machine learning literature and the reason for that lies on the question “relevant to what?”. Relevant information is necessary in the field of machine learning since it addresses larger and complex tasks. However, Yu and Liu (2004) defined the optimal feature set as composed of all strongly relevant features and the weakly relevant but not redundant features. They classify feature space from a dataset with respect to the predicting concept into three; strongly relevant, weakly relevant and irrelevant features. Features that are always necessary for an optimal subset are referred to as strong relevance features, their removal affects the conditional class distribution. Weak relevance features are features that are not always necessary, but may be necessary for optimal subset under certain conditions. Lastly, features that are not necessary are referred to irrelevant features, this includes features that completely irrelevant to the target attribute, and feature that is redundant given other features (Koller & Sahami, 1995). In general, a good set of features should not only be individually relevant, but also should not be redundant with respect to each other (Yu & Liu, 2004). And handling redundancy could lead to improved performances, especially when working with data that are highly redundant (Guyon & Elisseeff, 2003; Unler et al., 2011).

The next section provides research gap analysis of some of the old and recent important reviewed on mutual information estimation and information theoretic feature selection based on mutual information.

3.10.1 Research Gap Analysis

Since the last two decades, several approaches to feature selection using information theorem have been proposed and many of which are based on mutual information estimation. Brown (2009) and Vergara and Estévez (2014) have provided a unifying framework of such filter, the review of many of which are in Section 3.10.

3.10.1.1 Mutual information estimate

Battiti's MIFS (Battiti, 1994) and its variants such as JMI (Yang & Moody, 1999), MRMR (Peng et al., 2005) and MRNET (Meyer et al., 2007) are all low-order approximations to the conditional likelihood optimization problem (Vergara & Estévez, 2014). And they all underestimate mutual information, because the methods avoid direct computation of PDFs and their integral functions, instead they relied on histogram method of estimating PDF of a variable, as a result of that they have large errors in estimating the mutual information (Kwak & Choi, 2002b). besides that, they are all designed to estimated classification tasks (Battiti, 1994; Yang & Moody, 1999; Kwak & Choi, 2002a; Fleuret, 2004; Pen et al., 2005; Meyer et al., 2007; Verleysen et al., 2009; Doquire & Verleysen, 2011; Brown et al., 2012; Vergara & Estévez, 2014; Gao et al., 2015; Sluga & Lotric, 2017).

The only few proposed mutual information estimators that can be used to directly estimate mutual information with continuous variables without first discretizing a continuous variable into discrete are the k -nearest neighborhood structured mutual information estimators such as k -nearest-neighbors (Kraskov et al., 2004; Da'vid et al., 2010; Po'czos et al., 2011); kernel density estimate (Singh and Poczos, 2014); maximum likelihood density ratio (Suzuki et al., 2008); ensemble methods (Moon & Hero, 2014) and so on. With strongly dependents variables, all of those methods underestimate the mutual information (Gao et al., 2015a). However, Gao et al. (2015a) addresses this shortcoming by introducing a local non-uniformity

correction, but their estimator depended on a heuristically defined threshold parameter and lacked performance guarantees (Gao et al., 2015a). All of the methods as mentioned above including Gao et al. (2015a) do not directly estimate original Shannon (1994) entropy, instead they are based on highly non-parametric formulation of the estimate that circumvent estimation of PDF of variables and integration of those functions (Kwak & Choi, 2002b).

Despite that kernel density estimation is known to be affected by the curse of dimensionality or empty space phenomenon as the variable dimension increases, it is generally believed to be smoother than the other nonparametric methods of density estimation (Eggermont & LaRiccia, 2001; Doquire & Verleysen, 2011; Gao et al., 2015). Thus, the need to propose a methodology that can estimate mutual information for regression tasks based on the original Shannon entropy has been sighted in the literature. This thesis proposed a methodology based on Gaussian kernel method that enable mutual information estimates for continuous variables from data; this has been reported to be very difficult in the past (Kwak and Choi, 2002; Guyon & Elisseeff, 2003; Doquire & Verleysen, 2011; Unler et al., 2011; Brown et al., 2012; Benoît et al., 2013; Vergara & Estévez, 2014; Gao et al., 2015; Sluga & Lotric, 2017).

3.10.1.2 Greedy Feature Selection

The need to use greedy search for feature selection as opposed to an exhaustive search has been established in the literature (Detail in Section 3.4). Greedy search procedure along with relevance criterion function have been highlighted to be the main ingredient of feature selection (Verleysen et al., 2009). Also, it has been highlighted in the literature that using forward greedy procedure have advantages over the backward since the maximum size of the vectors (feature sets) that have to be evaluated by the criterion is equal to the final subset size (Reshef et al., 2011).

The original greedy feature selection method which is a forward procedure, by default have some drawbacks which include:

- It may give suboptimal and the initial choices always influence the final choice (Meyer et al., 2008; Verleysen et al., 2009).
- A variable cannot be deleted from the feature subset once it has been added.
- Relevant features are selected by ranking k features with the highest maximized relevancy measure. Selection on the basis of such ranking does not ensure weak dependency among features and could lead to redundancy (Guyon & Elisseeff, 2003; Fleuret, 2004).
- Finally, k feature selection condition does not guarantee stopping of the algorithm, it is possible to continue until all m features are selected.

Since Battiti proposal in 1994, many of the feature selection methods, particularly information theorem based feature selections adopted the same search methodology and only deferred in terms of evaluation criterion function formulation (Battiti, 1994; Blum & Langley, 1997; Yang & Moody, 1999; Kwak & Choi, 2002a; Fleuret, 2004; Pen et al., 2005; Meyer et al., 2007; Doquire & Verleysen, 2011).

Apart from the inherent drawback associated with sequential nature of the greedy forward search procedure through which the initial choice of feature influences the final choice (Meyer et al., 2008; Verleysen et al., 2009), most of the other drawbacks were associated with how to capture the relevancy and redundancy terms of the feature selection objective function within the greedy forward procedure. Battiti MIFS scheme and its variants are either attempted to capture or have captured the redundancy terms and instead of removing the redundancy, they are simply designed to penalize the relevance of a redundant feature, so that to reduce the chance of selecting such redundant feature. But obviously, this will only favor

the selection of perfectly correlated features and to the disadvantage of a weak redundant feature. Perfectly correlated features are truly redundant and provide no useful information to the selected feature subset (Guyon & Elisseeff, 2003; Fleuret, 2004). Hence, selecting strongly relevant variables is usually suboptimal for building a predictor most especially if the variables are redundant (Guyon & Elisseeff, 2003). Conversely, while irrelevant features need to be removed completely from the selected feature subset, strongly relevant features must be included in the optimal subset and weakly relevant features may become necessary for an optimal subset at certain conditions (Yu & Liu, 2004; Unler et al., 2011). Thus, the need for efficient feature selection method that can effectively remove irrelevant features from selected optimal feature subset and minimized the presence of weakly relevant features in the selected optimal feature subset is still an unresolved issue in feature selection community.

This thesis focuses on efficient feature selection method for both classification and regression supervised learning tasks with more emphasis on regression problems. The proposed IFS first addresses some of the highlighted drawbacks of the greedy forward procedure as follows:

- I. The proposed improved greedy forward procedure introduces a random selection of the next candidate feature step to address the sequential nature of the original greedy forward procedure designed by Battiti (1994) which makes it possible for the first feature selected to influence the subsequently selected features. By making the selection of the next candidate feature random it gives fair chances of selection to all features from a feature space. In Battiti's scheme, a feature with the highest MI value in the feature space after the first selected feature is removed from the space is automatically the next candidate feature. However, to avoid randomly selecting an irrelevant feature or very weakly relevant feature, the candidate feature is passed through null hypothesis stopping criterion test; one to check if the next candidate

feature is an irrelevant feature and two to serve as a scientifically sound stopping criterion for the proposed IFS method.

- II. The proposed IFS method further addresses the suboptimal feature selection problem by allowing the selection process to run t times after each cycle is terminated by the stopping criterion. This way it maintains the list of local optimal subsets which is subsequently used to rank selected features based on their consistent selection in t -times runs.
- III. Finally, the proposed IFS method explicitly handles redundancy. The first stage of the proposed algorithm selects feature subset purely based on relevance without considering redundancy, this makes the process to be fast. At the later stage, the selected suboptimal feature subset is further reduced by removing additional irrelevant and weakly relevant features from the subset through threshold value that is set empirically and redundancy check. The irrelevant feature removed by the redundancy check at the later stage are those features that may be perfectly correlated with a feature selected in the optimal feature subset while the threshold value condition removed weakly relevant features that are very weak from the final optimal feature subset.

CHAPTER 4

INFORMATION THEORETIC-BASED FEATURE SELECTION (IFS) METHOD

This thesis claims that efficient feature selection for supervised learning can be achieved by dependency information measure between features and the target attribute. Moreover, that such feature selector will be useful to a wide range of common machine learning algorithms. This chapter presents Information Theoretic-based Feature Selection (IFS) in support of this claim. Subsequent chapters examine various components of IFS.

Section 4.1 presents Rationale and Motivation for a dependency-information approach to feature selection, based on literature survey and related work. Section 4.2 discusses the fundamental of Mutual Information Estimation based on the information theory concept and various properties of Mutual Information. Section 4.3 discusses the theories and ideas that formed the basis for its estimation from data. Followed by the formulation of IFS selection criterion function in Section 4.4 and finally, Section 4.5 presents the IFS method.

4.1 Rationale

As the world is currently witnessing a sudden burst of data due to advances in technologies, author of this thesis believes that effective feature selection method will continue to play a role in uncovering most silent features in the data. Also, it will make machine learning algorithms application to big data easy in many years to come (Guyon & Elisseeff, 2003; Collins & Liu, 2005; Hoi et al., 2012; Wang et al., 2013; Wu et al., 2014). Empirical studies by Blum and Langley (1997) & Guyon and Elisseeff (2003) indicate that building good prediction learning models requires a reduction in feature subset. In addition, feature subset reduction reduces the burden of data collection as well as a reduction in computational complexity. These advantages of feature selection are of economic importance most

especially in some domains that are capital-intensive and time-consuming to organize facilities and collect or derive variables enough for a traditional empirical method of analysis.

Previous work in machine learning feature selection focuses more on classification tasks rather than regression (Battiti, 1994; Yang & Moody, 1999; Kwak & Choi, 2002; Fleuret, 2004; Pen et al., 2005; Verleysen et al., 2009; Doquire & Verleysen, 2011; Brown et al., 2012; Vergara & Estévez, 2014; Gao et al., 2015; Sluga & Lotric, 2017). However, many areas of machine learning application such as oil & gas (Ghouti & Al-Bukhitan, 2010; Gringarten, 2012; Sulaiman & Labadin, 2015a), pharmaceutical (Sekulic et al., 1996; Blanco et al., 1999; Rossi et al., 2006), food (Ozaki et al., 1992; Rossi et al., 2006), textile industries (Blanco et al., 1997; Rossi et al., 2006) and so on, require predicting quantitative variables (Rossi et al., 2006). This thesis focuses on efficient feature selection method for both classification and regression supervised learning tasks with more emphasis on regression problems. The thesis proposed a methodology based on Gaussian kernel method that enable mutual information estimate for continuous variables from data; this has been reported to be very difficult in the past (Kwak & Choi, 2002; Guyon & Elisseeff, 2003; Doquire & Verleysen, 2011; Unler et al., 2011; Brown et al., 2012; Benoît et al., 2013; Vergara & Estévez, 2014; Gao et al., 2015; Sluga & Lotric, 2017). Thus, feature selection presented in this thesis is based on the hypothesis that:

*A relevant **feature subset** is one in which the **dependency information** between each feature in the subset and a predictive output is **high**. Yet **with minimal redundancy** among features in the subset.*

Mutual information, a concept based on information theory by Shannon (1948) is design as selection criterion function to ascertain the thesis hypothesis claim. The rationale for chosen

Mutual information as basis for evaluation strategy in the proposed IFS is not only limited to its theoretical justification, but also satisfied the two criteria of good feature selection outlined by Song et al. (2012) namely:

1. *Selection criterion function should be able to capture both linear and nonlinear functional dependence between input data and the target output.*
2. *Selection criterion function should be able to guarantees with high probability the preservation of the detected functional dependency in test data.*

Again, despite that mutual information unlike correlation coefficient which measures linear dependence only, is a popular choice for selection criterion function. (Battiti, 1994; Zaffalon & Hutter, 2002; Peng et al., 2005; Rossi et al., 2006; François et al., 2007), most of the previous estimation in the literature focus on classification tasks. That is tasks with classes label as the target output. The reasons for that can be connected to the fact that it is not clear or rather difficult to estimate mutual information for two continuous variables from data (Guyon & Elisseeff, 2003; Unler et al., 2011). Thus, regression tasks are often transformed into classes before the feature selection is applied (Unler et al., 2011). Such transformation will block one to consider redundancy between features (Doquire & Verleysen, 2011) since redundancy in features is subject to instances of features.

Thus, Section 4.1 to Section 4.4 focus on the formulation of mutual information as selection criterion function, for estimating mutual information for both regression and classification data. The formulation is based on the traditional kernel density estimation and depends on the idea of convolution theorem, partition interval, and factorial distribution. Despite that kernel density estimation is known to be affected by the curse of dimensionality or empty space phenomenon as the variable dimension increases, it is generally believed to be smoother than the other nonparametric methods of density estimation (Eggermont & LaRiccia, 2001;

Doquire & Verleysen, 2011; Gao et al., 2015). Finally, Section 4.5 presents the complete IFS method, which is an improved greedy forward algorithm for feature selection. Here, a candidate feature is added to a selected feature set and the algorithm come to stop based on the idea of resampling method, permutation operation, and the null hypothesis.

4.2 Fundamental of Mutual Information (MI) Estimation

This section discusses the fundamental of mutual information estimation based on the information theory concept and as well presents various properties of mutual information.

4.2.1 Ideas and Definition

An Entropy of a random variable X in an information theory concept first introduced by Shannon (1948). It measures the uncertainty associated with variable X . If $f_x(X)$ is the probability density function of a random variable X , the entropy of X is expressed as:

$$h(X) = - \int f_x(X) \log f_x(X) d_x \quad (4.1)$$

In a similar way, MI of two random variables X and Y is formularized based on probability density of X, Y and joint probability density of X and Y as:

$$I(X, Y) = \iint f_{X,Y}(x, y) \log \frac{f_{X,Y}(x, y)}{f_X(x)f_Y(y)} d_x d_y. \quad (4.2)$$

Assuming that a pair of continuous random variables X and Y are correlated, from probability theory the joint probability density function of X and Y can be expressed as:

$$P_{X,Y}(x, y) = P_Y(y|x)P_X(x) \quad (4.3)$$

Similarly, Equation (4.4) expresses the joint differential entropy of X and Y as:

$$h(X, Y) = h(X) + h(Y|X) \quad (4.4)$$

In words, the Equation (4.4) expresses the uncertainty about X and Y as equal to the uncertainty about X plus the uncertainty about Y depending on X . In the other way round, the uncertainty about X and Y is equal to the uncertainty about Y plus the uncertainty about X when Y is known as expressed in Equation (4.5):

$$h(X, Y) = h(Y) + h(X|Y) \quad (4.5)$$

Considering a learning system where the applications of a continuous random variable X to the input of the system, produces a continuous random variable Y as the output of the system (see Figure 4.1). If the differential entropy $h(X)$ by definition is the uncertainty about the system input X before the observation of the system output X and the conditional entropy $h(X|Y)$ is the uncertainty about the system input X after observing the output Y . Then, the difference $h(X) - h(X|Y)$, is the uncertainty about the system input X determined by observing the system output Y (Haykin, 2009). That is referring to mutual information between the system input X and the system output Y , denoted as $I(X, Y)$:

$$I(X, Y) = h(Y) - h(Y|X) \quad (4.6)$$

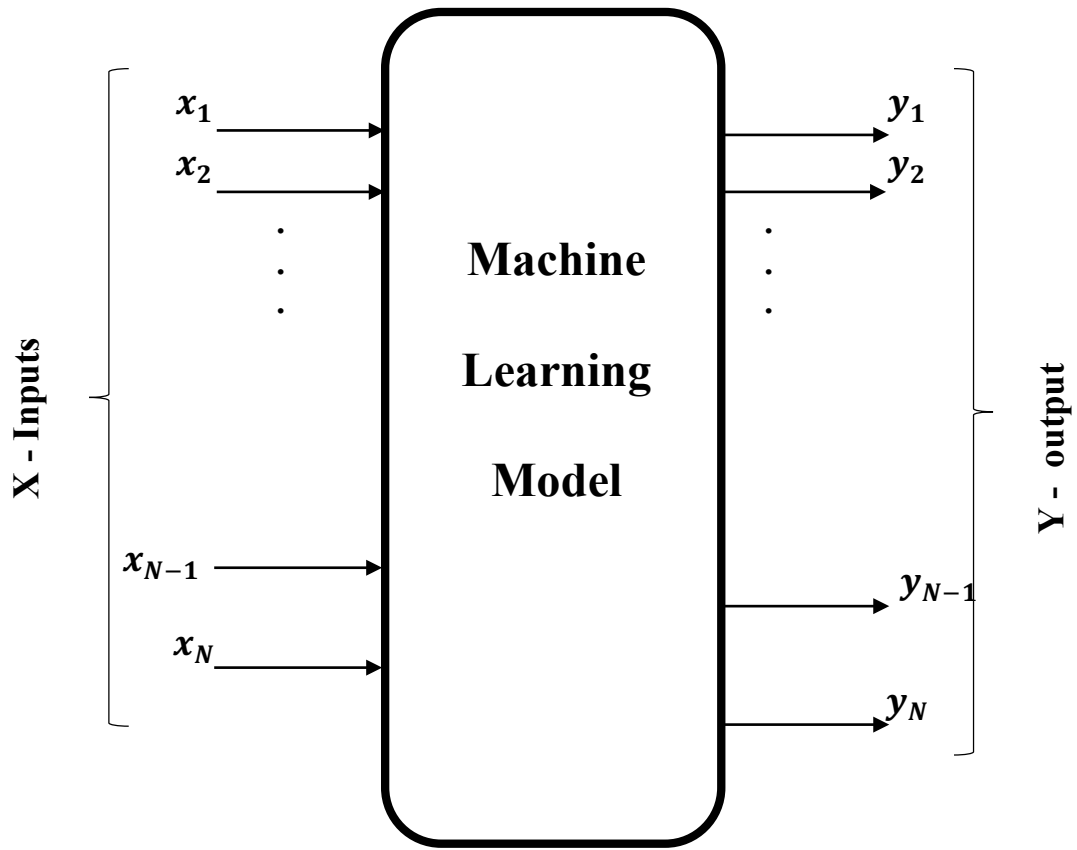


Figure 4.1: Learning System with Continuous Input Variable X and Continuous Output Variable Y

4.2.2 Properties of Mutual Information

The MI between two continuous random variables X and Y has three important properties as stated in Haykin (2009).

Property 1: Nonnegativity. $I(X, Y) \geq 0$

That is observing the system output Y does not lead to loss of information about system input X , and $I(X, Y) = 0$ only if the input and out variables of the system are independent. That is easily visible from Equation (4.6), statistically, if X and Y are independent then $h(X|Y) = h(X)$.

Property 2: Symmetry. $I(Y, X) = I(X, Y)$

Proof: this property can be proved based on Equation (4.5) and Equation (4.6).

$$\begin{aligned}
 I(X, Y) &= h(X) - h(X|Y) \\
 &= h(X) - (h(X, Y) - h(Y)) \quad (\text{by substituting } h(X|Y) \text{ with Equation (4.5)}) \\
 &= h(X) + h(Y) - h(X, Y) \\
 &= h(Y) - (h(X, Y) - h(X)) \quad (\text{by re-arranging terms}).
 \end{aligned}$$

$$I(X, Y) = h(Y) - h(Y|X) = I(Y, X) \quad (4.7)$$

Notice that Equation (4.7) is equal to $I(Y, X)$.

Property 3: Invariance. $I(X, Y) = I(U, V)$

Consider the invertible transformations: $u = f(x)$ and $v = f(y)$. Where x & y are sample values of the random variables X and Y . Also, u & v are equally sample values, for the transformable random variables U and V . Thus: $I(X, Y) = I(U, V)$ is the invariance property of mutual information.

Validation:

It can be validated intuitively through reasoning. Since the transformation of x to u and from y to v are both invertible, so there is no loss of information in the course of the two transformations.

4.3 Formulation of Mutual Information Estimator

This section provides the definition of some basic ideas that are put together to formulate the MI estimator from data in this thesis.

4.3.1 Basic Ideas

4.3.1.1 Definition 1: Joint Probability Distribution and Conditional Probability

When two or more variables, occupying a combined state, $X_1 = x_1, X_2 = x_2, \dots, X_n = x_n$, denoted by $P(x_1, x_2, \dots, x_n)$ or $P(X)$ is called joint probability distribution of X (Olshausen, 2004). such a distribution satisfies the following axioms:

1. $0 \leq P(X) \leq 1$
2. $\sum_x P(X) = 1$

Conditional probability expresses the interaction between variables: $P(x|y) = \frac{P(x,y)}{P(y)}$. Where, $P(x|y)$ refers to the conditional probability of $X = x$ given that $Y = y$. Thus, the joint probability distribution $P(x, y)$ is multiplication between $P(x|y)$ and $P(y)$.

4.3.1.2 Definition 2: Factorial Distribution

When the outcome of one variable has no effect on the other, a set of variables are said to be statistically independent (Olshausen, 2004). That is, for statistically independent variables, $P(x|y)$ is the same as $P(x)$. Thus, the joint probability distribution is said to be factorial:

$$P(x, y) = P(x)P(y)$$

In general, $P(x_1, x_2, \dots, x_n) = P(x_1) \times P(x_2) \dots \times P(x_n) = \prod_{i=1}^n P(x_i)$. This implies that:

$$\begin{aligned} P(X|Y) &= P(x_1|y) \times P(x_2|y) \times \dots \times P(x_n|y) \\ &= \prod_{i=1}^n P(x_i | y) \end{aligned} \tag{4.8}$$

4.3.1.3 Definition 3: Partition, Partition Interval, and Partition Refinement

Partition of a set X is a non-empty subset of X such that X is a disjoint union of the subsets (Olshausen, 2004). A partition of an interval $[a, b]$ on the real sets $\mathbb{R}$ is a finite sequence $X = (x_i)$ of real numbers such that: $a = x_0 < x_1 < x_2 < \dots < x_n = b$. Moreover, for every interval of the form $[x_i, x_{i+1}]$ is a sub-interval of the partition X .

Let α and ρ be each a partition of a set X , such that each monotonically increasing subset has a least upper bound and greater lower bound and $\alpha \leq \rho$. That is for every element in α is a subset of some element in ρ . Then partition α is said to be a refinement of the partition ρ .

4.3.2 Conventional Kernel Based Density Estimation

Consider $x_1, x_2, \dots, x_N$ where N is independent and identically distributed (i.i.d.) samples of f distribution. The probability density function (pdf) of X (i.e. $f(X)$) can be estimated as:

$$\hat{f}(X) = \frac{1}{Nb} \sum_{i=1}^N k\left(\frac{x-x_i}{b}\right) \quad (4.9)$$

Where k is the kernel and b is the bandwidth (also known as a smoothing parameter), b value is crucial in determining the quality of estimation. While different possible methods such as cross-validation and evolutionary search algorithm can be used to determine the value of b , in this thesis Silverman rule is used to determine the value of b in the same way as in Doquire and Verleysen (2011):

$$b_j = 1.06 \times \sigma_j \times N \quad (4.10)$$

j refers to the dimension of the data point, and σ_j is the standard deviation along the j th dimension of the dataset. Also, Gaussian kernel with zero mean and unit variance served as k :

$$k(x) = \frac{1}{\sqrt{2\pi}} e^{-0.5x^2} \quad (4.11)$$

4.4 Formulation of IFS Criterion Function Based on Mutual Information Estimation for Uncertain Data

Focusing on MI expression in Equation (4.7), that is, $I(X, Y) = h(Y) - h(Y|X)$.

Assuming that a dataset X contains N samples and d -features. Also, considering N as i.i.d. as stated above and a learning system where the application of a continuous variable X to the input of the system produces a continuous variable Y at the output of the system (see *Figure 4.1*). To find the *MI* estimate between X input and Y output, let evaluate $I(X_j, Y)$ for $j = 1, 2, \dots, d$ features.

Assuming that Y takes k -different partition $y^1, y^2, \dots, y^k$, such that every element y in Y is precisely in one of these k subsets. (refers to Subsection 4.3.1.3). Even though without the loss of generality, a continuous random variable Y includes discrete random variable Y . The two cases are considered separately in this thesis.

Case 1: when Y is discrete random variable, each y_i is represented by n_i - samples where $\sum_i n_i = N$ and the probability of $Y = y_i$ is denoted as:

$$\hat{P}(y_i) = \frac{n_i}{N} \quad (4.12)$$

So, the entropy of Y can be estimated as discrete version of Equation (4.1):

$$\hat{h}(Y) = -\sum_{i=1}^k \hat{P}(y_i) \log \hat{P}(y_i) \quad (4.13)$$

Case 2: when Y is a continuous random variable, each partition i of an interval $[a, b]$ is a finite sequence $Y = y^i$ of real numbers such that $a = y_0^i < y_1^i < \dots < y_n^i = b$ for n_i samples. Thus $\sum_i n_i = N$ and the probability of $Y = y^i$ is estimated as:

$$\hat{P}(Y = y^i) = \int_a^b f_y(y) dy \quad (4.14)$$

However, since N is independent and identically distributed (i.i.d.) samples of the f distribution, Y is assumed to be a factorial distribution (refers to Subsection 4.3.1.2). Thus, Equation (4.15) redefined the estimated probability of $\hat{P}(Y = y^i)$ as:

$$\begin{aligned}\hat{P}(Y = y^i) &= \hat{P}(y_{0i}^i) \times \hat{P}(y_{1i}^i) \times \dots \times \hat{P}(y_{n_i}^i) \\ &= \prod_{n_i} \hat{P}(y_{n_i}^i)\end{aligned}\quad (4.15)$$

Also, the entropy can be estimated as:

$$\begin{aligned}\hat{h}(Y) &= - \int f_y(y) \log f_y(y) dy \\ &= - \sum_i^k \prod_{n_i} \hat{P}(y_{n_i}^i) \log \hat{P}(y_{n_i}^i)\end{aligned}\quad (4.16)$$

Equation 4.16 is the expression for estimating the entropy of Y . The next focus is the estimation of the conditional entropy of X and Y variables. That is, the rightmost term of the lower part of the Equation (4.7) (i.e. $\hat{h}(Y|X)$):

$$\hat{h}(Y|X_j) = - \int_{x_j} \hat{f}_{x_j}(x_j) \sum_i^k \hat{f}_{y^i|x_j}(y^i|x_j) \log \hat{f}_{y^i|x_j}(y^i|x_j) dx \quad (4.17)$$

Equation (4.17) is the entropy of partitions of Y over each of X features. Based on Bayes theorems, $\hat{f}_{y^i|x_j}(y^i|x_j)$ in Equation (4.17) can be re-written (Kwak & Choi, 2002b; Doquire & Verleysen, 2011) as:

$$\hat{f}_{y^i|x_j}(y^i|x_j) = \frac{\hat{f}_{x_j|y^i}(x_j|y^i) \hat{P}(y^i)}{\hat{f}_{x_j}(x)} \quad (4.18)$$

Now, substituting Equation (4.18) into Equation (4.17) gives:

$$\hat{h}(Y|X_j) = - \int_{x_j} \hat{f}_{x_j}(x) \sum_i^k \frac{\hat{f}_{x_j|y^i}(x_{n_i}|y^i) \hat{P}(y^i)}{\hat{f}_{x_j}(x)} \log \frac{\hat{f}_{x_j|y^i}(x_{n_i}|y^i) \hat{P}(y^i)}{\hat{f}_{x_j}(x)} dx \quad (4.19)$$

In a similar way, applying factorial condition to Y gives Equation (4.20):

$$\hat{h}(Y|X_j) = -\int_{X_j} \hat{f}_{X_j}(x) \sum_i^k \prod_{n_i} \frac{\hat{f}_{X_j|Y^i}(x_{n_i}|y^i)^{\hat{P}(y^i)}}{\hat{f}_{X_j}(x)} \log \frac{\hat{f}_{X_j|Y^i}(x_{n_i}|y^i)^{\hat{P}(y^i)}}{\hat{f}_{X_j}(x)} dx \quad (4.20)$$

Equation (4.20) provides a means to estimate the conditional entropy of X_j and Y solidly based on probability density function (pdf) of X_j only, and exclude data points that do not belong to a partition y^i . Even though Equation (4.16) could be ignored, since the interest here is to evaluate the amount of dependency information that each feature in the input holds about Y and its value is theoretically the same for each X_j in X . However, to avoid underestimating MI for general purposes, Equation (4.16) and Equation (4.20) are substituted into Equation (4.7) to give the final expression of the *MI* Estimation:

$$\begin{aligned} \hat{I}(X_j, Y) = & -\sum_i^k \prod_{n_i} \hat{P}(y_n^i) \log \hat{P}(y_n^i) \\ & + \int_{X_j} \hat{f}_{X_j}(x) \sum_i^k \prod_{n_i} \frac{\hat{f}_{X_j|Y^i}(x_{n_i}|y^i)^{\hat{P}(y^i)}}{\hat{f}_{X_j}(x)} \log \frac{\hat{f}_{X_j|Y^i}(x_{n_i}|y^i)^{\hat{P}(y^i)}}{\hat{f}_{X_j}(x)} dx \end{aligned} \quad (4.21)$$

Looking at the Equation (4.21), it estimates Mutual Information between a feature X_j and output variable Y by considering each data point belonging to y^i partition of Y (i.e. $y_{0i}, y_{1i}, \dots, y_{n_i}$). Next is the description of how to compute the pdf of X_j . After which Section 5.2 discusses the implementation details for both numerical integrations in Equation (4.21) and the partitioning method adopted for Y . However, the pdf of X_j is estimated based on Parzen-window density estimation. Essentially, it defines a kernel function at each data point, so that each observed value $x_{j,i}$ contributes to the pdf estimate of X_j (Parzen, 1962).

Note that $X_j = [x_{j,1}, x_{j,2}, \dots, x_{j,N}]$ defines the N samples of X_j whose data uncertain can be modeled using Gaussian pdf

$X_{j,1} \sim N(\mu_{j,1}, \sigma_{j,1}), X_{j,2} \sim N(\mu_{j,2}, \sigma_{j,2}), \dots, X_{j,N} \sim N(\mu_{j,N}, \sigma_{j,N})$. Where $\mu_{j,i}$ is an observed value for the j^{th} dimension of the i^{th} sample and $\sigma_{j,i}$ is the corresponding standard deviation to be determined by the user (Bi & Zhang, 2004; Doquire & Verleysen, 2011; Tsang et al., 2011). Replacing Equation (4.9) with the Equation (4.22) is a natural way to consider the expected value of the kernel k (Doquire & Verleysen, 2011; Tsang et al., 2011).

$$\hat{f}(x) = \frac{1}{Nb} \sum_{i=1}^N E[k(\frac{x-x_i}{b})] \quad (4.22)$$

The expected value of the kernel function $k(x)$ is calculated using the law of the unconscious statistician (LOTUS) (Ross, 2010):

Proposition 1:

- a. If X is a discrete random variable with probability mass function $P(x)$, then for any real-valued function g , $E[g(x)] = \sum_{x:p(x)>0} g(x)p(x)$
- b. If X is continuous random variable with probability density function $f(x)$, then any real-valued function g , $E[g(X)] = \int_{-\infty}^{\infty} g(x)f(x)dx$

Applying the proposition 1(b) to Equation (4.22) gives:

$$\begin{aligned} \hat{f}(x) &= \frac{1}{Nb} \sum_{i=1}^N \int_{x_{ji}} k(\frac{x_{ji}-x}{b}) \frac{1}{\sqrt{2\pi\sigma_{ji}}} e^{-0.5(\frac{x_{ji}-\mu_{ji}}{\sigma_{ji}})^2} dx_{ji} \\ &= \frac{1}{Nb} \sum_{i=1}^N \int_{x_{ji}} \frac{1}{\sqrt{2\pi}} e^{-0.5(\frac{x_{ji}-x}{b})^2} \frac{1}{\sqrt{2\pi\sigma_{ji}}} e^{-0.5(\frac{x_{ji}-\mu_{ji}}{\sigma_{ji}})^2} dx_{ji} \\ &= \frac{1}{N} \sum_{i=1}^N \int_{x_{ji}} \frac{1}{\sqrt{2\pi}b} e^{-0.5(\frac{x_{ji}-x}{b})^2} \frac{1}{\sqrt{2\pi\sigma_{ji}}} e^{-0.5(\frac{x_{ji}-\mu_{ji}}{\sigma_{ji}})^2} dx_{ji} \quad (4.23) \end{aligned}$$

The convolution of a pair of distribution is equivalent to finding the distribution of the sum of the corresponding independent random variables. As for Gaussian, the sum of Gaussian

random variables also has a Gaussian distribution. That is, $f \sim N(\mu_f, \sigma_f)$ & $g \sim N(\mu_g, \sigma_g)$ gives

a Gaussian distribution $C \sim N(\mu_f + \mu_g, \sqrt{\sigma_f^2 + \sigma_g^2})$. Also, formally stated as:

$$\begin{aligned} f * g &= \int_{\tau} \frac{1}{\sqrt{2\pi}\sigma_f} e^{-0.5(\frac{\tau-\mu_f}{\sigma_f})^2} \frac{1}{\sqrt{2\pi}\sigma_g} e^{-0.5(\frac{t-\tau-\mu_g}{\sigma_g})^2} d\tau \\ &= \frac{1}{\sqrt{2\pi(\sigma_f^2 + \sigma_g^2)}} e^{-0.5\frac{(t-(\mu_f+\mu_g))^2}{\sigma_f^2 + \sigma_g^2}} \end{aligned} \quad (4.24)$$

Comparing the first part of Equation (4.24) with the last part of Equation (4.23) gives the following settings:

$\tau = x_{ji}$, $\sigma_f = b$, $\sigma_g = \sigma_{ji}$, $t = \mu_{ji}$, $\mu_f = x$, and $\mu_g = 0$. Thus, Equation (4.23) becomes:

$$\hat{f}(x) = \frac{1}{N} \sum_{i=1}^N \frac{1}{\sqrt{2\pi(\sigma_{ji}^2 + b^2)}} e^{-0.5\frac{(x-\mu_{ji})^2}{\sigma_{ji}^2 + b^2}} \quad (4.25)$$

Equation (4.21) and Equation (4.25) are sufficient in finding the estimated entropy of Y , the estimated probability density function of X_j , and the marginal variable X_j about Y^i . Where j refers to each feature from the input variables X and i refers to each partition from target variable Y . The implementation details for both integration and partitioning method adopted for Y are discussed in the next chapter.

4.5 IFS method

The proposed IFS method is an improved greedy forward feature selection method inspired by the proposed forward-backward greedy search in Verleysen et al. (2009). The original forward greedy search proposed by Battati in 1994 is sequential by default (Battati, 1994). The first feature selected X_{s1} is the feature with a maximum estimated MI value from the feature space, subsequently and sequentially next feature X_{st} to be selected must be the one

with maximum MI value after the previously selected feature. One of the drawback of the forward procedure is that it may give suboptimal and the initial choices always influence the final choice (Verleysen et al., 2009). In the proposed improved greedy forward method, next feature X_{st} is randomly selected as next candidate feature and passed through stopping criterion check. The introduction of null hypothesis stopping criterion provide the proposed IFS method a scientific way to terminate the forward search procedure as well as guarantee within a certain level of significant that each next candidate feature X_{st} that does not lead to terminating the search procedure is a relevant feature. The proposed IFS method further adds one more constraint to the X_{st} before it can finally be added to a selected feature subset s^i or all together be ignored; that is, X_{st} is only added to the list of selected feature subset if it leads to increase in the estimated MI value of the selected feature subset. In other word, it is added only if it increases $\hat{I}(s^i, Y)$.

Table 4.1 presents the proposed Improved Greedy Forward Feature Selection Algorithm, with X_j belongs to the input features X , where $j = 1, 2, \dots, m$ features. The Y is a continuous output variable. The s^i is a subset of S , both the S and s^i are initially empty, where $i = 1, 2, \dots, n$ number of repeated simulations. The s^i is an array of length m that stored selected feature subset for each $ith - run$, S is $m \times n$ matrix. That is, each ith column of S is a list of length m , it stores a selected feature subset s^i and correspond to $ith - run$, where $i = 1, 2, \dots, n$ runs. The algorithm consists of ten steps, first seven of which is an improvement of the greedy forward feature selection is the default procedure. Step 1 select the first feature X_{s1} as feature with maximum estimated MI value in the same way as traditional forward greedy feature selection method. Step 2 select the next candidate feature randomly such that the candidate feature is not in the list of already selected feature subset. However, the candidate feature is subjected to two additional constraints test. The first condition check for the

stopping criterion (the steps 3 and 4) details of which is in Figure 4.2, this condition ensure that the algorithm comes to halt if the candidate feature fails the null hypothesis test. That is, if the candidate feature X_{st} is weak feature at certain statistical significant level, and that any claim of dependency between the candidate feature X_{st} and Y is probably by chance. The second condition (the steps 5 to 7) ensure that the inclusion of the candidate feature X_{st} (after passing stopping criterion test) brings about gain to MI estimation of the selected feature subset s^i . So, this helps in preventing selection of feature with a very weak or no dependency relationship. Steps 2 to 7 are repeated for each new candidate feature X_{st} and added to the least of selected feature subset s^i for the same i th –run, until the stopping criterion naturally terminate the procedure, then s^i the final selected feature subset is added to the corresponding i th column of S and next $(i + 1)$ th run start from step 1. The traditional greedy forward algorithm does not have scientifically sound stopping criterion, it relies on continuously adding to or removing a feature from feature subset and come to stop only when addition or removal of a new candidate feature does not lead to improvement in a current feature subset (Blum & Langley, 1997; Dash and Liu, 2003). Moreover, the selection of the candidate feature is based on the feature with a maximum estimated MI value (Battiti, 1994; Verleysen et al., 2009; Sulaiman & Labadin, 2016) even if the higher MI estimated value is probably by chance rather than relevancy.

Table 4.1: The Proposed IFS Algorithm

Input:	1. $X_j \in X: j = 1, 2, \dots, m$ 2. $s^i \in S: i = 1, 2, \dots, n. s^i = \emptyset, S = \emptyset$ 3. Y 4. $bin_j \in bin: bin = \{ \}$ 5. θ is predefined numeric value, where $\theta \geq 1$
Output:	<i>FS (the optimal feature subset).</i>

Step 1: While ($i \leq n$):
 Select $X_{s1} \in X: X_{s1} = \underset{X_j: 1 \leq j \leq m}{argmax} \hat{I}(X_j, Y)$. Update $s^i = s^i \cup X_{s1}$
 GOTO step 2. Otherwise, GOTO step 9.
Step 2: Randomly select $X_{st} \in X: X_{st} = X_j: 1 \leq j \leq m, j \notin \{s1, s2, \dots, st-1\}$
Step 3: Check for stopping criterion function.
Step 4: If stopping criterion is true, GOTO step 8.
 Otherwise GOTO step 5.
Step 5: Check for $\hat{I}(s^i \cup X_{st}, Y) \leq \hat{I}(s^i, Y); s^i: s^i \equiv \{X_{s1}, X_{s2}, \dots, X_{st-1}\}$
 condition.
Step 6: If step 5 is true, ignore X_{st} , GOTO step 2.
 Otherwise GOTO step 7.
Step 7: Update $s^i = s^i \cup X_{st}$. Then GOTO Step 2.
Step 8: Update $S = S \cup s^i$. : Increment i by 1 and GOTO step 1.
Step 9: with bin, X, S , and θ run the optimal feature subset selection algorithm to
 generate FS .
Step 10: End.

However, the step 1 to step 7 of the proposed IFS method like the traditional greedy forward method only maximizes the mutual information between feature X_j in the feature space X and output attribute Y without considering redundancy within selected feature subset. Besides that, it is still not immune from falling into the local minimum, and this may lead to suboptimal feature selection with redundancy in selected feature subset. In order to mitigate this problem, step 8 to step 9 serve as an enhancement of the IFS method, step 9 is designed to prevent the proposed IFS method from falling into local minimum and to minimizes redundancy between features in the finally selected feature subset. Step 8 repeats step 1 to step 7 n -times and each time, S (a superset of s^i) stored s^i subset as selected feature set for $i = 1, 2, \dots, n$ repetition.

Finally, Step 9 is an optimal feature subset selection procedure. The details of which is in Figure 4.3. Step 9 basically count the frequency of each feature X_j for $j = 1, 2, \dots, m$ from $s^i \in S$ for $i = 1, 2, \dots, n$ as $bin_j \in bin$. Then add feature X_j to final optimal feature subset FS if $bin_j \in bin$ is the modal value with at least a frequency equal to θ and X_j is not redundant to any feature X_k in FS , such that $1 \leq k \leq m$ and $k \neq j$.

4.5.1 Algorithm for Stopping Criterion Function

The null hypothesis stopping criterion function provides a scientifically sound means of halting feature selection algorithm, unlike the traditional method in which the feature selection algorithm comes to stop when the addition of a new feature resulted in no additional gain in MI value. The “*null hypothesis*” is an inferential statistic method for testing relationship between two measured phenomena, or association among group (Everitt, 1998). In this study, null hypothesis is used as basis for stopping IFS method. The research thesis hypothesizes that the dependency information between a candidate feature X_{st} and Y is relevant and should be added to a selected feature subset, but the null hypothesis says that any claim for statistical significance between X_{st} and Y is probably not true. Thus, the idea presented here involves generating two MI distributions, the first distribution is from X_{st} and the second distribution comes from randomly permuted version of $\rho(X_{st})$. Randomly permuting X_{st} is assumed to introduce independency between $\rho(X_{st})$ and Y , so that $\hat{I}(\rho(X_{st}), Y) < \hat{I}(X_{st}, Y)$. This study assumed that this is true for variables with high information dependency.

Figure 4.2 presents the algorithm for stopping criterion function of the IFS method. X_{st} is the candidate feature selected randomly to test for stopping and s^i is the previously selected feature subset. The estimated value of $\hat{I}(s^i \cup X_{st}, Y)$ is temporary stored as *ref*, while the

estimated value of $\hat{I}(s^i \cup \rho(X_{st}), Y)$ is generated p times by randomly permuting X_{st} . Each time an estimated value of a permuted MI distribution is generated, a comparison is made between the estimated value ref and the value of the permuted MI distribution. If the value of the permuted MI distribution is greater than or equal to ref a counter is incremented by 1. Finally, the p-value (which is the fraction of times estimated value of the permuted MI distribution is greater or equal to the actual estimated MI value, ref) is computed after p times and null hypothesis is tested based on the p-value at a particular significant level. If the null hypothesis is rejected, accept the candidate feature X_{st} (as significant feature) and continue to the next step (Step 5 of the Table 4.1). Otherwise, the algorithm halt.

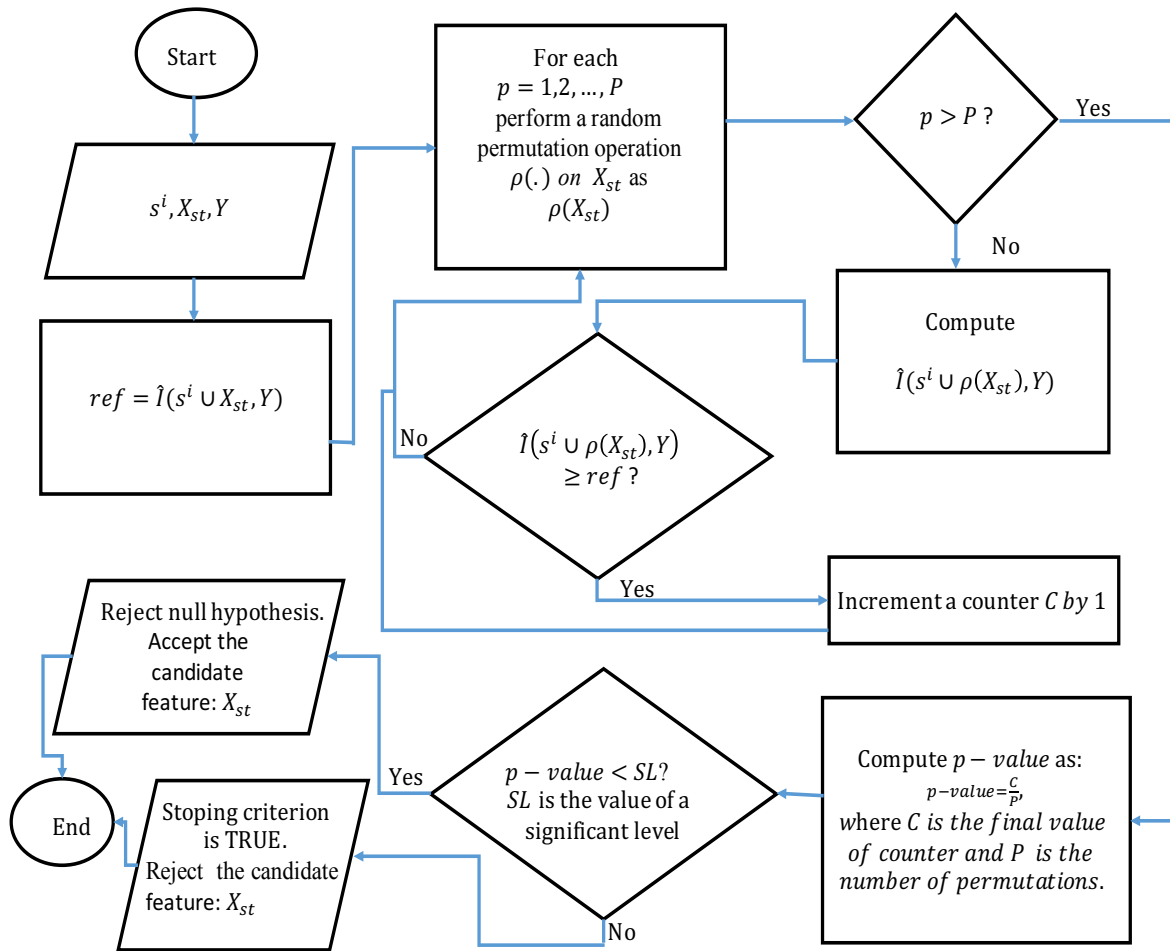


Figure 4.2: Algorithm for Stopping Criterion Function

Stopping Criterion Algorithm determines if the Feature Selection (IFS) Algorithm terminate after the previously selected feature subset in s^i , and this occur if the dependency information between a next candidate feature X_{st} and Y is proven through null hypothesis test to be by chance.

4.5.2 Algorithm for Optimal Feature Subset Selection Function

The algorithm for optimal feature subset selection is an enhancement on the proposed improved greedy forward method for feature selection (as explained in Section 4.5). The algorithm is introduced to mitigate the proposed IFS Method from selecting suboptimal feature subset. Inherently, the randomization nature of the greedy method may lead to local minimum feature selection.

The optimal feature subset selection algorithm uses statistical mode to rank features from feature subset space, selected by step1 to step 8 of the IFS Method. Mode may vary for a given distribution, since the probability density/mass function may take the same maximum value at several points $x_1, x_2, \dots, x_N$. Theoretically, when a probability density function has multiple local maxima, all the local maxima are referred as modes of the distribution. In addition, the optimal feature subset selection algorithm addresses the issue of redundancy in selected feature subset in an effective way.

As stated in Section 3.4, it is practically not feasible to test all the $2^M - 1$ non-empty feature subsets in order to select optimal feature subsets, particularly when M is considerably large. Thus, step 1 to step 8 of the proposed IFS method use heuristic method to select n feature subset from the $2^M - 1$ non-empty feature subsets, and subsequently use multiple modal approach to rank feature based on their high dependency information with the target variable

Y . This is based on the assumption that step 1 to step 7 of the IFS method select feature subset based on their high dependency information with the predictive output.

Figure 4.3 presents the algorithm for optimal feature subset selection procedure. S is $m \times n$ matrix that store the selected feature subset s^i in its i th-column after every complete run of step 1 to step 7 of the proposed IFS algorithm, with $i = 1, 2, \dots, n$ runs and $s^i = \{s_1^i, s_2^i, \dots, s_m^i\}$ is the feature subset selected where $s_j^i: s_j^i = \{f_j\} \mid \{\}$. Thus, each s^i may contain a minimum of one feature and a maximum of m features. Each bin_j in a bin partition is a counter for j feature in n – times IFS run. bin_k is a bin_j with modal value for $j = 1, 2, \dots, m$, $bin = bin - bin_k$ return a set of bin partitions after removing bin_k and θ is a threshold value that determines if a feature is to be added to the final selected feature subset FS . $\theta \geq 1$ and is determined by users empirically. If bin_k appears at least θ times in a total of n runs, select f_k for redundancy check. Finally, add f_k to the optimal feature set list FS if it is not redundant to any feature previously added to FS . The redundancy check procedure is designed to check for strongly dependency information between the modal feature f_k and a feature f_j in the optimal feature subset list FS . It checks for conditional mutual information between a feature in FS and the modal feature (i.e. $\hat{I}(f_j, f_k | Y)$) and compare with the mutual information between the corresponding feature f_j in FS and the output attribute Y (i.e. $\hat{I}(f_j, Y)$). Modal feature f_k is added to the FS only if $\forall f_j \in FS, \hat{I}(f_j, f_k | Y) < \hat{I}(f_j, Y)$. However, modal feature f_k is ignored at the first instance when $\hat{I}(f_j, f_k | Y) > \hat{I}(f_j, Y)$. This process is designed to address the drawback of the previous IFS methods that involved testing conditional independence for all possible features and estimate the probability density functions (pdfs) (Vergara, 2014). In the proposed IFS method, in the first stage, features are selected from a feature space on the basis of relevancy only (i.e. on the basis of maximum MI values) without considering

interdependency between all possible features. However, conditional independence testing is done only between some of the selected features in the optimal feature subset at the later stage of the proposed IFS method.

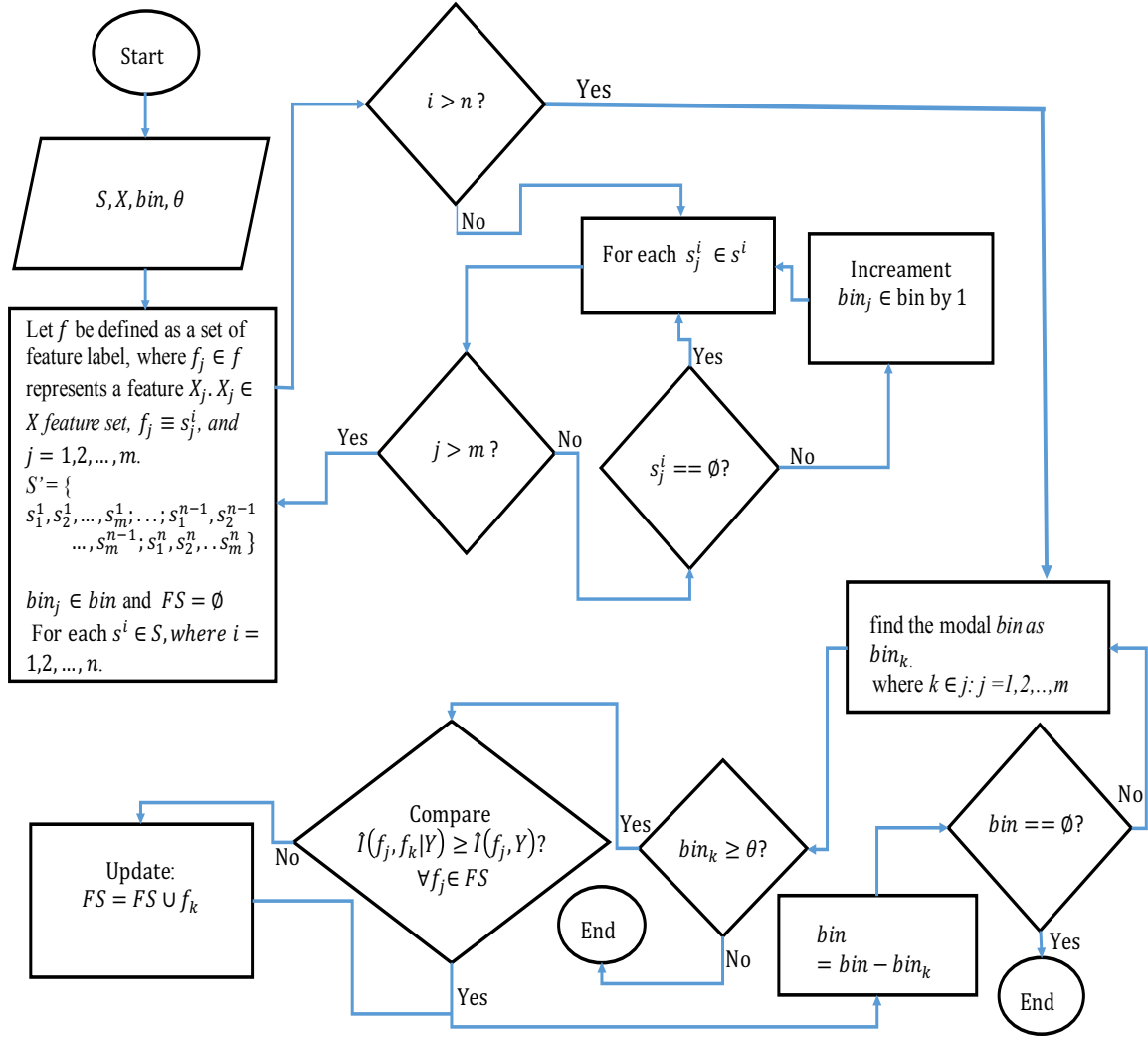


Figure 4.3: Algorithm for Optimal Feature Subset Selection Function

CHAPTER 5

DATASETS USED IN EXPERIMENT

This chapter consists of two sections. Section 5.1 discusses various datasets used to evaluate various components of IFS Method from a different domain. Finally, the experimental methodology for proper implementation of IFS method is discussed in Section 5.2.

5.1 Domains

This study used twelve datasets to evaluate IFS Method. Ten out of which are natural domain datasets obtained from the UCI Repository and used in several experiments with three Machine Learning Algorithms for both Classification and Regression Tasks as well as for comparison between IFS Method and other existing Feature Selection Benchmarks. The remaining two datasets are Artificial domains, generated from studied Functional Relationships. Many of the natural datasets were selected based on their usage in the literature while other are suitable for specific tasks. Table 5.1 presents the summary of the datasets from the UCI Repository and below is the brief summary of each of the dataset used in this thesis.

Table 5.1: Details of the Datasets from UCI Repositories³

Dataset	No. of instances	No. of features	Task type
Breast cancer	197	33	Classification & Regression
Mammographic mass data	289	5	Classification
Communities and Crime	530	99	Regression
Concrete	1030	8	Regression
Energy utilization	768	8	Classification & Regression*
Forest fire	517	12	Regression
Glass	213	9	Classification
Iris	149	4	Classification
Parkinson disease	4454	19	Regression*
Wine	177	13	Classification

³ * consist of two separate target variables, and gives two different simulations.

Breast Cancer: this dataset is from Wisconsin Prognostic Breast Cancer (WPBC) and donated to UCI Repository since 1995. The dataset is suitable for both supervised classification and regression tasks. The detail description of the dataset is in Street et al. (1995). In this thesis, Recurrent and Non-Recurrent Outcomes were used for Classification, while Time which represents Recurrence time and Disease free time was used for Regression, since it was quantitative variable and of significant to the dataset. In both cases, there are 197 Instances and 33 Features.

Mammographic Mass Data: this data is for Breast cancer screening, and can be utilized for predicting the Benign or Malignant severity of a Mammographic mass lesion from BI-RADS attributes and patient's age (Elter et al., 2007). The original dataset has a lot of missing attribute values, many of which were corrected and others removed. After the initial preprocessing, the dataset used in this thesis consist of 289 Instances. 166 Benign and 123 Malignant Class Distribution.

Communities and Crime: this dataset combines socio-economic data from the 1990 US Census, law enforcement data from 1990 US LEMA Survey, and crime data from the 1995 FBI- Uniform Crime Reporting (UCR). The data is described as multivariate with real attributes and it is suitable for Feature Selection test. With the supervised learning regression as associated task. The dataset is obtained from UCI machine learning repository and related dataset usage can be found in Redmond and Baveja (2002). This dataset was used for regression task only as indicated in Table 5.1.

Concrete: concrete compressive strength dataset was obtained from UCI Repository. It is multivariate data, a highly nonlinear function of age and ingredients. The ingredients include cement, blast furnace slag, fly ash, water, superplasticizer, coarse aggregate and fine

aggregate. The actual concrete compressive strength (Mpa) for a given mixture under a specific age (in days) was determined from a laboratory. It contains no missing data and has 1030 Instances and 8 quantitative input variables and 1 quantitative output variable which is the concrete compressive strength. Past usage of the dataset by the original owner is in Yeh (1998). In this thesis, the dataset was used for regression task only.

Energy Utilization: this dataset is about assessing the Heating and Cooling load requirements of buildings based on some building parameters. That is Energy Efficiency as a function of building parameters. A multivariate dataset which is associated with both Classification and Regression tasks. it consists of 8 features which include Relative Compactness, Surface Area, Wall Area, Roof Area, Overall Height, Orientation, Glazing Area, and Glazing Area Distribution. There are two predictive output variables; Heating load and Cooling load. In this thesis, this dataset was used for both classification and regression task in two categories. Each of heating and cooling loads is considered as a separate output variable and each is tested for classification and regression separately as suggested by the donor. Like other dataset, it was obtained from the UCI repository, and further details about the dataset can be found in the past usage by the donor Tsanas and Xifara (2012).

Forest Fires: the forest fires dataset was obtained from UCI Repository. It was described as very difficult regression task by the donor and suggested that it will be suitable to apply feature selection since several of the attributes are correlated. However, the dataset contains many outliers but not clear how many outliers are present. It contains 517 instances and 12 features with no missing data. Burned area of the forest (in ha) is the predicting output. Details of the forest fires can be found in the past usage by the donor in Cortez and Morais (2007). The dataset was used for regression task only in this thesis.

Glass: glass dataset is used to identify glass type and has been widely used in many supervised learning studies. 7 glass type distribution is the predictive output. The dataset consists of 213 instances, 9 of which are continuous input features. It is mainly associated with classification task and past usage in Evett and Spiehler (1987).

Iris: Iris plant dataset from UCI Repository is one of the most widely used benchmarks in Machine Learning Classification task. It consists of 4 numeric input attributes, 1 predictive output, and 149 instances. The predictive attribute contains three types of iris namely Iris setosa, Iris versicolor and iris virginica. It has no missing attribute.

Wine: dataset is obtained from UCI Repository, it consists of 177 Instances and 13 attributes. It is used for the classification task. The target output consists of three class type distribution with 59 instances class type 1, 71 instances class type 2 and 47 instances class type 3. Further details can be found in Aeberhard et al. (1992).

Parkinson Disease: Parkinson Telemonitoring Dataset was retrieved from the UCI Repository. It is a multivariate dataset suitable for regression task. It consists of 4454 instances, 19 input variables and 2 predictive continuous target outputs. Clinician's motor UPDRS score (motor_UPDRS) and clinician's total UPDRS score (total_UPDRS) both linearly interpolated are two separate target outputs. Details of the dataset can be found in the past usage in Little et al. (2009).

Artificial Dataset 1 (A1): A1 is an Artificial Dataset synthesized based on a functional relationship of the form $Y = f(x) + \mu(0, \theta)$, which is similar to the relationships that were studied in (Reshef et al., 2011; Kinney & Atwal, 2014; Gao et al., 2015). The $\mu(0, \theta)$ is noise term uniformly distributed over the interval $(0, \theta)$ with θ as the noise control level. Six synthetic datasets were generated using functions that range from linear to trigonometric: $Y =$

$X + \mu$, $Y = X^2 + \mu$, $Y = X^3 + \mu$, $Y = 2^X + \mu$, $Y = \sin(4\pi X) + \mu$, and $Y = \cos(5\pi X(1 - X)) + \mu$. Where X is randomly generated from a uniform distribution within the interval of $[0, 1]$ and consist of 2500 instances or data points. A1 is used in this thesis to test the MI estimate of IFS Selection Criterion Function by comparing with different existing MI estimator benchmarks. This methodology provides a means of testing or verifying formulated estimator in a controlled study. Further discussion is in Section 6.1.

Artificial Dataset 2 (A2): A2 like an A1 is the second Artificial Dataset synthesized based on the functional relationship of the form $Y = 10 \sin(X_1, X_2) + 20(X_3 - 0.5)^2 + 10X_4 + 5X_5 + \epsilon$. Where ϵ is a Gaussian noise with zero mean and unit variance, a similar relationship was studied in (François et al., 2007; Verleysen et al., 2009). This dataset was generated to test the discriminating ability of IFS Algorithm in a controlled study. 1000 instances were randomly generated for ten features $X_i: 1 \leq i \leq 10$ from uniform distribution within the interval $[0, 1]$ and are assumed to be iid.

5.2 Experimental Methodology

The experimental methodology adopted in this thesis follow the general Machine Learning supervised task (as shown in Figure 3.1). With the exception of Artificial Datasets all other datasets used in this thesis were meant to test the effectiveness of the IFS (Feature Selection) Method in a natural domain. Hence, two separate experiments were run with Machine Learning Algorithms, one without applying feature selection (full feature set) and the second experiment with feature selection (selected feature subset). In addition, IFS method is compared with five feature selection benchmarks using the performance of classification algorithm as metric for measuring the relative performance of the IFS Method.

The first step was to preprocess the dataset obtained from UCI Repository, those with few missing data were corrected by comparing k-nearest neighbors of the missing data point in a scientific way. However, where correction will prove difficult, the complete Instances were removed. Also, where necessary features are normalized with zero mean and unit variance (see details in Table 5.2). Furthermore, since IFS Method requires continuous output variable to be partitioned into real value intervals, simple binning discretization method is adopted in this thesis. It is a static, global and unsupervised discretization method (Catlett, 1992; Fayyad & Irani, 1993; Dougherty et al., 1995; Chmielewski & Grzymala-Busse, 1996) with k bin to be defined by the user. So, Y continuous output variable is divided into a partition i of an interval $[a, b]$, where $i = 1, 2, \dots, k$ partitions of Y , and the proposed IFS data formatting method preprocesses the partitions of Y into an $N \times k$ matrix called $y_formatted$ where $r = 1, 2, \dots, N$ rows correspond to the data points in Y (i.e. length of Y) and $i = 1, 2, \dots, k$ columns correspond to the partitions of Y . Each $y_formatted_{r,i}$ has a value of zero or a value of Y at $row - r$ in $column - i$. That is, all values of data points in Y belonging to $partition - i$ are placed into the corresponding $row - r$ and $column - i$ of $y_formatted$. In this thesis, k partition dataset is usually based on a domain expert knowledge (if available) or based on five statistical summaries which include minimum, lower-quartile, mean, upper-quartile and maximum values of dataset distribution. The five-statistical summary partitioning contains meaningful properties of a distribution.

Table 5.2: Some Details of the Dataset that Required Normalization and/or Discretization

Dataset	No. of instances	No. of features	Required Normalization? (Yes/No)	Required Discretization? (Yes/No)	Associated task
Breast cancer1	197	33	Yes	No	Classification
Breast cancer2	197	33	Yes	Yes	Regression
Mammographic mass data	289	5	Yes	No	Classification
Communities and Crime	530	99	No	Yes	Regression
Concrete	1030	8	Yes	Yes	Regression
Energy (cooling load)	768	8	Yes	Yes	Regression & classification
Energy (heating load)	768	8	Yes	Yes	Regression & classification
Forest fire	517	12	Yes	Yes	Regression
Glass	213	9	Yes	No	Classification
Iris	149	4	No	No	Classification
Parkinson (motor_updrs)	4454	19	Yes	Yes	Regression
Parkinson (total_updrs)	4454	19	Yes	Yes	Regression
Wine	177	13	Yes	No	Classification

The second step was sampling and splitting of a dataset into training and testing data. Random permutation (e.g. Knuth shuffles) of the original dataset was made and then splitting into two datasets; testing and training dataset in a ratio of either 1:4 or 3:7 respectively. The choice of dataset splitting ratio is done empirically based on some considerations. For instance, all Artificial Neural Networks (ANN) Algorithm and Multi-Regression Learning Implementation used 1:4 ratio splits, in the case of ANN, it uses 10-fold cross validation during training, enough training data is required for this purpose, however, in the case of Multi-Regression machine learning it requires more training data because the gradient descent learning rate is determined in a cross-validation manner so as to achieve a better result. Support Vector Regression (SVR) or Support Vector Machine (SVM) Learning Algorithms Implementation used 3:7 ratio splits for test and train data respectively because both use learning optimization

algorithms that do not require learning rate and are faster than the gradient descent. Table 5.3 contains details of some parameter settings used for machine learning training in this thesis.

The third step is the final preprocessing step which in this case, the feature selection. Training dataset is passed to IFS Method, which is a filter based as previously described in Chapter 4 of this thesis. For practical implementation of the IFS Method, the integral in Equation 4.21 is done numerically using the Trapezoidal Rule. Considering X_{ji} as continuous random variable on $[a, b]$, then $[a, b]$ is divided into n subintervals of equal length: $\Delta_x = \frac{b-a}{n}$. To evaluate the integral over X_{ji} for $\hat{f}_{X_j}(x_j)$ and $\hat{f}_{X_j|Y^i}(x_{n_i}|y^i)$ $n + 1$ points $x_0 = a, x_1 = a + \Delta_x, x_2 = a + 2\Delta_x, \dots, x_n = a + n\Delta_x = b$ are used. The mean μ_{ji} is the observed values and the standard deviation σ_j in this thesis is defined as studied in (Bi & Zhang, 2004; Doquire & Verleysen, 2011; Tsang et al., 2011) as $\sigma_j = 0.25 \times (\max_j - \min_j) \times w\%$. The w is the level of uncertainty about the values of X_j and it is proportional to the size of the range of values taken by the variable (Doquire & Verleysen, 2011). When the observed values of a given variable are in a close range, then the value of w for the variable is expected to be small. The value of w for feature X_j is determined empirically using cross-validation method with range of value intervals. Finally, the outcomes from the proposed IFS Method is a selected feature subset as stated in Table 4.1. Appendix D provides settings for w and size of partition for datasets used in this thesis.

The remaining steps are for testing the effectiveness of the IFS Method. The training of Machine Learning Models in this thesis involves the use of Learning Optimization Algorithms like Gradient Descent, Conjugate Gradient, and Quadratic Programming. The Gradient Descent was used with Multi-Regression, Conjugate Gradient with ANN and Quadratic Programming with SVM. Three Machine Learning Algorithms were used for

Regression Tasks namely ANN, Multi-Regression and support vector regression (SVR). As for classification tasks, only ANN and SVM were used. But in order to compare the proposed IFS method and some selected feature selection benchmarks, only SVM classifier was used for this purpose. However, with the exception of SVM, n -fold cross-validation was used for best model selection during training for all other learning algorithms (details as shown in Table 5.3). The SVM used in this thesis was based on MATLAB implementation of error-correcting output codes (ecoc) multiple models. It is one versus one (OVO) binary learners, that is for each binary learner one class is positive, another one is negative and the rest are ignored. It is an exhaust all combination of class pair assignments with $\frac{K(K-1)}{2}$ learners. For example, if $k = 2$, the learner is 1 and the coding matrix is $\begin{bmatrix} 1 \\ -1 \end{bmatrix}$, and if $k = 3$, the learners

are 3 and the coding matrix is designed as $\begin{bmatrix} 1 & 1 & 0 \\ -1 & 0 & 1 \\ 0 & -1 & -1 \end{bmatrix}$. Figure 5.1 illustrates the design of

coding matrix for $k = 1$ to $k = 5$

Table 5.3: Details of Machine Learning Algorithm Parameters Settings

Machine learning algorithm	Associated tasks	Learning Optimization Algorithm	Test-Training data ratio	Cross-validation method	Kernel function
ANN	both	Conjugate gradient	1:4	10-folds	Sigmoid
Multi-Regression	Regression only	Gradient descent	1:4	5-folds	Linear
Support Vector Regression (SVR)	Regression only	Quadratic programming	3:7	5-folds	Gaussian
SVM	Classification only	Broyden-Fletcher-Goldfarb-Shannon quasi-Newton algorithm (bfgs) or dual stochastic gradient descent (dua sgd)	3:7	One versus one	Radial basic

With n -fold cross validation and performance metrics, the optimal parameter settings for the best performing model of each machine learning algorithm is selected for final training using all the training dataset to generate final classification or regression model. Sample of the optimal parameter settings are available in Appendix D.

The final model (classification or regression) is evaluated using test dataset. Finally, the effectiveness of the proposed IFS Method is inferred by comparison between the performance of final model built that are based on selected feature subset and the performance of a final model that are based on actual full feature set from the dataset. Also, relative performance of IFS method is concluded by comparison between its performance and the performance of some selected well known filter based feature selection benchmarks using SVM under the same experimental conditions.

1									
-1									
1	1	0							
-1	0	1							
0	-1	-1							
1	1	1	0	0	0				
-1	0	0	1	1	0				
0	-1	0	-1	0	1				
0	0	-1	0	-1	-1				
1	1	1	1	0	0	0	0	0	0
-1	0	0	0	1	1	1	0	0	0
0	-1	0	0	-1	0	0	1	1	0
0	0	-1	0	0	-1	0	-1	0	1
0	0	0	-1	0	0	-1	0	-1	-1

Figure 5.1: Illustration of Coding Matrix of SVM Binary Learners for $k = 2$ up to $k = 5$

CHAPTER 6

EVALUATION OF THE PROPOSED IFS METHOD

This chapter presents various experiments carried out in this thesis to test various components and effectiveness of the IFS Method. Two artificial datasets were used to verify IFS Method without the use of Machine Learning Algorithms. However, the remaining datasets from UCI were used with Machine Learning Algorithms to test both classification and regression tasks. Also, the UCI datasets were used to compare the proposed IFS method with five other feature selection benchmarks.

Thus, Section 6.1 describes the use of artificial datasets to test the proposed IFS. The first artificial dataset (A1) was used to test IFS selection criterion function (i.e. the formulated mutual information estimator) in a controlled experiment. That is, the formulated MI estimator is compared with various related MI estimator benchmarks. This evaluation is necessary in ensuring that the first thesis objective is satisfied. In the research gap analysis section, it has been shown that many of the early lower order information theoretic mutual information estimators do under estimate MI values because they avoid computation of pdf of variables. Thus, this evaluation checks the effectiveness of the new proposed methodology for MI estimation (i.e. to check the suitability of the proposed MI estimator to adequately estimate MI values of a feature). The Second Artificial dataset (A2) was used to test the ability of IFS method to select relevant features in a controlled study (i.e., the variables that are relevant to dependent output variable are known in advanced, unlike in the natural domain). Also, copies of A2 were used to test the ability of the proposed IFS method to handle redundancy and irrelevant features and the experimental results are presented in Section 6.1. Finally, natural domain datasets were used with Machine Learning Algorithms

(i.e., since the relevant features are not known in advanced), in Section 6.2 the effectiveness of the proposed IFS method was tested by comparing the performance evaluation of the machine learning models trained before and after feature selection. While in Section 6.3, the relative performance of the IFS Method is tested along with five feature selection benchmarks.

6.1 Evaluation of IFS Method with Artificial Datasets

This section has two objectives. The first objective is to evaluate the MI estimator (IFS Selection Criterion Function) formulated as Equation (4.21) in terms of its ability to estimate MI value. And the second objective is to test the ability of IFS Method to select relevant features and eliminate redundancy between features as presented in Section 4.5 and Subsection 4.5.2.

6.1.1 Evaluation of IFS Selection Criterion Function (IFS_MI estimator)

In order to evaluate the IFS criterion function (IFS_MI estimator), artificial dataset A1 which contains six synthetic features each from different functions as described in Section 5.1 was used. IFS_MI estimator was compared with some of the existing MI estimator benchmarks namely; the k -nearest neighbor (kNN) estimator (Kraskov et al., 2004), the generalized nearest neighbor graphs (GNN) based on Renyi entropy estimator (Da'vid et al., 2010), MI estimator of minimum-redundancy maximum-relevancy (mRMR) (Peng et al., 2005), Adaptive partitioning MI (MI_AP) estimator from information theoretical estimators (ITE) toolbox (Szabó, 2015), and entropy estimator based on empirical probability distribution (EMP) from information theoretic measures (Meyer, 2014).

Figure 6.1 is the comparison between proposed IFS_MI estimator and some MI estimator benchmarks using different functional relationships as explained in the previous chapter. In all

relationships, the IFS_MI estimator performed competitively high with most known MI estimator benchmarks like kNN, GNN and MI_AP implementation of ITE. MI estimator for mRMR and EMP methods underestimated MI in all functional relationships synthesized in this study. Also, KNN and MI_AP underestimated MI with the exponential, sine and cosine functional relationships.

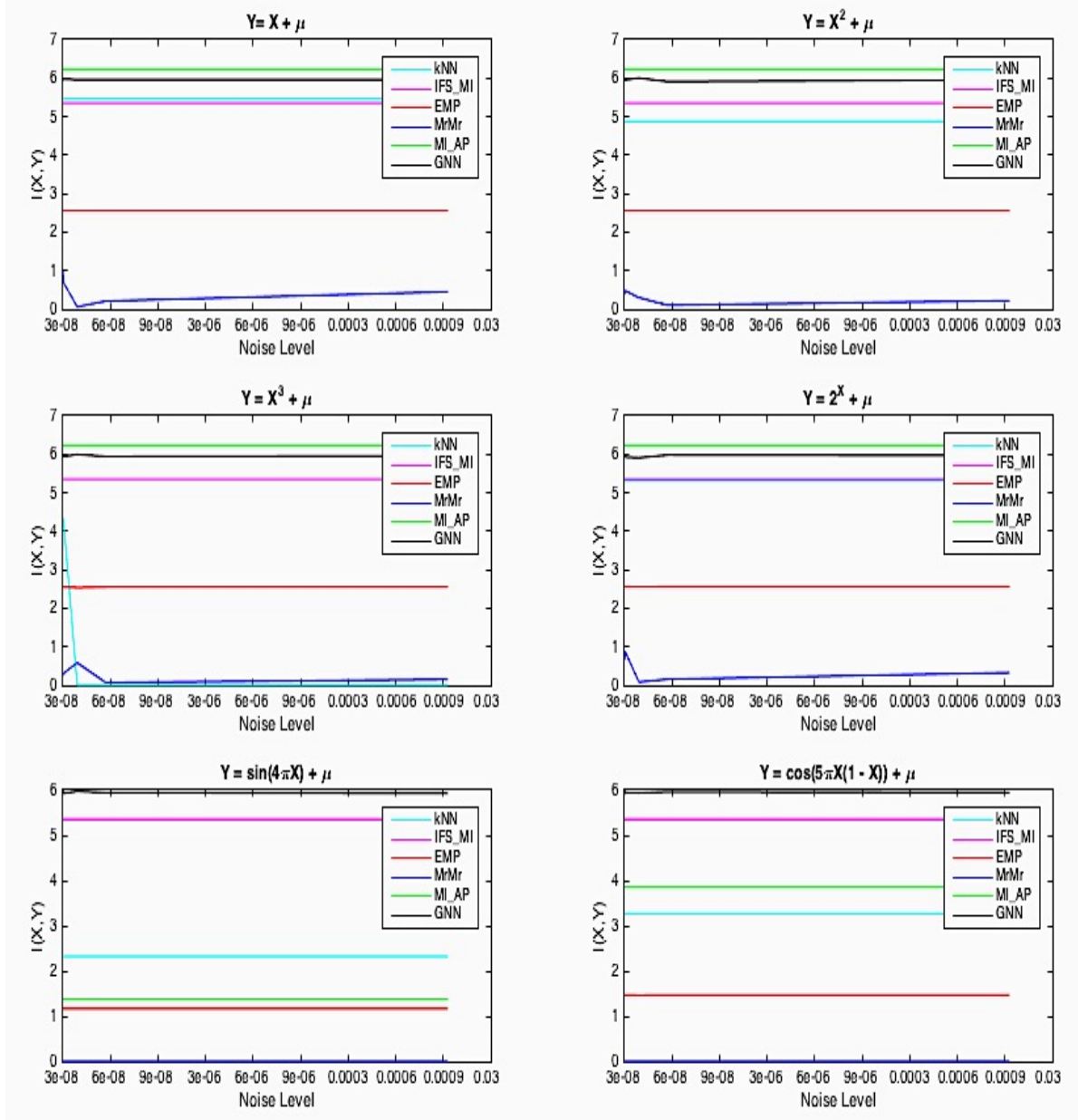


Figure 6.1: Comparison between IFS_MI Estimator and some of the Benchmarks using Ranges of Functional Relationships

6.1.2 Evaluation of IFS Method for Feature Selection using Artificial Dataset

The objective here is to evaluate IFS method for feature selection in a controlled study. That is relevant features are known in advanced and required that the method selects those features that are relevant to the predicting target output. Also, to evaluate the ability of IFS method to handle redundancy between features.

6.1.2.1 Irrelevant Features

In this study, Artificial dataset A2 was used (Section 5.1 provides detail description on how A2 was synthesized) to evaluate the proposed IFS method in a control study. A total of 10 features were uniformly generated from an interval of $[0,1]$ and based on the functional relationship, the first five features (X_1, X_2, X_3, X_4 , and X_5) were involved in the generation of the output variable Y . And at the same time, the output variable is independent of the remaining last five features (X_6, X_7, X_8, X_9 , and X_{10}). Table 6.1 presents the result obtained after running IFS method 100 times with the Artificial dataset A2. The result contains the percentage of occurrences of each feature in a feature subset for the total runs.

Table 6.1: % Appearance of each Feature in Selected Feature Subsets for 100 Runs

Feature Label	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10
% of Appearance	2	4	85	9	0	0	0	0	0	0

The result in Table 6.1 is an illustration on how the proposed IFS method is capable of discriminating between relevant features which output variable Y depends on and the irrelevant or weak (independent) features. This is one of the several results of running the proposed IFS method with A2 presented here for convenience. But, in generally, most of the times X_3 and X_4 are always selected and whereas $X_6, X_7, \dots, X_{10}$ are rarely selected.

6.1.2.2 Redundant Features

In order to test the ability of IFS method to handle redundancy in feature set, a copy of artificial dataset A2 was modified several times. First of all, since X_3 and X_4 have shown much strong mutual dependency with Y , an exact copy of X_3 and X_4 was made to replace X_9 and X_{10} respectively. This made X_9 and X_{10} 100% redundant to X_3 and X_4 respectively. After running the proposed IFS several times with redundancy check off and with redundancy check on, all of the time either X_3 with its redundant copy X_9 or X_4 with its redundant copy X_{10} or both, along with other features are selected when the redundancy check is off and only X_3 or X_4 or either of their redundant copy, along with other features are selected when the redundancy check is on. In a nutshell, redundancy is always minimized when the redundancy check is on. This study confirmed that irrespective of the location of the redundant values, when two variables are redundant (strongly correlated) to each other they tend to influence the feature selection method. Section 3.10.1 presents a review on relevant, irrelevant and redundant features.

Another relevant question that came up in the course of this evaluation was how much of a percentage of redundancy between two variables should be considered significant and can be able to affect the feature selection process? In a similar test, a copy of A2 was made to contain 80%, 60%, 50%, 30%, 20% and 10% of a random copy of X_3 and X_4 to replace a copy of the same percentage in the same location of X_9 and X_{10} respectively. All random copies datasets (with the exception of 10% random copy) were seen to select at least one of the both original and redundant copy of features along with other features when the redundancy check is off. This study further confirmed that based on the functional relationship that was used to generate A2 in this thesis, up to 20% random redundant values between two variables is significant enough to influence feature selection method when the redundancy in feature is not

handled. Results of the evaluation of the proposed IFS method's ability to handle redundancy in feature set is presented in Figure 6.2.

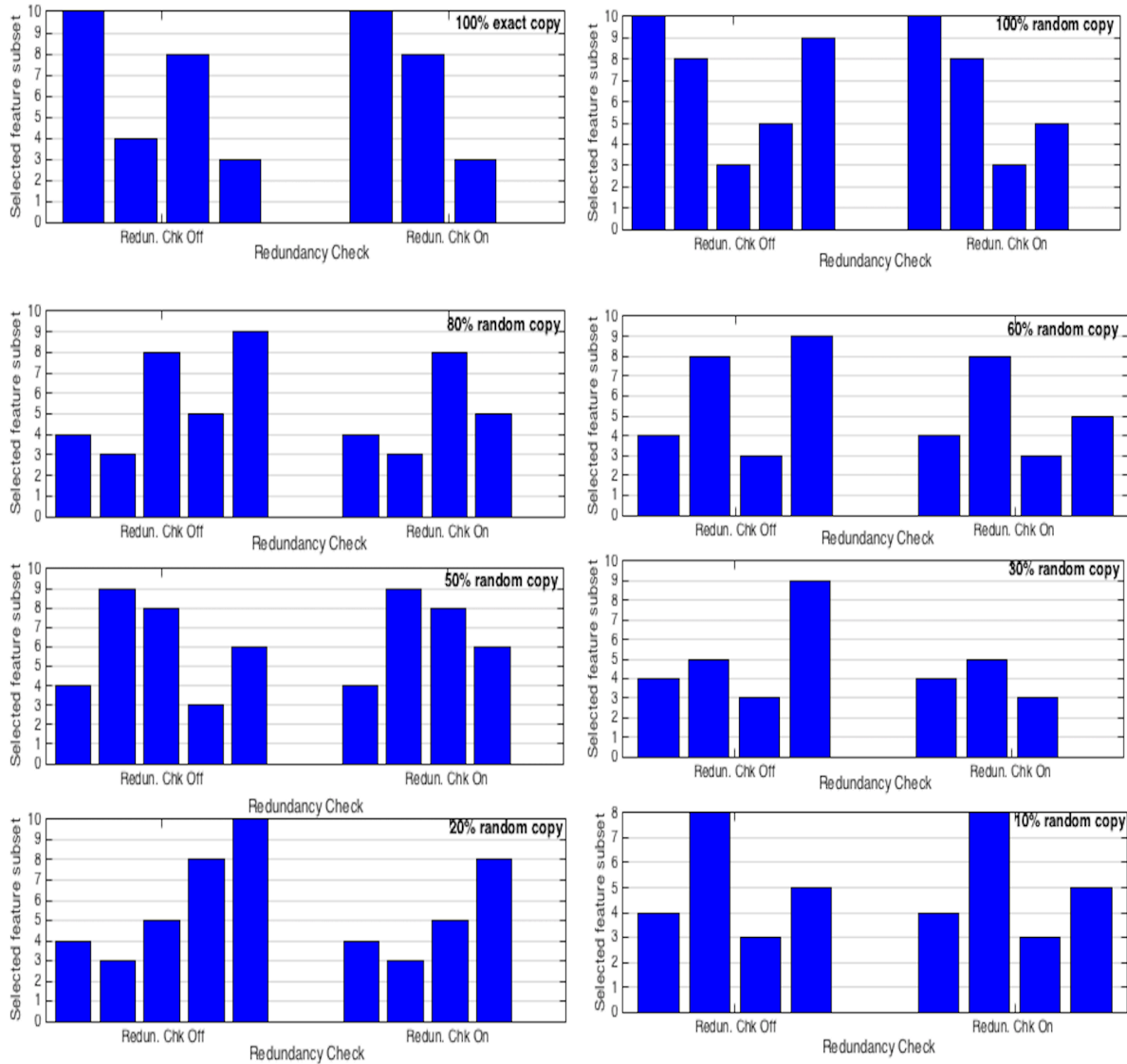


Figure 6.2: Result of IFS Evaluation of Redundancy in Features

For instance, the result of 100% exact copy of X_3 and X_4 replacing X_9 and X_{10} respectively and that of the 100% random copy of X_3 and X_4 replacing X_9 and X_{10} respectively as shown in Figure 6.2 is explained as follow; In the case of 100% exact copy of X_3 and X_4 replacing X_9 and X_{10} respectively, with the redundancy check off, the selected feature subset in sequence is $\{X_{10}, X_4, X_8, X_3\}$. However, with the redundancy check on the feature subset selected in

sequence becomes $\{X_{10}, X_8, X_3\}$. Because of the selection of X_{10} , its exact copy X_4 was removed during the redundancy check. Also, in the case of 100% random copy of X_3 and X_4 replacing X_9 and X_{10} respectively, with redundancy check off the selected feature subset is $\{X_{10}, X_8, X_3, X_5, X_9\}$. However, when the redundancy check is on, the feature subset selected has reduced to $\{X_{10}, X_8, X_3, X_5\}$. Because of the selection of X_3 , its random copy X_9 was removed during the redundancy check.

6.2 Evaluation of IFS Method with Natural Domains

This section described the evaluation of the proposed IFS method using datasets from natural domains. In the natural domain settings, relevant features for a particular dataset are not known in advance. So, the standard way to evaluate the effectiveness of a feature selection is through training and testing of machine learning algorithms using dataset first without the use of feature selection algorithm to select feature subset from the available feature space, and secondly using the same dataset with feature selection to select feature subset from the available feature space. Finally, compare the results of the two set of experiments based on the performance of the machine learning algorithms. More so, the alternative way to evaluate the effectiveness of a feature selection algorithm is to compare its relative performance with some of the existing feature selection methods. The alternative methodology is presented in Section 6.3.

In this first study, a total of fifteen experiments were made for the two supervised learning tasks namely, regression and classification using ten natural datasets obtained from UCI Repository. The detail description of natural domain datasets is in Section 5.2.

6.2.1 Proposed IFS Evaluation for Regression Tasks

Table 6.2 presents the result of IFS method evaluation using the natural domain, before and after feature selection is applied to the natural domain regression datasets. The IFS method has significantly reduced the original feature sets to fewer number of features in a subset, particularly in Breast cancer, Forest fire and Parkinson datasets. In energy utilization dataset with two separate output variables namely energy efficiency cooling loads and heating load, in both cases, the number of features in the selected feature subset is 4. However, that was not the case in Parkinson disease dataset, which also has two separate output variables namely motor_UPDRS (clinician's motor UPDRS score) and total_UPDRS (clinician's total UPDRS score). Twelves features were selected with motor_UPDRS as the output variable, and four features were selected in the case of total_UPDRS.

Table 6.2: Feature Set Before and After IFS Feature Selection Method is Applied for Regression Datasets

Dataset	No of features before feature selection	No. of the feature after feature selection is applied.	Size of Partition	w (in %)
Breast cancer	33	7	2	3
Communities and Crime	99	85	5	1
Concrete	8	3	5	3
Energy (cooling load)	8	4	5	3
Energy (heating load)	8	4	5	3
Forest fire	12	4	5	20
Parkinson (motor updrs)	19	12	5	20
Parkinson (total updrs)	19	4	5	20

IFS method failed to significantly reduces features in Communities dataset, the reason for this could be due to large number of features in the dataset and the efficiency of the proposed MI

estimation method to handle feature selection for higher dimensional dataset cannot be guarantee. This limitation of the Proposed methodology is stated in Section 1.5 and the reasons explained in Subsection 3.10.1.1.

However, in order to evaluate the relevant of the feature selection, two separate models for each dataset were simulated before and after IFS is applied using three different machine learning algorithms namely Artificial Neural Networks (ANN), Multi-Regression (MultiReg.) model and Support Vector Regression (SVR) in the case of Regression tasks. Table 6.3 presents machine learning models performance evaluation using testing dataset for regression tasks. Table 6.3 is a performance evaluation of the three trained learning models for the regression tasks based on the selected feature subsets. Three performance metrics were used in this study namely Mean Absolute Deviation (MAD), Normalized Mean Squared Error (NMSE) and Root Mean Squared Error (RMSE) metrics. The most frequent used performance evaluation metrics are MAE (= MAD) and MSE (Benoît et al., 2013).

However, while some authors in the literature argued that MAE should be the natural measure of average error instead of RMSE (Willmot & Matsuura, 2005), others believed that the RMSE is more appropriate to represent model performance than the MAE, particularly when the error distribution is expected to be normal distribution (Chai & Draxler, 2014). But what is more important is that the best statistics metrics should provide a representation of error distribution in addition to a performance measure (Chai & Draxler, 2014).

In this regards combination of metrics are required to assess model performance, thus, the three metrics MAD, NMSE, and RMSE were used in this thesis. Since the objective of regression prediction tasks is to reduce as much as possible error criterion, the best performing model is the one with consistent low error values across the metrics.

Table 6.3: The Machine Learning Models Performance Evaluation for Regression Tasks

Name of the dataset	Method type	MAD test score	NMSE test score	RMSE test score
Breast cancer	ANN	0.0148	0.9876	0.9151
	MultiReg.	0.0083	0.9028	0.8373
	SVR	0.0092	0.9602	0.9126
Community	ANN	0.1623	1.3744	0.3000
	MultiReg.	0.00085	0.6140	0.1511
	SVR	0.17623	1.1108	0.2641
Concrete	ANN	0.0068	0.9976	0.9376
	MultiReg.	0.0026	0.6889	0.6997
	SVR	0.0111	0.7315	0.7100
Energy Efficiency (cooling loads)	ANN	0.0625	0.9989	0.9601
	MultiReg.	0.0182	0.4552	0.4458
	SVR	0.0182	0.2832	0.2781
Energy Efficiency (heating loads)	ANN	0.0041	0.9968	0.9431
	MultiReg.	0.0082	0.405	0.4063
	SVR	0.0292	0.2744	0.2753
Forest Fires	ANN	0.00982	0.9952	0.1540
	MultiReg.	0.0039	0.9970	0.5987
	SVR	0.0050	1.0559	1.0405
Parkinsons (UPDRS)	ANN	0.0108	0.9995	0.9882
	MultiReg.	0.00015	0.8936	0.8743
	SVR	0.0520	0.6219	0.6123
Parkinsons (Total UPDRS)	ANN	0.0097	0.9995	0.9731
	MultiReg.	0.0006	0.908	0.8867
	SVR	0.0154	0.6856	0.6910

From Table 6.3, based on the datasets used it appears that the selected feature subsets have performed competitively with Multi-Regression learning model and support vector regression. Nevertheless, Table 6.4 compares the performance evaluation of the complete feature set (before feature selection) and that of the selected feature subset (after feature selection) using Multi-Regression model. Column two (No. of the feature in a feature set) of the table, presents the number of the feature before feature selection / the number of the feature after

feature subset selection. Also, all metrics results before a forward slash (/) represent predictive evaluation of the Multi-Regression learning algorithm with a dataset before feature selection is applied and the metric results after a forward slash (/) represents predictive evaluation of the Multi-Regression learning algorithm with a dataset after feature subset is selected from the feature space using IFS feature selection algorithm.

Table 6.4: Performance Evaluation between Features before IFS Method and Features after IFS Method is Applied for Regression Task

Dataset	No. of Feature in a feature set	MAD test score	NMSE test score	RMSE test score
Breast cancer	33 / 7	0.02020 / 0.00830	0.84000 / 0.90280	0.86112 / 0.83727
Community	99 / 85	0.00086 / 0.00085	0.68470 / 0.61398	0.16470 / 0.15112
Concrete	08 / 03	0.00697 / 0.00260	0.62940 / 0.68890	0.66330 / 0.69969
Energy Efficiency (cooling loads)	08 / 04	0.031412 / 0.01823	0.34140 / 0.45520	0.34761 / 0.44578
Energy Efficiency (heating loads)	08 / 04	0.01205 / 0.00825	0.28760 / 0.40500	0.30130 / 0.40625
Forest Fires	12 / 04	0.19209 / 0.00392	0.99656 / 0.99700	1.74080 / 0.59867
Parkinson (motor UPDRS)	19 / 12	0.01243 / 0.00015	0.82400 / 0.89360	0.81612 / 0.87428
Parkinson (total UPDRS)	19 / 04	0.03506 / 0.00060	0.84320 / 0.90800	0.86014 / 0.88670

Looking at the performance metric it can be seen that there is not a much significant difference between the models based on the datasets before feature selection is applied and those models that are based on selected feature subsets after applying feature selection. However, it is good to say that the use of feature selection with some datasets has improved the predictive performance while other datasets do not. In addition, having competitive models with fewer feature subset is an achievement of some of the objectives of the feature selection methods. Specifically, in terms of having a concise model which are believed to

avoid overfitting and generalized better. Also, models built based on selected feature subsets instead of very large feature set reduces the burden of heavy costs involves in the collection or generation of some features in some of the application areas which could have been irrelevant to the predicting output property.

6.2.2 Proposed IFS Evaluation for Classification Tasks

Similar to Table 6.2, Table 6.5 presents the result of IFS method evaluation using the natural domain, before and after feature selection is applied to the natural domain classification datasets. The IFS method has significantly reduced the original feature sets to fewer number of features in a subset, particularly in Breast cancer, Glass and Wine datasets.

Table 6.5: Feature Set Before and After IFS Feature Selection Method is Applied for Classification Datasets

Dataset	No of features before feature selection	No. of the feature after feature selection is applied	Size of Partition	w (in %)
Breast cancer	33	7	2	3
Mammographic mass data	5	2	2	3
Glass	9	4	3	3
Iris	4	3	3	3
Wine	13	6	3	3
Energy (cooling load)	8	4	5	3
Energy (heating load)	8	4	5	3

However, in order to evaluate the relevant of the feature selection, two separate models for each dataset were simulated before and after IFS is applied using three different machine learning algorithms namely Artificial Neural Networks (ANN) and Support Vector Machine (SVM). Table 6.6 presents comparison between the performance of the two machine learning

algorithms (ANN and SVM) with complete feature set (before feature selection is applied) and with the selected feature subset (after IFS feature selection is applied). For each dataset in Table 6.6, the evaluation results at the top row is for ANN classifier and the results below is for the SVM classifiers. Also, in each case, results before the forward slash (/) represents the results obtained when the feature set is used (before IFS feature selection is applied to the dataset) and the results after the forward slash (/) is the results obtained when the selected feature subset (after IFS feature selection is applied to the dataset).

The results obtained in Table 6.6 has indicated superior performance of the models that are based on selected feature subsets. With the exception of the last two datasets were the evaluation results based on the full feature set model performed better with both classifiers, all other results have shown that the selected feature subset model has superior performance with both classifiers. More so, the same benefits observed in Table 6.4 for regression tasks learning models, is seen in the case of classification tasks models in Table 6.6. That is, achieving superior performance with little or few features is better in terms of building a concise model that avoid overfitting and generalized well, also it will reduce the cost of data collection for large features with little or irrelevant benefit to the output variable. This is in addition to reducing the complexities in terms of a model training time.

Table 6.6: Performance Evaluation between Features Before IFS Method and Features After IFS Method is Applied for Classification Task

Dataset	No of feature in feature set	Method type	MAD test score	NMSE test score	RMSE test score
Breast cancer	33 / 10	ANN	0.025 / 0	1.29452 / 0.98897	0.5244 / 0.44721
		SVM	0.0167 / 0.0167	0.9322 / 0.8421	0.38730 / 0.28868
Glass	09 / 04	ANN	0.4186 / 0.2558	0.65488 / 0.5927	1.43056 / 1.1513
		SVM	0.328126 / 0.17188	0.51405 / 0.43248	1.05327 / 1.02317
Iris	04 / 03	ANN	0 / 0	0 / 0	0 / 0
		SVM	0.044 / 0	0.24270 / 0	0.2108 / 0
Breast tumor	05 / 02	ANN	0.03448 / 0.0172	0.7403 / 0.46089	0.37139 / 0.22740
		SVM	0.023 / 0.01149	0.80980 / 0.73314	0.40115 / 0.35558
Wine	13 / 06	ANN	0 / 0	0 / 0	0 / 0
		SVM	0.0185 / 0.0185	0.30574 / 0.29428	0.2357 / 0.2357
Energy efficiency (cooling load)	08 / 04	ANN	0.11688 / 0.04545	0.3875 / 0.4672	0.42640 / 0.52841
		SVM	0.00433 / 0.04546	0.36747 / 0.41993	0.41089 / 0.48349
Energy efficiency (heating load)	08 / 04	ANN	0.10390 / 0.16234	0.404189 / 0.5151	0.44137 / 0.5976
		SVM	0.01732 / 0.00433	0.3341 / 0.42265	0.37219 / 0.46987

6.3 Comparing Proposed IFS Method with Some Other Feature Selection Methods

This Section, just like Section 6.2 provides evaluation of IFS algorithm using some selected natural domain datasets. But unlike Section 6.2, here the study evaluates the performance of IFS method relative to the performance of some feature selection method benchmarks obtained from Arizona State University (ASU) Feature Selection Repository (Zheng, et al., 2010; Jundong, et al., 2016). The ASU feature selection repository is designed to collect existing and the state of the art feature selection algorithms developed in the machine learning

research community. This is with the aimed at facilitating comparison across spectrum of different existing feature selection algorithms and obtained comprehensive view on the performance of both the new and existing ones.

In this regard, this study obtained five popular and filter based feature selections algorithms from ASU feature selection repository namely; Kruskal Wallis, Information Gain, Fast Correlation Based Filter (FCBF), Correlation Based Feature Selection (CFS) and Minimum-Redundancy-Maximum-Relevance (mRmR) feature selection algorithms. These five features selection represents generally some of the well-known feature selection methods and brief description of the selected five benchmark feature selection methods is in Section 3.8. For IFS method, level of uncertainty for dataset ($w = 3$) and size of the partition ($P = 5$) are the default settings.

6.3.1 Experimental Setups for Evaluation of IFS Method with Five Other Feature Selection Methods

As earlier on introduced, the five feature selection methods were obtained from ASU feature selection repository as a package. The package was installed according the provided installation and environment setup see Zheng, et al. (2010) for details. However, in order to have a fair evaluation only classification datasets previously used in this experiments were used. This is because the available feature selection methods obtained from ASU feature selection repository are designed for classification tasks only. The dataset includes Iris, Breast cancer, Mammographic mass data (Breast tumor), Glass, Wine, Energy utilization dataset (cooling load) and Energy utilization dataset (heating load). Refers to the Section 5.1 for the detail description of the datasets. Support Vector Machine (SVM) classifier was used for performance evaluation of the feature selection methods after each feature subset are selected. With the exception of mRMR method which requires k -parameter (indicating $k - top$

features) to be set empirically, all the remaining feature selections including IFS method, were run using default settings.

A common script was prepared to read the preprocessed dataset file and indicate feature selection method to run. Subsequently, it uses the new feature subset selected to extract a new dataset for training with SVM machine learning algorithm and finally uses the generated SVM classifier with the test dataset to run performance evaluation.

6.3.2 Comparison of IFS Method with Five Other Feature Selection Method

Table 6.7 presents a comparison of the number of features in feature subsets selected by each of the six feature selection algorithms. All the feature selection methods in this study were able to select significantly few number of features in feature subsets except in two to three datasets were some of the algorithms relatively select large or very few number of features. For instance, Kruskal Wallis method appears to have selected too many irrelevant features from the feature space in breast cancer dataset (32 out of 33) and Information Gain, CFS and FCBF methods seemed to have ignored some relevant features for the same dataset (3, 2, and 2 respectively out of 33). However, in all the datasets, the proposed IFS method seems to have optimally selected significant number of features from feature spaces.

Table 6.7: Comparison of the Number of Features Selected in the Feature Subsets from the Datasets by the Feature Selection Algorithms

Dataset	fsInfoGain	fsKruskal	fsMRMR	fsIFS	fsCFS	fsFCBF
iris:(4)	3	3	3	3	2	2
breast cancer:(33)	3	32	10	11	2	2
breast tumor:(5)	4	4	4	2	3	2
glass:(9)	7	8	4	5	8	5
wine:(13)	12	12	10	7	11	10
energy-cooling:(8)	6	7	6	7	2	3
energy-heating:(8)	7	7	6	7	2	3

In terms of classification accuracy, the proposed IFS method has competitively performed very well when compare with other feature selection methods using SVM classifier. This is shown in the results presented in Table 6.8, Figure 6.3, Table 6.9 and Table 6.10.

Table 6.8 show the misclassification rate in percentage for the same feature selection methods. In all the datasets under this study, Table 6.8 shows that the proposed IFS method (fsIFS) either has least misclassification rate or is among those with very low misclassification rate. Appendix D3 presents performance rating of the six feature selection methods for comparison purposes.

Table 6.8: Percentage Misclassification for the Six Feature Selection Algorithms

Dataset	fsInfoGain	fsKruskal	fsMRMR	fsIFS	fsCFS	fsFCBF
iris	0	0	3.33	0	3.33	3.33
breast cancer	17.50	25.00	22.50	7.50	20.00	15.00
breast tumor	13.79	13.79	15.52	13.79	13.79	13.79
glass	37.21	39.53	32.56	32.56	37.21	27.91
wine	0	0	2.78	2.78	0	0
energy-cooling	20.78	19.48	22.73	18.83	37.66	20.13
energy-heating	13.64	13.64	12.99	11.04	33.12	9.74

Figure 6.3 is the classification accuracy for the same feature selection algorithms as presented in Table 6.7 using the same datasets.



Figure 6.3: Classification Accuracy of the Six Feature Selection Methods Using: (A) Iris, (B) Breast Cancer, (C) Breast Tumor, (D) Glass, (E) Wine, (F) Energy Utilization (cooling load) and (G) Energy Utilization (heating load) Datasets

SVM was used for training the machine learning model using selected features by each feature selection algorithms from each dataset under this study. Figure 6.3 (A) to Figure 6.3 (G) is the classification accuracy of six feature selection algorithms with Breast cancer, Iris, Breast tumor, Glass, Wine, Energy utilization (cooling load) and Energy utilization (heating load) respectively. Horizontal axis represents feature selection algorithms. The number above the name of each feature selection method represented the exact number of selected features by the feature selection method out of the total number of features in full feature set. For example, in Iris dataset, fsInfoGain method with the number (3/4) means that, fsInfoGain method selected 3 features out of a total of 4 feature in feature set.

Table 6.9 presents the Root mean squared error (RMSE) evaluation of the six feature selection methods based on the SVM classification performance with the selected feature subset selected by each of the methods. Similarly, Table 6.10 presents Normalized mean squared error (NMSE) for the same feature selection methods. As previously stated, the best performing model is expected to consistently have low error values across performance metrics. Both Table 6.9 and Table 6.10 show that IFS evaluation result is consistent for all the datasets. Out of the seven datasets used in this comparison test, IFS method has lowest error values with two datasets (Breast cancer and Energy utilization-cooling loads), jointly lowest error values with another two datasets (Iris and Breast tumor) and jointly relatively worst in only one dataset (Wine). Appendix D3 presents performance rating of the six feature selection methods for comparison purposes.

Table 6.9: Performance Comparison between IFS Method and Other Feature Selection Methods Based on the RMSE

Dataset	fsInfoGain	fsKruskal	fsMRMR	fsIFS	fsCFS	fsFCBF
iris	0	0	0.1826	0	0.1826	0.1826
breast cancer	0.4183	0.5	0.4743	0.2739	0.4472	0.3873
breast tumor	0.3714	0.3714	0.3940	0.3714	0.3714	0.3714
glass	1.0997	1.0455	1.2295	1.1310	1.1206	1.4942
wine	0	0	0.1667	0.1667	0	0
energy-cooling	0.4558	0.4414	0.4767	0.4340	0.6294	0.4487
energy-heating	0.3692	0.3693	0.3604	0.3322	0.6084	0.3121

Table 6.10: Performance Comparison between IFS Method and Other Feature Selection Methods Based on the NMSE

Dataset	fsInfoGain	fsKruskal	fsMRMR	fsIFS	fsCFS	fsFCBF
iris	0	0	0.2189	0	0.2064	0.2103
breast cancer	1.0871	1.1823	1.1216	0.8177	1.1040	1.0710
breast tumor	0.7475	0.7475	0.7983	0.7435	0.7475	0.7435
glass	0.6008	0.5882	0.5713	0.5348	0.4835	0.6707
wine	0	0	0.2088	0.2057	0	0
energy-cooling	0.4323	0.3932	0.4165	0.3923	0.5925	0.3921
energy-heating	0.3282	0.3381	0.3180	0.2993	0.5114	0.2700

CHAPTER 7

HYBRID FEATURE SELECTION, IMPLEMENTATION AND EVALUATION

This chapter presents hybrid feature selection for the proposed IFS method. Previous Chapters (4, 5 and 6) discussed filter based IFS method only. However, this chapter presents a hybrid filter-metaheuristic feature selection method, which combines the IFS method as filter method alongside with a meta-heuristic Ant Colony Optimization (ACO) in a wrapper like scheme. The reason for the hybrid method is to see the alternative way of finding high quality optimal solution to feature selection problem in a filter-like scheme without involving wrapper. For complete justification for the hybrid method referred to Section 3.7. However, filters generalized better due to the independency of any learning algorithm, very fast to process but generally less accurate compared to Wrappers, but accuracy of wrappers is biased by a specific machine learning algorithm embedded within the wrapper as evaluation function and incurred more computational overhead cost (see Section 1.2, Section 1.5 and Section 3.6). Like filter, metaheuristic algorithm (like ACO) does not require machine learning algorithm to solve a combinatorial optimization problem. Thus, when used to solve the feature selection problem, will produce a generic optimal feature subset that is independent to any machine learning algorithms and much faster than Wrapper. ACO guided by heuristic information and pheromone trails can further reduced the feature subset selected by the Filter method. The idea here, is to use the proposed Improved IFS method only (involving Step 1 to Step 8 of the proposed IFS method as in Table 4.1 only) without the enhancement to greedy forward procedure that gives optimal feature subset. The enhancement to the improved greedy forward procedure in the proposed IFS method is presented as Step 9 of the proposed IFS method as in Table 4.1 and its detailed algorithm is in Figure 4.3.

In this study, feature selection is presented as Multidimensional Knapsack Problem (MKP). Thus, a Binary Ant Colony System (BACS) was proposed for MKP to find a global optimal feature subset in a hybrid scheme.

Section 7.1 presents the proposed Ant Colony Optimization for MKP. The proposed hybrid feature selection algorithm is presented in Section 7.2, where IFS filter method is first used to select local optimal feature subset and BACS as a metaheuristic method is subsequently used to select global optimal feature subset. Detail experimental setups and experiments are in Section 7.3. Finally, Section 7.4 presents a comparison between IFS method as a filter based method only and IFS-BACS method as a hybrid based method.

7.1 Binary Ant Colony System (BACS) for Multidimensional Knapsack Problem (MKP)

In this Section, a meta-heuristic ant colony optimization system for solving multidimensional knapsack problem is proposed and investigated. This study is concerned with the 0-1 knapsack problem where a feature selection of an n -dimensional features is defined as a multidimensional knapsack problem. Detail review of ACO, MKP and formulation of ACO for MKP is shown in Section 3.9.

Consider a set of n items and m resources (the knapsacks) are to be selected into knapsacks of a total capacity W . Each item i has a value v_i , each resource j has a budget C_j (a knapsack capacity) and weight w_{ij} of resource j by item i . For convenience, we re-write Equation (3.18) here:

Formally, the goal is to find a subset of items to be selected into the knapsack so that:

- the objective function is to maximize value v_i :

$$\text{maximize } \sum_{i=1}^n v_i x_i \quad (7.1)$$

- subject to the constraint that the items fit into the knapsack:

$$\sum_{i=1}^n w_{ij} x_i \leq C_j \quad j = 1, 2, \dots, m \quad (7.2)$$

$$\text{and } x_i \in \{0, 1\} \quad i = 1, 2, \dots, n.$$

Since a well declared MKP assumes that $v_i > 0$ and $w_{ij} \leq C_j \leq \sum_{i=1}^n w_{ij}$ (Leguizamón & Michalevitz, 1999; Fidanova, 2007), the following settings are valid and assumed for this hybrid method: $\forall i \& j: w_{ij} = 1, C_j = 1, W = \sum_{j=1}^m C_j$ and $x_i \in \{0, 1\}$.

This setting allowed us to define MKP as Feature Selection Problem, where n items refers to the feature in the feature space or subset and total capacity of knapsacks is equal to W which indirectly determine the subset of n features to be selected to maximize the objective function $\sum_{i=1}^n v_i x_i$. This is a special case of MKP in which several knapsacks (m resources) each with similar budget (i.e. $\forall j: C_j = 1$) are to be filled simultaneously by either a feature is selected or not selected. Under this assumption we can simplify the constraint in Equation (7.2) as:

$$\text{Subject to: } \sum_{i=1}^n w_i x_i \leq W \quad (7.3)$$

where w_i is the capacity of item i and $x_i \in \{0, 1\} \quad i = 1, 2, \dots, n$.

Let $X = [x_1, x_2, \dots, x_n]$ be a solution vector in which $x_i = 0$ if an item i is not in the knapsack and $x_i = 1$ if it is in the knapsack.

This study considered multidimensional knapsack problem as consist of n -dimensional 0/1 knapsacks in which each item or object i is either taken or not is represented in a solution vector $X = [x_1, x_2, \dots, x_n]$. This way, ant populations can be seen to construct tours in a similar manner with the first ant colony framework. Detail of the algorithm is in the subsequent Subsection.

7.1.1 Solution Representation and Construction

This section describes the proposed BACS algorithm for MKP as presented in Table 7.2. BACS algorithm focuses on feature selection problem and modeled feature selection as a binary knapsack problem. However, it can equally be useful for most problems defined as a 0-1 knapsack problem. Essentially, the algorithm tends to achieve the objective function in Equation (7.1) subject to the constraints as specified in Equation (7.2).

7.1.1.1 Tour Construction by Ants

Figure 7.1 presents Ant's Tour construction as a state machine diagram, where Ants move along n -states ($s_1, s_2, \dots, s_n$) (and additional two states s_0 and s_{end} representing source and destination of the ant's tour respectively) to construct tour. That is, each Ant from the population of Ants, moves from s_1 to s_n to build a tour as a potential solution. At each state s_i , an Ant uses transition probability $P_{S_i}^k$ to make choice of the path to transit on T_{i0} or T_{i1} , where $T_{i0} = x_i = 0$ and $T_{i1} = x_i = 1$ in solution vector $X = [x_1, x_2, \dots, x_n]$.

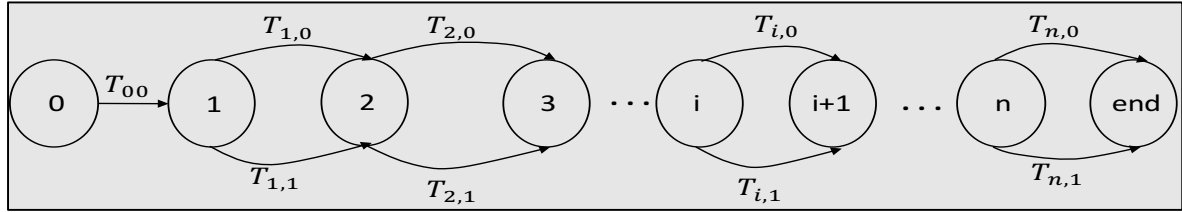


Figure 7.1: State Machine Representation of Ants Tour Construct for BACS

The transition probability $P_{S_i}^k$ is biased by pheromone trail $\tau_{s_{i,q}}$ and heuristic information $\eta_{s_{i,q}}$. Assuming that Ant k , in state i , has so far constructed the partial solution as potential solution T^p , it will then use the probability ($P_{S_i}^k$) to decide which part to transit to the next state l :

$$p_{s_i}^k(t) = \begin{cases} \frac{[\tau_{s_{i,q}}(t)]^\alpha [\eta_{s_{i,q}}(t)]^\beta}{\sum_{c_{il} \in N(T^p)} [\tau_{s_{il}}(t)]^\alpha [\eta_{s_{il}}(t)]^\beta} & \text{if } c_{il} \in N(T^p), \text{ and } q \in \{0, 1\}. \\ 0 & \text{otherwise} \end{cases} \quad (7.4)$$

$N(T^p)$ is the set of possible components; l is the states yet to be visited by Ant k . Also, α and β control relative importance of the pheromone versus the heuristic information (Dorigo et al., 2006). Subsection 7.1.1.2 discusses the computation of heuristic information and Subsection 7.1.1.4 presents pheromone update.

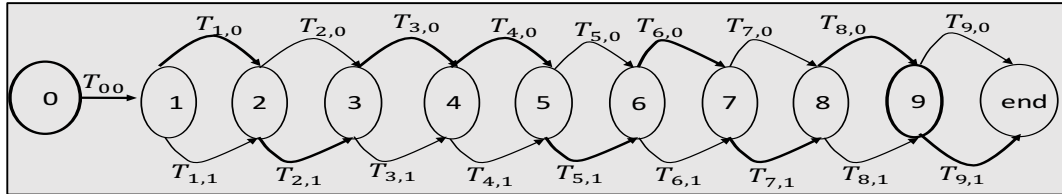


Figure 7.2: State Diagram Illustrating Tour Construction

In moving from state to state, Ant takes a path T_{i1} which means that the state s_i is selected or included into a knapsack, and path T_{i0} means that the state s_i is not selected or included into the knapsack.

For example, Figure 7.2 illustrates tour construction using state transition diagram for an Ant that travels along 9-states from the nest to the source of food. The tick curve direction lines indicate the paths taken by an Ant, and Figure 7.3 illustrates the binary representation of the tour construction by the Ant. This is then translated into potential solution vector $X = [0, 1, 0, 0, 1, 0, 1, 0, 1]$. For instance, MKP with nine items vector X indicates that items 2, 5, 7 and 9 included into the knapsacks. However, a knapsack problem needs to satisfy certain objective (i.e. *maximized profits*) as in Equation (7.1). Under the constraint that the total weight of the items in the knapsack is within the limit of the weight capacity of the knapsack as in Equation (7.2) and simplified in Equation (7.3).

S_i :	1	2	3	4	5	6	7	8	9
T_i :	0	1	0	0	1	0	1	0	1

Figure 7.3: Example Illustrating Binary Representation of a Tour Construction

Table 7.1: BACS Algorithm for MKP

Input:
1. v : an array to store Values of items
2. w : an array to store weights of items
3. n : number of distinct items
4. W : Knapsack capacity
5.
Output: best solution consists of:
6. T_b : Best tour constructed in all iterations or cycles.
7. X_b : Best solution vector in all iterations or cycles.
8. L_b : Best cost in all iterations or cycles

9. // Parameters setting: Ant's population, initial pheromone, evaporation rate,
9. // pheromone exponential weight, heuristic exponential weight, etc.
10.
11. // Ant states initialization: heuristic information, pheromone trails, Best solution,
12. // Ant's states, etc.
13. // Begins Ant cycle: move ants to construct tours
14. <i>while $t \leq \text{total number of cycles}$ do:</i>
15.
16. <i>for k from 1 to Ant's population do:</i>
17.
18. <i>for i from 1 to Ant's states do:</i>
19.
20. <i>Move Ant_k biased by pheromone trails and heuristic information</i>
21. <i>to construct Tour_k solution</i>
22.
23. <i>Compute cost for Tour_k solution and violation</i>
24.
25. <i>Check: is Tour_k has minimum cost value and the solution is feasible?</i>
26.
27. <i>Store Tour_k as Best solution if line 25 is True</i>
28.
29. Update pheromone trail by
29. Computing pheromone evaporation for Ant's state
30.
31. Update pheromone trails for Ant's states
32.
33. Store Best cost for Tour_t // t cycle.

7.1.1.2 Heuristic Information

Two models of heuristic information have been used with ACO for MKP as discussed in the literature review in Section 3.9.4; static and dynamic heuristic information. This study used

static heuristic information model, in this model the heuristic information η is set one time only at the beginning of the run to a fixed value. This study considered a profit or value v_i of an item or feature i (been the most important information for objective function) and the capacity of item or feature i (as one of the important component of the MKP constraint) in computing the heuristic information:

$$\eta = \begin{bmatrix} w_1/v_1 & \dots & w_n/v_n \\ v_1/w_1 & \dots & v_n/w_n \end{bmatrix} \quad (7.5)$$

Equation (7.5) assigned more heuristic value (visibility) to the path that correspond to the selection of an item or feature (i.e. 2nd row of η). To illustrate the heuristic information computation let assigned values to the illustration in Figure 7.2, $v = \{4, 9, 3, 6, 13, 2, 10, 4, 10\}$ and $w = \{1, 1, 1, 1, 1, 1, 1, 1, 1\}$. Thus:

$$\eta = \begin{bmatrix} 1/4 & 1/9 \dots & 1/10 \\ 4/1 & 9/1 \dots & 10/1 \end{bmatrix} = \begin{bmatrix} 0.25 & 0.11 & \dots & 0.10 \\ 4 & 9 & \dots & 10 \end{bmatrix}.$$

7.1.1.3 Cost and Violation Computation

Line 16 of Table 7.1 allows an Ant to construct a tour solution as a potential solution as described in Subsection 7.1.1.1. However, to ensure that the objective function of MKP is maximized and the solution is feasible, line 23 to line 25 in Table 7.1 compute cost and violation for the constructed tour solution by Ant k . Violation is used to ensure that the MKP constraints is enforced and the cost computation ensure that a global optimal solution is realized. The possible solution vector with minimum cost is the best solution.

In this regard, the computation of cost and violation are as follows:

$$\text{Violation} = \frac{\sum_{i=1}^n w_i \cdot x_i}{W} : x_i = 1 \quad (7.6)$$

$$\text{Cost} = \sum_{i=1}^n v_i \cdot (1 - x_i) : x_i = 0 \quad (7.7)$$

Essentially, Equation (7.6) computes the summation of w_i for those i items that are included or selected into the knapsacks and divided it by the total capacity of the knapsacks W . The solution vector X generated is feasible if the value of violation is between 0 and 1 ($0 < \text{violation} \leq 1$). On the other hand, Equation (7.7) computes the sum of the values of the i items not selected into the knapsacks as cost for the tour construct. Therefore, the feasible solution vector with the least cost value after the maximum cycles or iteration t is the best solution for the MKP instance. To compute violation and cost for the same illustration in Figure 7.2. Let $W = 4$, $XX = [0, 1, 0, 0, 1, 0, 1, 0, 1]$, $v = \{4, 9, 3, 6, 13, 2, 10, 4, 10\}$ and $w = \{1, 1, 1, 1, 1, 1, 1, 1, 1\}$.

$$\text{Violation} = \frac{(w_2 \times x_2 + w_5 \times x_5 + w_7 \times x_7 + w_9 \times x_9)}{W} = \frac{(1+1+1+1)}{4} = 1$$

$$\text{Cost} = (v_1 \times 1 + v_3 \times 1 + v_4 \times 1 + v_6 \times 1 + v_8 \times 1) = (4 + 3 + 6 + 2 + 4) = 19$$

This illustrate the case of a single ant constructing a tour and computation of violation and cost for the constructed tour. In the actual implementation of BACS for MKP several ants construct tour and each tour that is constructed a violation and cost are computed. Unlike Leguizamon and Michalevitz (1999), Fidanova (2003) and Alaya et al. (2004) that use constraint information to guide ants to find a feasible solution which incurred additional overhead cost for tour construct, BACS for MKP is designed to allow free tour construct by ant which may give an infeasible solution and only check for violation of the MKP constraint after the tour construct is completed. More so, BACS for MKP differed from BAC by Kong et al. (2008) that relaxes MKP constraint to allow infeasible solutions and applied repair operation to fix infeasible solution if possible. BACS for MKP simply ignore any infeasible

solution since it is logical to see that BACS does not require any additional overhead cost to either guide the ant to a feasible solution or relaxes MKP constraint and applying costly repair operation to repair any infeasible solutions. Empirical results have indicated that a well-designed ACO for MKP gives best optimal solution in less than 500 iterations or cycles (Fidanova, 2007; Lee & Bau, 2012).

7.1.1.4 Pheromone Trails Update

Pheromone trails are modified by pheromone update process line 29 to line 31. Whenever Ants deposit pheromone on the paths or connection they use, this action to increase the trails value on the path. However, the pheromone evaporation helps to reduce or decrease the trails value. Both pheromone deposit and evaporation are essential towards reaching an optimal solution.

When there is a deposit of a new pheromone on a connection or a chosen path, it increases the probability of those components/connections/paths that were either used by many Ants or by at least one Ant to produce a good solution. So is likely that those paths will be used again and again by Ants in future. On the other hand, pheromone evaporation contributes a sense of forgetfulness by avoiding rapid convergence of the Ants toward a suboptimal solution. Thus, favoring exploration of the new region of search space.

In this study, pheromone update follows the Ant system (AS) pheromone update with some little modification. First, pheromone values are lowered by a constant factor, after which the values of the pheromone are added on the paths taken by the ants in their tours.

Pheromone evaporation takes place globally by all population of Ants after all tours are constructed as follow:

$$\tau_{s_{i,q}}(t) \leftarrow (1 - \rho) \cdot \tau_{s_{i,q}}(t) \quad (7.8)$$

Where ρ is the evaporation rate, and its value is in the range of $0 < \rho \leq 1$. Evaporation controls the unlimited accumulation of the pheromone trails, and this is useful in making the algorithm forget any bad decision previously taken. Once evaporation takes place, All Ants deposit pheromone on the paths taken by them in the course of constructing their tours.

Pheromone update also takes place globally by all population of ants after all tours are built as follow:

$$\tau_{s_{i,q}}(t) \leftarrow \tau_{s_{i,q}}(t) + \sum_{k=1}^m \Delta\tau_{s_{i,q}}^k, \quad q \in \{0,1\} \quad (7.9)$$

Where m is the number of Ants, $\Delta\tau_{s_{i,q}}^k$ is the quantity of pheromone laid on the path (i, q) by ant k :

$$\Delta\tau_{s_{i,q}}^k = \begin{cases} Q/L_k & \text{if ant } k \text{ used path } (i, q) \text{ in its tour,} \\ 0 & \text{otherwise,} \end{cases} \quad (7.10)$$

Where Q is a constant, and L_k is the cost of tour constructed by ant k as expressed in Equation (7.7). That means the quantity of pheromone laid on the path (i, j) by ant k depends on how good is the tour solution. Moreover, the pheromone value is in the range of $0 < Q/L_k \leq Q$.

Finally, line 33 of the proposed BACS for MKP store the best cost for cycle t as in Table 7.1 and continue to next cycle or iteration until the maximum number of cycles or iterations attained.

7.2 Hybrid Feature Selection Method

This section presents a detailed design of the hybrid filter-metaheuristic method for feature selection. Section 3.7 provides details background of the hybrid method.

This study presents a hybrid filter-metaheuristic method in a wrapper-like scheme. The proposed IFS method, which is a filter based, presented and discussed previously in Chapter 4 and Chapter 6, is combined with the BACS (details in Section 7.1) method. BACS method is an Ant colony optimization solution for a multidimensional knapsack problem.

Figure 7.4 presents the flowchart of IFS_BACS hybrid feature selection method. This hybrid method consists of three methods (that can be seen to be independently standalone) joined.

The first method is the filter-based proposed IFS method (detailed in Chapter 4 and 6) implemented to rapidly select feature subset from feature space without involving or including the optimal feature subset selection procedure as detailed in Section 4.5.2 of this thesis. The output from IFS method is a list of values of the selected candidate feature subset by the IFS method. Each value is a frequency of selected feature in the feature subset after running IFS multiple times. The output from proposed IFS method is the input to the BACS method. The idea is that BACS will further reduce the feature subset selected by IFS method while maximizing the relevance of the optimal subset.

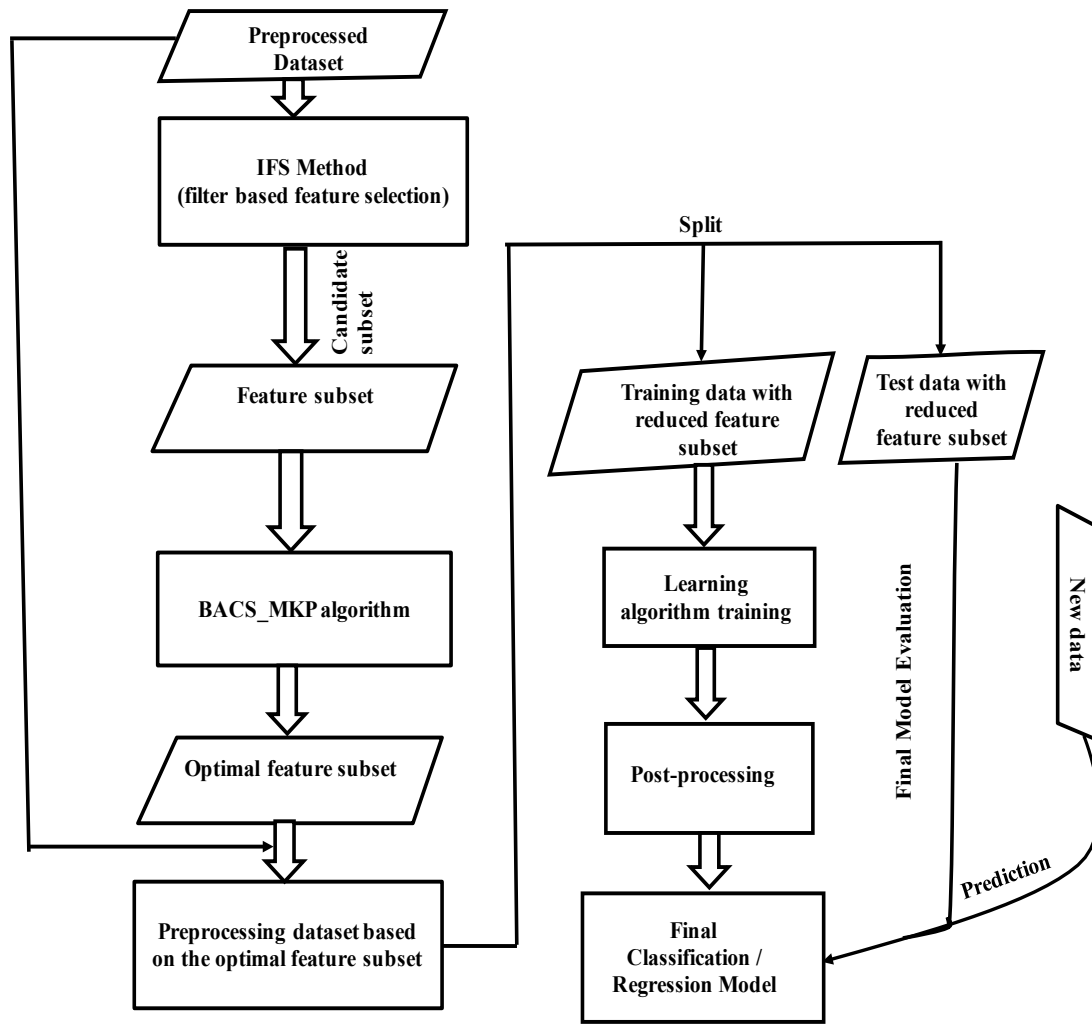


Figure 7.4: Hybrid (IFS_BACS) Feature Selection Method Flowchart

The second method is the binary ant colony optimization system (BACS) method (Section 7.1). This method received feature subset from the previous method as an input and optimized the feature subset by further reducing it the feature subset size while maximizing the relevance of the final feature subset. This BACS method is a metaheuristic solution to multidimensional knapsack problem (MKP), this means that feature selection method is defined as MKP. This hybrid method is unlike a wrapper method, in which the machine learning algorithm is used as means of evaluating the goodness of the selected feature subset, in this study BACS serves as feature subset optimizer. The output from the hybrid method is a reduced list of features selected. This optimal feature subset is then used to further

preprocessed a new dataset by extracting only the selected feature subset from the original dataset. The new dataset mapped based on optimal feature subset is split into training and test dataset. The training dataset served as an input to the next method.

The last method trains a machine learning algorithm based on the reduced feature subset dataset. This method uses cross-validation method to select optimal parameters of a machine learning algorithm and used the chosen model with the whole training dataset to finally trained a machine learning model. The performance of this final machine learning model is evaluated and reported based on the test dataset.

The Figure 7.5 (C) illustrates an optimal output from IFS_BACS method.



Figure 7.5: Illustration of the Outputs from IFS and IFS_BACS Methods: (A) *fcount* array, (B) Processed Features and their Values, and (C) Optimal Feature Subset Selected by IFS_BACS

Empirical evidence suggests that an optimal feature subset needs not to be unique as there may be more than one way to achieve accuracy using different sets of features (Kohavi & John, 1997). Figure 7.5 illustrates the complete outputs from IFS_BACS method, Figure 7.5 (A) is the output from the IFS method as explained previously in this Section. The original dataset has 33 features, but after the first stage of the IFS_BACS method, it was reduced to 31

features. *fcount* is the output of IFS method of IFS_BACS, it is the frequency or occurrence of each features after the IFS method is run multiple times; from Figure 7.4 (A) f_3 and f_5 were never selected by IFS after several cycles or runs. *fcount* serves as input to BACS method of IFS_BACS. Figure 7.4 (B) is the processed features and their occurrences extracted by BACS method during parameters settings and initialization. Based on Figure 7.4 (B), BACS setups v , w , and W (details in Section 7.3). Finally, Figure 7.4 (C) is the output from IFS_BACS hybrid method. It is the final optimal feature subset selected. From Figure 7.5, it is clear that there are many features with likely similar mutual correlation to the output variable, and could be a good replacement to those features selected in Figure 7.5 (C).

Figure 7.6 present the cost curve for BACS using three of the datasets. The Best Cost (vertical) axis represents the best cost after each cycle or iteration. After all ant from ant population construct tours, the tour with a minimum cost in a cycle is referred to as the best cost for that cycle. The solution vector with minimum cost is the best solution in the cycle or iteration. Figure 7.6 has shown that the cost is reducing as the number of iteration or cycle increases. On average the best cost is achieved in less than 300 iterations.

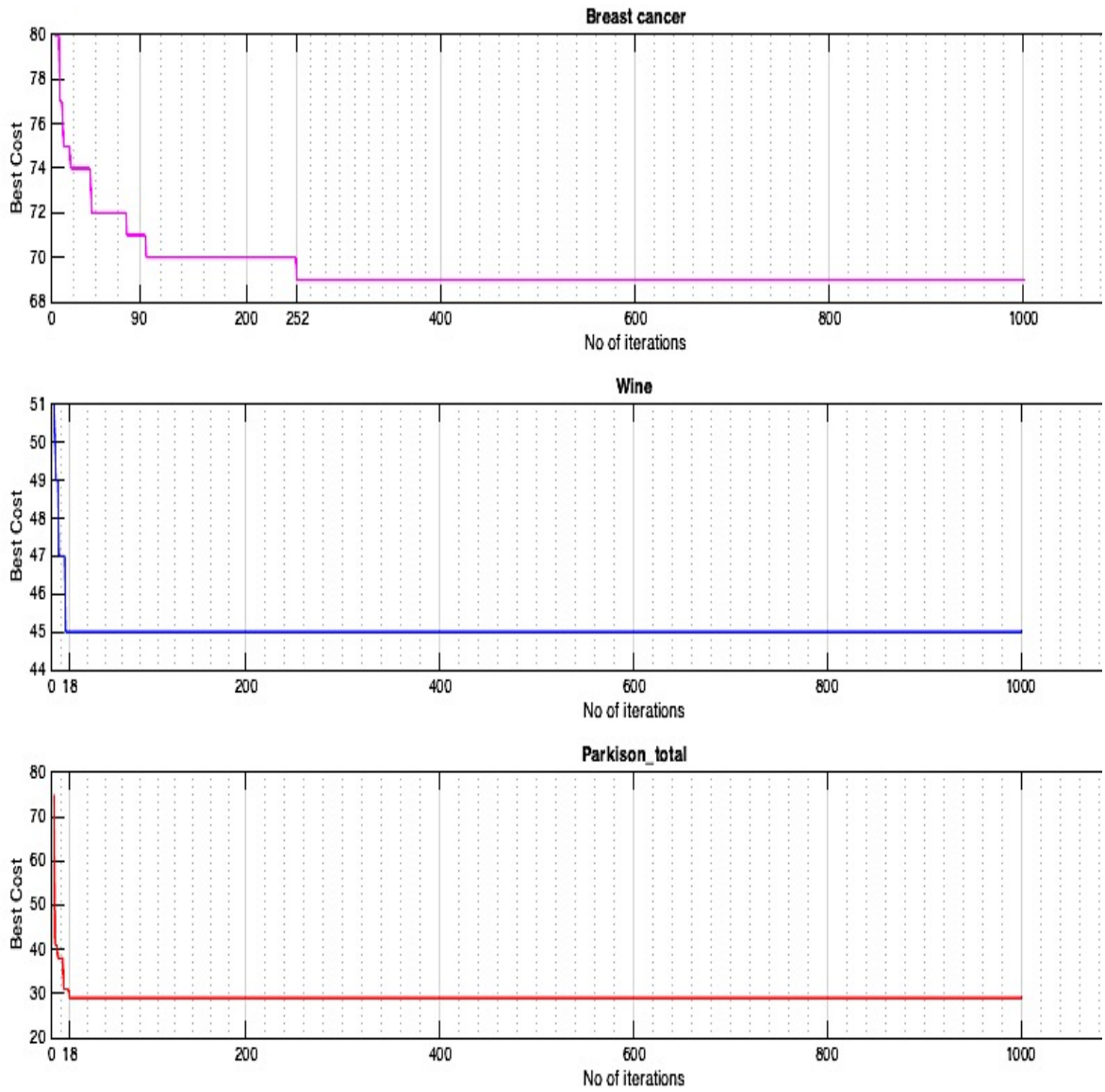


Figure 7.6: The Cost Curve for BACS

Section 7.3 described experimental details and result discussion. Also, the details comparison between the result of the proposed filter based IFS method in Chapter 4 and the results obtained based on the hybrid IFS_BACS method is presented in Section 7.4.

7.3 IFS_BACS Experimental Setups

This section presents the experimental setups, detail experiments, and results discussion. Like the previous experiments in Chapter 6, this section uses UCI Repository datasets for both

classification and regression as discussed in Chapter 5. Two separate experiments were carried out in this section using UCI datasets. The first experiment was on classification task using UCI classification dataset (See Chapter 5 for details). For this experiment support vector machine (SVM) was used as machine learning algorithm. The second experiment was on regression task using UCI regression datasets as presented in Chapter 5. Multiple linear regression (Multi-Reg.) served as the machine learning algorithm for training and evaluation. Section 5.2 provides the details of the machine learning algorithms used in this study. In both SVM and Multi-Reg. experiments, the dataset, are splits into 4:1 training and testing datasets.

In the hybrid method, the IFS method implementation consists of Step 1 to Step 8 and excluding Step 9 as presented in Table 4.1. Step 9 was part of IFS method in Chapter 4 as a method of enhancing the greedy forward algorithm on which the IFS method was developed, it does like what a metaheuristic BACS method is doing. Specifically, the method mitigates IFS method from selecting suboptimal feature subset (See Section 4.5.2 for details). In a similar way, BACS is designed to optimize further the candidate feature subset selected by IFS method which could be suboptimal by default. Two parameter settings required for IFS method namely, w (in IFS method) which is the level of uncertainty about the value of the dataset variables and the partition interval (details is in Section 5.2). The w is selected based on cross-validation method from values within the range of $[8 \times 10^{-5}, 0.5]$ and partition interval obtained from five statistical summaries namely, *minimum value*, *lower quartile*, *average*, *upper quartile* and *maximum value* in ascending order of increments. By default, IFS method is run 100 times and the number of times a feature appears is counted. Thus, an output from IFS method is an array of feature's counts (*fCount*) indicating the number of occurrence(s) of features. Figure 7.5 (A) illustrates *fCount* array where the number of

occurrences of each feature referred to as Counts and the indices of the array which represented the label of each feature called Features.

Features with weak mutual relationship (irrelevant) to the output variable of a dataset are often not selected by IFS method, hence will have zero occurrences. However, features which have strong or very strong mutual relationship with an output variable will be selected many times in the total runs. For this reason, such features will have one or more counts and are represented as a selected feature subset.

The *fCount* array serves as the input to BACS method. Figure 7.5 (B) illustrates the processing of *fCount* array to extracts *v* array and the exact features selected by the IFS method. The number of distinct items *n* is equal to the number of features selected (that is a non-zero index of *fCount*), the array of values *v* is the number of times each selected feature appears. This study believes that the more is the number of times a feature is selected shows its strong mutual relationship with the output variable. Hence, the best representation of value for a feature. As for *w* (in BACS), the array to store the weight of the items, in this study a unit weight is given to each feature (details in Section 7.1). It assumed that feature should have equal weight. Moreover, since MKP required that $\sum_{i=1}^n w_i \cdot x_i \leq W$ (Equation (7.3)), where *W* is the knapsack capacity. BACS provide two options for *W* values. One is to be supplied by user apriori, and the second option is that *W* is automatically determine based on the number of unique value from processed features as in Figure 7.4 (B). In a case where a number of unique value from processed features is equal to the number of feature subset, at least $W = \text{size of feature subset} - 1$. For example, if in Figure 7.4 (B) *W* is to be determined automatically, $W = 8$.

Other parameter settings are:

$\rho = 0.1, \tau_0 = 0.1, Q = 1, \alpha = 1, \beta = 0.02$, Max. cycle = 1000, No. of Ants = no. of states (items or features) = n and the Total Ants = Ant's population = $5 \times n$.

Both heuristic information (η) and pheromone trail matrices are $2 \times n$.

7.3.1 Presentation and Discussion of IFS_BACS Results

Table 7.2 and Table 7.3 present the results of **IFS_BACS** method of feature selection. For the classification task, the support vector machine algorithm was used to trained the selected feature subset and for the regression tasks, multi-linear regression algorithm was used for the training. In both cases, the Normalized mean squared error (NMSE) and Root mean squared error (RMSE) were used for evaluation the performance of IFS_BACS methods. Section 7.4 provides the comparison of the results between IFS method only and IFS_BACS method.

Table 7.2: Result of IFS_BACS for Classification Tasks

Dataset	No. of feature before IFS_BACS	No. of features selected based on IFS_BACS	NMSE test score	RMSE test score
Breast cancer	33	10	0.8177	0.2739
glass	9	5	0.3929	0.8760
iris	4	3	0	0
Breast tumor	5	4	0.7166	0.3474
Wine	13	5	0	0
Energy utilization (cooling)	8	7	0.3336	0.3780
Energy utilization (Heating)	8	7	0.2838	0.3223

However, in both Table 7.2 and Table 7.3, IFS_BACS method has significantly reduced the number of features from the feature set before applying the selection algorithm in most of the dataset. Moreover, the NMSE and RMSE evaluation results indicated good performance. For instance, above 70% prediction accuracy were recorded for all classification tasks, except the glass dataset which has 63% classification accuracy. Even though a similar prediction

accuracy is irrelevant in the case of regression tasks, when rounded up to a whole number, regression tasks shown closer results in some tasks with the same evaluation indices.

Table 7.3: Results of IFS_BACS for Regression Tasks

Dataset	No. of feature before IFS_BACS	No. of feature in Selected Feature subset	NMSE test score	RMSE test score
Breast cancer	33	9	0.7562	0.6907
Community	99	22	0.6000	0.1331
concrete	8	7	0.5906	0.6018
Energy utilization (cooling)	8	3	0.3611	0.4071
Energy utilization (Heating)	8	3	0.3593	0.3626
Parkinson (motor UPDRS)	19	10	0.8516	0.8566
Parkinson (total UPDRS)	19	12	0.8453	0.8521

For instance, for the community and crime dataset, IFS_BACS prediction result was 90.57% prediction accuracy after both the test data and predicted data were rounded to the nearest whole number before computing the percentage prediction accuracy. Also, in a similar way Energy utilization datasets which are both used for classification and regression tasks have prediction accuracy of 85.71% and 84.42% for energy consumption (cool loads) and 89.61% and 75.32% for energy utilization (heat loads) respectively. Moreover, these results were achieved with 3 out of 8 features for the regression tasks against 7 out of 8 features for classification tasks. The percentage prediction accuracy for classification tasks is presented in Section 7.4.

7.4 Comparison of Filter Based IFS and Hybrid Based IFS_BACS Methods

This section presents the comparison between the filter based IFS method and hybrid based IFS_BACS method. IFS method builds on the complete algorithm presented and detailed in Chapter 4. IFS_BACS method is proposed as an alternative method for IFS method as in

Chapter 4, this comparison is intended to reveal the strength and weakness of the proposed IFS method.

Table 7.4 and Table 7.5 presented the result of comparison between feature set before applying feature selection and after applying both feature selection methods. Also, in the same table, comparison between the features selected by IFS and those chosen by IFS_BACS methods can be observed. Comparing feature set before applying feature selection with the feature subset after applying IFS or IFS_BACS method, it shows that both methods have significantly selected fewer number of features from the feature set before applying the selection algorithm in most of the dataset. However, in the case of comparing the two methods for the number of selected features in feature subset, IFS method select fewer features in classification tasks whereas, in regression tasks IFS_BACS method select fewer features in some tasks compared to IFS method, and other tasks IFS method have fewer feature selected without significant difference. However, there is a significant difference between IFS and IFS_BACS methods in the case of community & crime dataset where IFS_BACS selected 22 out of 99 features and IFS method select 85.

Table 7.4: Comparison of the Number of Features in Feature Subset for IFS and IFS_BACS Methods using Classification Datasets

Dataset	No. of feature before IFS BACS	No. of feature Selected by IFS only	No. of feature selected by IFS BACS
Breast cancer	33	10	10
glass	9	4	5
iris	4	3	3
Breast tumor	5	2	4
Wine	13	6	5
Energy utlization (cooling)	8	4	7
Energy utlization (Heating)	8	4	7

Table 7.5: Comparison of the Number of Features in Feature Subsets for IFS and IFS_BACS Methods fusing Regression Datasets

Dataset	No. of feature before IFS_BACS	No. of feature Selected by IFS only	No. of feature selected by IFS_BACS
Breast cancer	33	7	9
Community	99	85	22
concrete	8	3	7
Energy utlization (cooling)	8	4	3
Energy utlization (Heating)	8	4	3
Parkinson(motor_UPDRS)	19	12	10
Parkinson(total_UPDRS)	19	4	12

Table 7.6 and Table 7.7 present the performance evaluation for both IFS and IFS_BACS method with classification and regression tasks respectively. In both performance evaluation, result shows that IFS_BACS method significantly outperformed IFS method. Figure 7.7 presents the comparison of the percentage classification accuracy between the two methods for classification tasks.

Table 7.6: Comparison of the Performance Evaluation between IFS and IFS_BACS for Classification Tasks

Dataset	NMSE test score by IFS only	NMSE test score by IFS_BACS	RMSE test score IFS only	RMSE test score by IFS_BACS
Breast cancer	0.8421	0.8177	0.2887	0.2739
glass	0.4325	0.3929	1.0232	0.8760
iris	0	0	0	0
Breast tumor	0.7331	0.7166	0.3556	0.3474
Wine	0.2943	0	0.2357	0
Energy utilization (cooling)	0.4199	0.3336	0.4835	0.3780
Energy utilization (Heating)	0.4227	0.2838	0.4699	0.3223

Table 7.7: Comparison of the Performance Evaluation between IFS and IFS_BACS Methods for Regression Tasks

Dataset	NMSE test score by IFS only	NMSE test score by IFS_BACS	RMSE test score IFS only	RMSE test score by IFS_BACS
Breast cancer	0.9028	0.7562	0.83727	0.6907
Community	0.6140	0.6000	0.15112	0.1331
concrete	0.6889	0.5906	0.699692	0.6018
Energy utilization (cooling)	0.4552	0.3611	0.44578	0.4071
Energy utilization (Heating)	0.4050	0.3593	0.40625	0.3626
Parkinson (motor_UPDRS)	0.8936	0.8516	0.87428	0.8566
Parkinson (total_UPDRS)	0.9080	0.8453	0.8867	0.8521

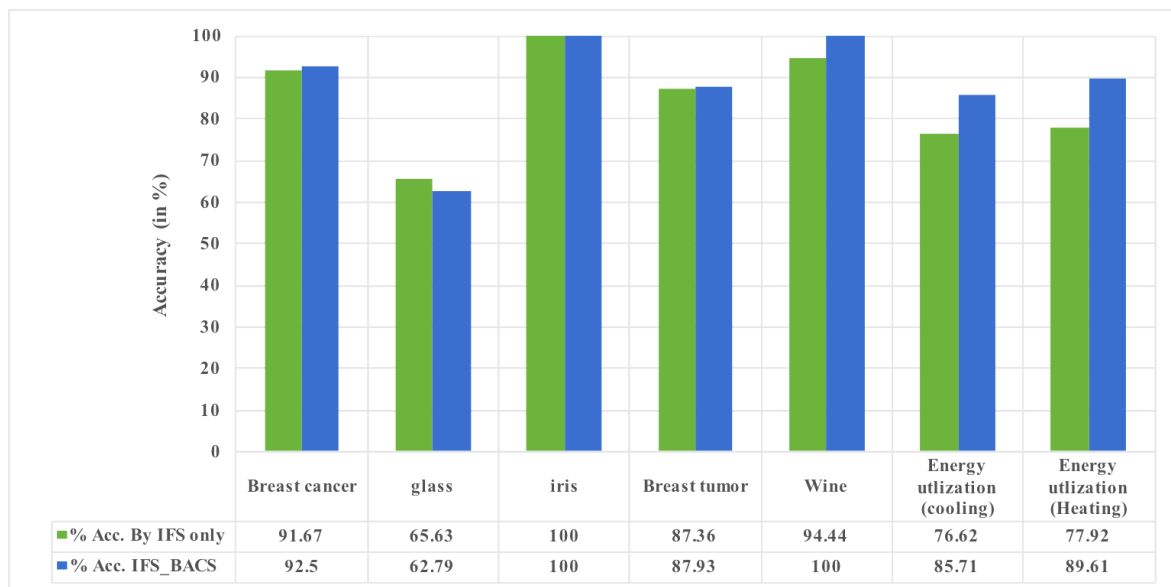


Figure 7.7: Comparison of the Percentage Prediction Accuracy between IFS and IFS_BACS Methods for Classification Tasks

Despite that there is a significant performance difference between IFS and IFS_BACS methods based on their performance evaluation, Figure 7.7 shows that in some cases IFS competitively performed very well, for example, glass, breast cancer, iris and breast tumor

datasets. However, overall results show that IFS_BACS method has superior performance to IFS method.

CHAPTER 8

CONCLUSION

This chapter presents a summary of the essential claims of this thesis. The theory and the ideas supporting the investigated methods. Also included in this chapter is the account of the significant of various experiments conducted in support of the methods. Finally, the conclusions reached.

8.1 Summary

The essential claim of this thesis is that effective feature selection for supervised learning can be achieved based on the dependency information between features and the target attribute. On this basis, an information theoretic based feature selection (IFS) algorithm was implemented and tested empirically.

Chapter 4 presents details methodology for formulating mutual information (MI) as a selection criterion to IFS algorithm, an estimator that enable IFS algorithm to support feature selection for both classification and regression machine learning tasks. IFS algorithm is designed and implemented to support the essential claim of this thesis in such a way that features with strong mutual information with the target attribute are selected whereas a feature with very weak correlation with the target attribute is rejected. The ability of mutual information to capture both linear and nonlinear dependency information (functional dependency) between features and preserved it in the test data was cited (as one of the reasons for its choice) among the outlined rationale and motivation in support of the essential thesis claim.

In addition to the proposed IFS algorithm design to support the essential thesis claim, IFS equally handle other claims of this thesis such as the hypothesis that "*A relevant **feature subset** is one in which the **dependency information** between each feature in the subset and a predictive output is **high**. Yet **with minimal redundancy** among features in the subset.*"

Chapter 5 discusses datasets from various domain which were used to evaluate IFS method. Out of the 12 datasets, there are 2 artificial domains and 10 natural domains. One of the 2 Artificial domain (A1) was synthesized using 6 different functional relationships to test the accuracy of the formulated MI estimator and the second one (A2) was generated to test the discriminating ability of IFS algorithm to select feature in a control experiment. It consists of 10 randomly generated variables, where some of the variables were deliberately used in a function relationship to produce the output variable. Also, the same A2 dataset was modified several times to test the redundancy check steps of the IFS algorithm. All the remaining ten natural domains were used to test the performance of IFS method based on their predictive performance with machine learning algorithms, and they cut across both the classification and the regression tasks of supervised machine learning. This chapter ends with detail experimental methodology adopted in this thesis to test the effectiveness of IFS method.

Chapter 6 detailed information on the various experiments conducted to evaluate IFS method with or without the use of machine learning algorithms. To evaluate IFS selection criterion function (IFS_MI estimator), the comparison between the IFS_MI estimator with six MI estimator benchmarks conducted and the IFS_MI estimator has performed competitively high with most known MI estimator benchmarks.

The second artificial dataset (A2) was used in a controlled study to test the ability of IFS algorithm to discriminate between relevant and irrelevant features. Five out of the ten

variables of A2 were used to generate the output variable, whereas the remaining five variables were assumed to be independent of the output variable. After severally repeated simulations with the IFS method, the result indicated that this approach is capable of selecting features that are relevant to the output variables in most of the runs. Also, to test the redundancy in features, several copies of A2 were made and modified to insert redundancy at different percentage level. In the first copy, two most strongly relevant variables were identified and made to replace the last two irrelevant variables in the original A2. The other remaining copies were modified with a certain percentage (100%, 80%, 50%, etc.) of the two strongly relevant variables after randomly permuted replaced the last two irrelevant variables. Experimental results indicated that except 10% redundant copy, all random copies of A2 were seen to select at least one or both original and redundant copies of the two strongly relevant variables, along with other variables when the redundancy check function was off. In fact, this study further confirmed that regarding the functional relationship used to generate A2 in this thesis, up to 20% random redundant values between two variables is significant enough to influence feature selection method if redundancy is not handled.

Lastly in the same Chapter 6, ten remaining datasets obtained from UCI repository representing natural domains were used to evaluate IFS algorithm using three or four different machine learning algorithms. Since the relevant features for a dataset are unknown in advance in a natural domain setting, the effectiveness of a feature selection algorithm is known through training and testing a machine learning algorithm using a dataset with and without feature selection. In most datasets, IFS algorithm was able to reduce the original full feature set to optimal feature subsets. However, concerning the effectiveness of the IFS method based on the performance of a machine learning algorithm, 15 sets of experiments were conducted for both classification and regression tasks. In each of the experiment, training, and testing of

machine learning algorithm were made using a dataset without and with IFS algorithm. The overall results of the experiments have indicated that IFS algorithm performed competitively high. Also, achieving a competitive performance with reduced feature subset has a positive impact in building a concise model which often avoid overfitting and generalized better. Also, it reduces the cost of generating or collecting large dimensional dataset and as well as reduced model complexities. Finally, in the same Chapter 6, an alternative comparison of IFS method with some other five popular feature selection methods were made. Using SVM machine learning algorithm with classification datasets the results indicated the superior performance of IFS method most of the time and some other times it performed competitively with other feature selection methods.

Chapter 7 proposed and investigated a hybrid feature selection. Moreover, the hybrid method involves IFS method and an Ant Colony Optimization System (ACO). The idea is to improve the performance of IFS method as a filter method using a metaheuristic global search method. Feature selection problem was defined as a Multidimensional Knapsack Problem (MKP), and a Binary Ant Colony System (BACS) was proposed as a 0-1 knapsack problem for optimal feature selection. Performance evaluation of IFS_BACS hybrid method was conducted using nine of the ten natural domain datasets. Also, the results obtained were compared with the results of the previous filter based IFS algorithm. There is a significant reduction in the number of features for all dataset when IFS_BACS method was used to select optimal feature subset. However, regarding comparison for feature subset selection between IFS and IFS_BACS methods, both methods performed competitively. Finally, concerning prediction performance evaluation, by and large, IFS_BACS is more accurate compare to IFS method or compare to original feature before feature selection method is applied to the datasets.

8.2 Conclusion

In most cases, an optimal feature subset is not distinctive, more often similar performance can be achieved with a different set of feature subsets, this was observed empirically throughout this study. Thus, an effective feature selection algorithm should be able to significantly reduce weak or irrelevant features relatively. Also, empirical observations in this study suggested that having a reduced feature subset does not guaranty superior performance with any machine learning algorithms. Different machine learning algorithms are suitable for different supervised learning tasks, and none can be said to be superior to other. It requires adequate knowledge and skill on a specific machine learning algorithm to be able to choose its parameters that will lead to superior performance with the right supervised learning task. For these reasons, in many cases, IFS can significantly enhance the performance of a machine learning algorithm and some other cases provides a competitive performance of a machine learning algorithm, and in all the cases with a reduced feature subset. However, IFS often fail to significantly reduce irrelevant or weak features for a dataset with high feature dimension.

The hybrid filter-metaheuristic method provides a better result than filter only. However, this comes with an additional overhead cost. For instance, IFS_BACS outperformed the proposed IFS method, however, on average it takes 4 to 5 times the execution time of IFS (Samples of the runtime complexity and optimization are presented in Appendix A). Also, IFS_BACS method shows promising results in terms of mitigating the cost of dimensionality affecting filter based IFS method in high dimensional datasets.

Lastly, although it makes some sense to compute percentage prediction accuracy for classification tasks, such computation is baseless for regression tasks. What is more important to regression task is to reduce as much as possible error criterion thus, the best performing

machine learning model is one with consistent low error values across all performance evaluation metrics.

8.3 Future work

One of the limitations of IFS is that occasionally it fails to remove all irrelevant features in a higher dimensional dataset possibly due to the empty space phenomenon. However, this limitation can be mitigated by IFS_BACS method with additional overhead cost. For this reason, the future work is to see how to be able to scale IFS method to handle higher dimensional features.

REFERENCES

- Aeberhard, S., Coomans, D., & Vel, O. D. (1992). *Comparison of Classifiers in High Dimensional Settings*. Technical Report No. 92-02, James Cook University of North Queensland, Department of Computer Science and Department of Mathematics and Statistics.
- Alaya, I., Solnon, C., & Ghéira, K. (2004). Ant algorithm for the multi-dimensional knapsack problem. *International conference on bioinspired optimization methods and their applications. BIOMA*, (pp. 63–72).
- Anonymous. (2011). *What is Machine Learning?* (P. D. Kappen, P. D. Kok, Producers, & SNN Adaptive Intelligence) Retrieved 2015, from Machine Learning in the Netherlands.: <http://www.mlplatform.nl/what-is-machine-learning/>.
- Battiti, R. (2008). Mutual Information for Feature Selection. *Workshop: Information beyond Shannon*. Venice: Lion Laboratory (Machine Learning and Intelligent Optimization).
- Battiti, R. (1994). Using Mutual Information for Selecting Features in Supervised Neural Net Learning. *IEEE Transaction on Neural Networks*, 5(4), 537-550.
- Benoît, F., Gauthier, D., & Michel, V. (2013). Is mutual information adequate for feature selection in regression? *Neural Networks letter*, 48, 1-7.
- Bi, J., & Zhang, T. (2004). Support Vector Classification with Input Data Uncertainty. *Advances in Neural Information Processing Systems*, 17, 161-168.
- Bishop, C. M. (2006). Pattern Recognition and Machine Learning. In M. Jordan, J. Kleinberg, & B. Schölkopf (Eds.), *Information Science and Statistics*. Cambridge, UK: Springer.
- Black, P. E. (2005). Greedy Algorithm. In V. Pieterse, & P. E. Black (Eds.), *Dictionary of Algorithms and Data Structures* [Online]. US: National Institute of Standards and Technology (NIST).
- Blanco, M., Coello, J., Eustaquio, A., Iturriaga, H., & Maspoch, S. (1999). Analytical control of pharmaceutical production steps by near infrared reflectance spectroscopy. *Analytica Chimica Acta*, 392(2-3), 237-246.

- Blanco, M., Coello, J., Fraga, J. M., Iturriaga, H., Maspoch, S., & Pagés, J. (1997). Determination of finishing oils in acrylic fibres by Near-Infrared Reflectance Spectroscopy. *Analyst*, 122, 777-781.
- Blum, A. L., & Langley, P. (1997). Selection of Relevant Features and Examples in Machine Learning. *Artificial Intelligence*, 97(1-2), 245-271.
- Blum, C., & Dorigo, M. (2004). The hyper-cube framework for ant colony optimization. *IEEE Transactions on Man, Systems and Cybernetics—Part B*, 34(2), 1161–72.
- Bonabeau, E., Theraulaz, G., & Dorigo, M. (1999). *Swarm Intelligence: From Natural to Artificial Systems*. New York: Oxford University Press.
- Boryczka, U. (2007). Ants and Multiple Knapsack Problem. *Computer Information Systems and Industrial Management Applications. CISIM '07*. Minneapolis, MN, USA.
- Brown, G. (2009). A New Perspective for Information Theoretic Feature Selection. *12th International Conference on Artificial Intelligence and Statistics*, 5, 49-56.
- Brown, G., Pocock, A., Zhao, M.-j., & Luján, M. (2012). Conditional Likelihood Maximisation: A Unifying Framework for Information Theoretic Feature Selection. *Machine Learning Research*, 13, 27-66.
- Carvalho, V. R., & Cohen, W. W. (2006). Single-pass Online Learning: Performance, Voting, Schemes and Online Feature Selection. *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '06)* (pp. 548-553). Philadelphia: ACM.
- Catlett, J. (1991). *Inductive Learning from Subsets or Disposal of Excess Training Data Considered Harmful*. University of Sydney, Basser Department of Computer Science. Sydney: University of Sydney.
- Catlett, J. (1992). Peephaling: Choosing Attributes Efficiently for Megainduction. In L. A. Birnbaum, & G. C. Colins (Eds.), *ML1992 Proceedings of the Ninth International Workshop on Machine Learning* (pp. 49-54). Morgan Kaufmann Publishers.

- Chai, T., & Draxler, R. R. (2014). Root Mean Square Error (RMSE) or Mean Absolute Error (MAE)? – Arguments Against Avoiding RMSE in the Literature. *Geoscientific Model Development*, 7(3), 1247-1250.
- Charikar, M., Guruswami, V., Kumar, R., Rajagopalan, S., & Sahai, A. (2000). Combinatorial feature selection problems. *41st Annual Symposium on Foundations of Computer Science, FOCS* (pp. 631-64). California, USA: IEEE Computer Society.
- Chen, Y.-W., & Lin, C.-J. (2006). Combining SVMs with Various Feature Selection Strategies. In I. Guyon, M. Nikravesh, S. Gunn, & L. A. Zadeh (Eds.), *Feature Extraction Foundations and Applications* (Vol. 207, pp. 315-324). Berlin Heidelberg: Springer.
- Chiu, D. K., Cheung, B., & Wong, A. K. (1990). Information Synthesis based on Hierarchical Entropy Discretization. *Journal of Experimental and Theoretical Artificial Intelligence*, 2(2), 117-129.
- Chmielewski, M. R., & Grzymala-Busse, J. W. (1996). Global Discretization of Continuous Attributes as Preprocessing for Machine Learning. *International Journal of Approximate Reasoning*, 15(4), 319-331.
- Chowdhury, R. A., Hai-Son, L., & Ramachandran, V. (2010). Cache-oblivious Dynamic Programming for Bioinformatics. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 7(3), 495–510.
- Collins, R. T., & Liu, Y. (2005). On-Line Selection of Discriminative Tracking Features. *Pattern Analysis and Machine Intelligence*, 27(10), 1631 - 1643.
- Cortez, P., & Morais, A. (2007). A Data Mining Approach to Predict Forest Fires using Meteorological Data. *Proceedings of the 13th EPIA 2007 - Portuguese Conference on Artificial Intelligence* (pp. 512-523). Associação Portuguesa para a Inteligência Artificial (APPIA).
- Da'vid, P., Barnabá's, P., & Csaba, S. (2010). Estimation of Rényi Entropy and Mutual Information Based on Generalized Nearest-Neighbor Graphs. *Advances in Neural Information*

- Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems*, (pp. 1849 -1857). British Columbia.
- Daniel, W. W. (1990). Kruskal–Wallis One-Way Analysis of Variance by Ranks. In W. W. Daniel, *Applied Nonparametric Statistics* (2nd ed., pp. 226–234). Boston: PWS-Kent.
- Dantzig, G. (1957). Discrete-Variable Extremum Problems. *Operations Research*, 5(2), 266–277.
- Das, S. (2001). Filters, Wrappers and a Boosting-based Hybrid for Feature Selection. In *Proceedings of the Eighteenth International Conference on Machine Learning* (pp. 74–81). San Francisco: Morgan Kaufmann Publishers Inc.
- Dash, M. (1997). Feature Selection via Set Cover. *Knowledge and Data Engineering Exchange Workshop, 1997. Proceedings.* (pp. 165 - 171). New port Beach: IEEE Explore Digital library.
- Dash, M., & Liu, H. (2003). Consistency-based search in feature selection. *Artificial Intelligence*, 151, 155–176.
- de Werra, D., & Hertz, A. (1989). Tabu search techniques A tutorial and an application to neural networks. *OR Spektrum*, 11(3), 131-141.
- Ditzler, G. C. (2015). *Scalable Subset Selection with Filters and Its Applications*. Drexel University.
- Doerr, B., Ereimeev, A., Neumann, F., Theile, M., & Thyssen, C. (2011). Evolutionary Algorithms and Dynamic Programming. *Theoretical Computer Science*, 412(43), 6020-6035.
- Doquire, G., & Verleysen, M. (2011). Feature Selection with Mutual Information for Uncertain Data. In A. Cuzzocrea, & U. Dayal (Eds.), *Data Warehousing and Knowledge Discovery DaWaK 2011. Lecture Notes in Computer Science* (Vol. 6862, pp. 330- 341). Berlin Heidelberg: Springer.
- Dorigo, M., & Gambardella, L. M. (1997). Ant Colony System: Cooperative Learning Approach to the Travelmg Salesman Problem. *IEEE Transaction on Evolutionary Computation*, 1(1), 53-66.
- Dorigo, M., & Stutzle, T. (2004). *Ant Colony Optimization*. MIT Press.

- Dorigo, M., Birattari, M., & Stutzle, T. (2006). Ant Colony Optimization: Artificial Ants as a Computational Intelligence Technique. *IEEE Computational Intelligence Magazine*, 1, 28-39.
- Dorigo, M., Caro, G. D., & Gambardella, L. M. (1999). Ant Algorithms for Discrete Optimization. *Artificial Life*, 5(3), 137-172.
- Dougherty, J., Kohavi, R., & Sahami, M. (1995). Supervised and Unsupervised Discretization of Continuous Features. *Machine Learning: Proceedings of the Twelfth International Conference* (pp. 194-202). Morgan Kaufmann.
- Duch, W. (2006). Filter Methods. In I. Guyon, S. Gunn, M. Nikraves, & L. A. Zadeh (Eds.), *Feature Extraction Foundations and Applications* (Vol. 207, pp. 89-117). Berlin Heidelberg: Springer.
- Eggermont, P., & LaRiccia, V. (2001). *Maximum Penalized Likelihood Estimation*. New York: Springer.
- Elter, M., Schulz-Wendtland, R., & Wittenberg, T. (2007). The Prediction of Breast Cancer Biopsy outcomes using two CAD Approaches that both Emphasize an Intelligible Decision Process. *Medical Physics*, 34(11), 4164-4172.
- Everitt, B. S. (1998). *The Cambridge Dictionary of Statistics* (3rd ed.). Cambridge, UK: Cambridge University Press.
- Evetts, I. W., & Spiehler, E. J. (1987). *Rule Induction in Forensic Science*. Central Research Establishment, Home Office Forensic Science Service, Aldermaston, Reading, Berkshire RG7 4PN, Aldermaston.
- Fayyad, U. M., & Irani, K. B. (1993). Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning. *13th International Joint Conferences on Uncertainty in Artificial Intelligence*, (pp. 1022-1027).
- Fayyad, U. M., Piatetsky-Shapiro, G., & Smyth, P. (1997). From Data Mining to Knowledge Discovery: an Overview. *ACM Digital Library Advances in Knowledge Discovery and Data Mining (1996)*, 1297, 1-34.

- Fidanova, S. (2003). ACO algorithm for MKP using various heuristic information. In D. I. I. Lirkov, S. Margenov, & Z. Zlatev (Eds.), *Numerical methods and applications* (pp. 434-330). Berlin, Germany: Springer.
- Fidanova, S. (2007). Ant Colony Optimization and Multiple Knapsack Problem. In J. P. Renard (Ed.), *Handbook of Research on Nature-Inspired Computing for Economics and Management* (Vol. 1, pp. 498-509). Idea Group Inc.
- Fidanova, S. (2005). Heuristics for Multiple Knapsack Problem. In N. Guimarães, & P. Isaías (Eds.), *IADIS International Conference on Applied Computing* (Vol. 2, pp. 255-260). Algarve, Portugal.
- Fleuret, F. (2004). Fast Binary Feature Selection with Conditional Mutual Information. *The Journal of Machine Learning Research*, 5, 1531–1555.
- François, D., Rossi, F., Wertz, V., & Verleysen, M. (2007). Resampling Methods for Parameter-free and Robust Feature Selection with Mutual Information. *Neurocomputing*, 70(7-9), 1276-1288.
- Gadat, S., & Younes, L. (2007). A Stochastic Algorithm for Feature Selection in Pattern Recognition. *Journal of Machine Learning Research*, 8, 509-547.
- Gambardella, L. M., Taillard, E. D., & Agazzi, G. (1999b). MACS-VRPTW: A multiple Ant Colony System for Vehicle Routing Problems with Time Windows. In D. Corne, M. Dorigo, & F. Glover (Eds.), *New Ideas in Optimization* (pp. 63–76). London, UK: McGraw- Hill.
- Gambardella, L. M., Taillard, E. D., & Dorigo, M. (1999a). Ant Colonies for the Quadratic Assignment Problem. *Journal of Operational Research Society*, 50(2), 167-176.
- Gao, S., Ver Steeg, G., & Galstyan, A. (2015a). Efficient Estimation of Mutual Information for Strongly Dependent Variables. *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics (AISTATS)*. 38, pp. 277-286. San Diego: W&CP.
- Gao, S., Ver Steeg, G., & Galstyan, A. (2015b). Estimating Mutual Information by Local Gaussian Approximation. *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*, (pp. 278-287). Amsterdam, Netherlands.

- Ghouthi, L., & Al-Bukhaitan, S. (2010). Hybrid Soft Computing for PVT properties Prediction. *proceeding of European Symposium on Artificial Neural Networks – Computational Intelligence and Machine Learning*. Bruges (Belgium).
- Gloer, K., Eads, D., & Theiler., J. (2005). Online Feature Selection for Pixel Classification. *Proceedings of the 22nd International Conference on Machine Learning*, (pp. 249 - 256). Bonn, Germany.
- Gringarten, E. (2012). Integrated uncertainty assessment – from seismic and well-logs to flow simulation. *PARADIGM, Annual Meeting*. Las Vegas.
- Grotschel, M., & Nemhauser, G. L. (2008). George Dantzig's Contributions to Integer Programming. *Discrete Optimization*, 5, 168-173.
- Grunwald, P. (2005). A Tutorial Introduction to the Minimum Description Length Principle. In P. Grunwald (Ed.), *Advances in Minimum Description Length: Theory and Applications* (p. 80). MIT Press.
- Guyon, I., & Elisseeff, A. (2003). An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research*, 3, 1157-1182.
- Hall, M. A. (2000). Correlation-based Feature Selection for Discrete and Numeric Class Machine Learning. *ICML '00 Proceedings of the Seventeenth International Conference on Machine Learning* (pp. 359-366). San Francisco: Morgan Kaufmann Publishers Inc.
- Harnad, S. (2008). The Annotation Game: On Turing (1950) on Computing, Machinery, and Intelligence. In R. Epstein, & G. Peters (Eds.), *Parsing the Turing Test: Philosophical and Methodological Issues in the Quest for the Thinking Computer. Evolving Consciousness* (pp. 23-66). Springer.
- Haykin, S. (2009). *Neural Networks and Learning Machines*. Ontario, Hamilton, Canada: Peason International Edition.
- Hoi, S. C., Wang, J., Zhao, P., & Jin, R. (2012). Online Feature Selection for Mining Big Data. *BigMine '12 Proceedings of the 1st International Workshop on Big Data, Streams and*

Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications (pp. 93–100). Beijing: ACM.

Huan, L., & Lei, Y. (2005). Toward Integrating Feature Selection Algorithms for Classification and Clustering. *IEEE Transactions on Knowledge and Data Engineering*, 17(4), 491-502.

Huang, J., Cai, Y., & Xu, X. (2007). A Hybrid Genetic Algorithm for Feature Selection Wrapper based on Mutual Information. *Pattern Recognition Letters*, 28(13), 1825–1844.

Jordan, M. I. (2014). *Comments*. Retrieved 12 20, 2015, from reddit Machine Learning: https://www.reddit.com/r/MachineLearning/comments/2fxi6v/ama_michael_i_jordan/ckelmt/?context=3.

Jundong, L., Kewei, C., Suhang, W., Fred, M., Robert, P. T., Jiliang, T., & L., Huan (2016). Feature Selection: A Data Perspective. *arXiv:1601.07996 [cs.LG]*.

Kappen, P. D., & Kok, P. D. (2011). *SNN Adaptive Intelligence*. Retrieved 2015, from Machine Learning in the Netherlands.: <http://www.mlplatform.nl/what-is-machine-learning/>.

Kerber, R. (1992). ChiMerge: Discretization of Numeric Attributes. *The Ninth National Conference on Artificial Intelligence*, (pp. 123-128).

Kinney, J., & Atwal, G. (2014). Equitability, Mutual Information, and the Maximal Information Coefficient. *Proceedings of the National Academy of Sciences*, 111, 3354–3359.

Kira, K., & Rendell, L. A. (1992). A Practical Approach to Feature Selection. *Preceedings of the 9th International workshop on machine learning (ML 1992)* (pp. 249-256). ACM.

Kohavi, R., & John, G. H. (1997). Wrappers for Feature Subset Selection. *Journal for Artificial Intelligence*, 97(1-2), 273-324.

Koller, D., & Sahami, M. (1995). Toward Optimal Feature Selection. *13th International Conference on Machine Learning*.

Koller, D., & Sahami, M. (1995). Toward Optimal Feature Selection. In L. Saitta (Ed.), *13th International Conference on Machine Learning, 1995* (pp. 284-292). Italy: Morgan Kaufmann Publishers.

- Kong, M., Tian, P., & Kao, Y. (2008). A New Ant Colony Optimization Algorithm for the Multidimensional Knapsack Problem. *Computers & Operations Research*, 35(8), 2672-2683.
- Kononenko, I. (1994). Estimating attributes : Analysis and Extension of RELIEF. *Proceedings of the Sixth European Conference on Machine Learning* (pp. 171 - 182). Springer-Verlag: ACM Digital Lib.
- Kozachenko, L. F., & Leonenko, N. N. (1987). Sample estimate of entropy of a random vector. *Problems of Information Transmission*, 23(2), 95–101.
- Kraskov, A., Stögbauer, H., & Grassberger, P. (2004). Estimating Mutual Information. *Physical Review E*, 69:066138. doi: 10.1103/PhysRevE.69.066138.
- Kwak, N., & Choi, C. (2002a). Input Feature Selection for Classification Problems. *Neural Networks*, 13(1), 143–159.
- Kwak, N., & Choi, C. (2002b). Input feature selection by mutual information based on Parzen window. *Pattern Analysis and Machine Intelligence*, 24(12), 1667 – 1671.
- Langely, P. (2011). The Changing Science of Machine Learning. *Machine Learning*, 82(3), 275–279.
- Lee, S.-V., & Bau, Y.-T. (2012). An Ant Colony Optimization Approach for Solving the Multidimensional Knapsack Problem. *2012 International Conference on Computer & Information Science (ICCIS)*, 1, 441 - 446. IEEE.
- Leguizamon, G., & Michalevitz, Z. (1999). A new version of ant system for subset problems. *International Conference on Evolutionary Computations*. Washington.
- Leguizamon, G., & Michalewicz, Z. (1999). A New Version of Ant System for Subset Problems. *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99*, 2, 1464. Washington, DC: IEEE.
- Lei, Y., & Huan, L. (2003). Feature Selection for High-Dimensional Data: A Fast Correlation-Based Filter Solution. In T. Fawcett, & N. Mishra (Eds.), *Proceedings, Twentieth International Conference on Machine Learning* (Vol. 2, pp. 856-863). Washington, DC.

- Little, M., McSharry, P., Hunter, E., & Ramig, L. (2009). Suitability of Dysphonia Measurements for Telemonitoring of Parkinson's Disease. *IEEE Transactions on Biomedical Engineering*, 56(4), 1015-1022.
- Liu, H., & Motoda, H. (2000). *Feature Selection for Knowledge Discovery and Data Mining* (1st ed., Vol. 454). New York: Springer originally published by Kluwer Academic Publishers, New York in 1998.
- Liu, H., & Yu, L. (2005). Toward Integrating Feature Selection Algorithms for Classification and Clustering. *IEEE Transactions on Knowledge and Data Engineering*, 17(4), 491–502.
- Liu, H., Motoda, H., & Yu, L. (2004). A Selective Sampling Approach to Active Feature Selection. *Artificial Intelligence*, 59 (1–2), 49-74.
- Maniezzo, V., & Colomi, A. (1999). The Ant System Applied to the Quadratic Assignment Problem. *IEEE Transaction on Data Knowledge Engineering*, 11 (5), 769-778.
- McCarthy, J., & Hayes, P. J. (1987). Some Philosophical Problems from the Standpoint of Artificial Intelligence. In *Readings in Nonmonotonic Reasoning* (pp. 26-45). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.
- Meehan, J. (n.d.). *Heuristic Restart Algorithms for Local Search*. Retrieved 08 16, Lat visit:2017, from University of Wisconsin Madison, Wisconsin: <http://pages.cs.wisc.edu/~jmeehan/ClassProjects/HueristicRestart.pdf>.
- Meyer, P. E. (2014, 7 26). *Information-Theoretic Measures software*. Retrieved 2 8, 2016, from Patrick E. Meyer: <http://homepage.meyerp.com/software>.
- Meyer, P. E., Kontos, K., Lafitte, F., & Bontempi, G. (2007). Information-theoretic inference of large transcriptional regulatory networks. *Journal on Bioinformatics and Systems Biology*.
- Meyer, P. E., Schretter, C., & Bontempi, G. (2008). Information-Theoretic Feature Selection in Microarray Data Using Variable Complementarity. *IEEE Journal of Selected Topics in Signal Processing*, 2(3), 261-274.
- Moon, K. R. and Hero, A. O. (2014) Ensemble estimation of multivariate f-divergence. In *Information Theory (ISIT), 2014 IEEE International Symposium* (pp. 356–360). IEEE.

- Mosteller, F. (1948). A K-sample Slippage Test for an Extreme Population. *The Annals of Mathematical Statistics (JSTOR)*, 19(1), 58–65.
- NASDAC. (2011, 10 27). Retrieved 10 27, 2011, from NASDAC Composite Daily Closing Stock: <http://finance.yahoo.com/q/hp?s=%5EIXIC+Historical+Prices>.
- NIST. (1999). *greedy heuristic in Dictionary of Algorithms and Data Structures*, [online]. (V. Pieterse, & P. E. Black, Eds.) Retrieved 6 12, 2016, from National Institute of Standards and Technology (NIST), U.S.: <https://xlinux.nist.gov/dads//HTML/greedyHeuristic.html>.
- Nixon, M. S., & Aguado, A. S. (2002). *Feature Extraction and Image Processing* (1st ed.). Jordan Hill, Oxford: Butterworth-Heinemann Linacre House.
- Oliveira, A. L., & Vinceritell, A. S. (1992). Constructive Induction Using a Non-greedy Strategy for Feature Selection. *ML92 Proceedings of the 9th International Conference on Machine Learning* (pp. 355-360). Aberdeen, Scotland: Morgan Kaufmann.
- Olshausen, B. (2004, 3 1). *A probability primer*. Retrieved 2016, from NPB 163 / PSC 128 - Winter 2004: Information Processing Models in Neuroscience and Psychology: <http://redwood.berkeley.edu/bruno/npb163/>.
- Ozaki, Y., Cho, R., Ikegaya, K., Muraishi, S., & Kawauchi, K. (1992). Potential of near- infrared Fourier transform Raman spectroscopy in food analysis. *Applied Spectroscopy*, 46, 1503-1507.
- Parzen, E. (1962). On the estimation of a probability density function and the mode. *Annals of Mathematical Statistics*, 33, 1065-1076.
- Paul, J., Maurice, D., & Brian, J. R. (1999). Preparing Clean Views of Data for Data Mining. *12th ERCIM Workshop Proceedings on Database Research* (p. 15). Amsterdam: European Research Consortium for Informatics and Mathematics.
- Peng, H., Long, F., & Ding, C. (2005). Feature Selection based on Mutual Information: criteria of max-dependency, max-relevance, and min- redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8), 1226-38.

- Perkins, S., & Theiler, J. (2003). Online Feature Selection Using Grafting. *ICML'03 Proceedings of the Twentieth International Conference on International Conference on Machine Learning* (pp. 592–599). Washington DC: AAAI Press.
- Pfahring, B. (1995). Compression-Based Discretization of Continuous Attributes. In A. Prieditis, & S. Russell (Ed.), *Proceedings of the twelfth International Conference on Machine Learning* (pp. 456–463). Morgan Kaufmann.
- Po'czos, B., Xiong, L., & Schneider, J. (2011). Nonparametric divergence estimation with applications to machine learning on distributions. *Proceedings of Uncertainty in Artificial Intelligence (UAI)* (pp. 599–608). ACM Digital Library.
- Potts, C. N., & Kovalyov, M. Y. (2000). Scheduling with Batching: A Review. *European Journal of Operational Research*, 120(2), 228–249.
- Pudil, P., & Somol, P. (2005). Current Feature Selection Techniques in Statistical Pattern Recognition. (M. Kurzyński, E. Puchała, M. Woźniak, & A. Żolnier, Eds.) *Computer Recognition Systems. Advances in Soft Computing*, 30, 53–68.
- Redmond, M. A., & Baveja, A. (2002). A Data-Driven Software Tool for Enabling Cooperative Information Sharing among Police Departments. *European Journal of Operational Research*, 141(3), 660–678.
- Reshef, D. N., Reshef, Y. A., Finucane, H. K., Grossman, S. R., McVean, G., Turnbaugh, P. J., Lander, E. S., Mitzenmacher, M., & Sabeti, P. C. (2011). Detecting Novel Associations in Large Data Sets. *Science*, 334(6062), 1518–1524.
- Richeldi, M., & Lanzi, P. L. (1996). Performing Effective Feature Selection by Investigating the Deep Structure of the Data. *KDD'96 Proceedings of the Second International Conference on Knowledge Discovery and Data Mining* (pp. 379–382). Oregon: AAAI Press.
- Richeldi, M., & Rossotto, M. (1995). Class-driven Statistical Discretization of Continuous Attributes (Extended Abstract). In N. Lavrac, & S. Wrobel (Eds.), *Machine Learning: ECML-95. ECML 1995. Lecture Notes in Computer Science (Lecture Notes in Artificial Intelligence)* (Vol. 912, pp. 335–338). Heraklion, Berlin: Springer.

- Rissanen, J. (1997). Stochastic Complexity and Modelling. *Journal of Computer and System Sciences*, 55(1), 89-95.
- Ross, S. M. (2010). *Introduction to Probability Models* (10th ed.). Los Angeles, California, USA: Academic Press.
- Rossi, F., Lendasse, A., Fran'cois, D., Wertz, V., & Verleysen, M. (2006). Mutual Information for the Selection of Relevant Variables in Spectrometric Nonlinear Modelling. *Chemometrics and Intelligent Laboratory Systems*, 80(2), 215–226.
- Saeys, Y., Inza, I., & Larrañaga, P. (2007). A Review of Feature Selection Techniques in Bioinformatics. *Bioinformatics*, 23(19), 2507–2517.
- Sekulic, S. S., Ward, H. W., Brannegan, D. R., Stanley, E. D., Evans, C. L., Sciavolino, S. T., Hailey, P. E., & Aldridge, P. K. (1996). Online monitoring of powder blend homogeneity by near-infrared spectroscopy. *Analytical Chemistry*, 68(3), 509-513.
- Seni, G., & Elder, J. (2010). Ensemble Methods in Data Mining: Improving Accuracy Through Combining Predictions. In R. Grossman, G. Seni, J. F. Elder, N. Agarwal, & H. Liu (Eds.), *Synthesis Lectures on Data Mining and Knowledge Discovery* (Vol. 2, p. 1-126). Morgan & Claypool Publishers.
- Seymour, G. (1993). *Predictive Inference: An Introduction*. (Vol. 55). New York: Chapman & Hall.
- Shannon, C. E. (1948). A Mathematical Theory of Communication. *The Bell System Technical Journal*, 27(3), 379–423.
- Singh, S., & Poczos, B. (2014a). Generalized exponential concentration inequality for renyi divergence estimation. *31st International Conference on Machine Learning (ICML-14)*, (pp. 333–341).
- Singh, S., & Póczos, B. (2014b). Exponential Concentration of a Density Functional Estimator. *Advances in Neural Information Processing Systems*, (pp. 3032-3040).
- Sivagaminathan, R. K., & Ramakrishnan, S. (2007). A Hybrid Approach for Feature Subset Selection Using Neural Networks and Ant Colony Optimization. *Expert Systems with Applications*, 33(1), 49–60.

- Slovic, P., & Lichtenstein, S. (1971). Comparison of Bayesian and Regression Approaches to the Study of Information Processing in Judgment. *Organizational Behavior and Human Performance*, 6(6), 649–744.
- Song, L., Smola, A., Gretton, A., Bedo, J., & Borgwardt, K. (2012). Feature Selection via Dependence Maximization. *Journal of Machine Learning Research*, 13(1), 1393 – 1433.
- Street, W. N., Mangasarian, O. L., & Wolberg, a. W. (1995). An Inductive Learning Approach to Prognostic Prediction. In A. Prieditis, & S. Russell (Ed.), *Proceedings of the Twelfth International Conference on Machine Learning* (pp. 522-530). San Francisco: Morgan Kaufmann.
- Sulaiman, M. A., & Labadin, J. (2015a). Feature Selection Based on Mutual Information for Machine Learning Prediction of Petroleum Reservoir Properties. *The 9th International Conference on IT in Asia*. (pp. 1-6). Sarawak: IEEE.
- Sulaiman, M. A., & Labadin, J. (2015b). Feature Selection Based on Mutual Information for Regression Problems. *The 9th International Conference on IT in Asia* (pp. 1-6). Sarawak: IEEE.
- Sulaiman, M. A., & Labadin, J. (2016). Improved Feature Selection Based on Mutual Information for Regression Tasks. *Journal of Information Technology in Asia*, 6(1), 11-24.
- Sumathy, K., & Chidambaram, M. (2013). Text Mining: Concepts, Applications, Tools and Issues – An Overview. *International Journal of Computer Applications*, 80(4), 29 – 32.
- Suzuki, T., Sugiyama, M., Sese, J., & Kanamori, T. (2008). Approximating mutual information by maximum likelihood density ratio estimation. In Y. Saeys, H. Liu, I. Inza, L. Wehenkel, & Y. Van de Peer (Eds.), *FSDM, JMLR Proceedings*, 4, pp. 5–20.
- Szabó, Z. (2015, 2 8). *Information Theoretical Estimators (ITE) Toolbox Release 0.61*. Retrieved 2 8, 2016, from Atlassian Bitbucket: <https://bitbucket.org/szzoli/ite/>.
- Ting, K. M. (1994). *Discretization of Continuous-Valued Attributes and Instance-Based Learning*. University of Sydney. Basser Department of Computer Science, University of Sydney.

- Tsanas, A., & Xifara, A. (2012). Accurate Quantitative Estimation of Energy Performance of Residential Buildings Using Statistical Machine Learning Tools. *Energy and Buildings*, 49, 560-567.
- Tsang, S., Kao, B., Yip, K., Ho, W. S., Ho, W. S., & Lee, S. (2011). Decision Trees for Uncertain Data. *IEEE Transactions on Knowledge and Data Engineering*, 23(1), 64–78.
- Unler, A., Murat, A., & Chinnam, R. B. (2011). A Maximum Relevance Minimum Redundancy Feature Selection Method based on Swarm Intelligence for Support Vector Machine Classification. *Information Sciences*, 181(20), 4625 – 4641.
- Vergara, J. R., & Estévez, P. A. (2014). A review of feature selection methods based on mutual information. *Neural Computing & Applications*, 24(1), 175–186.
- Verleysen, M., Rossi, F., & François, D. (2009). Advances in Feature Selection with Mutual Information. In *Similarity-Based Clustering* (Vol. 5400, pp. 52-69). Springer Berlin Heidelberg.
- Wang, D., Irani, D., & Pu, C. (2012). Evolutionary Study of Web Spam: Web Spam Corpus 2011 Versus Web Spam Corpus 2006. *COLLABORATECOM '12 Proceedings of the 8th International Conference on Collaborative Computing: Networking, Applications and Worksharing* (pp. 40–49). Pittsburgh: IEEE.
- Wang, J., Zhao, Z.-Q., Hu, X., Cheung, Y.-m., Wang, M., & Wu, X. (2013). Online Group Feature Selection. *IJCAI '13 Proceedings of the Twenty-Third international joint conference on Artificial Intelligence* (pp. 1757-1763). Beijing: AAAI Press.
- Wattenberg, A. J. (1994). Stochastic hill climbing as a baseline method for evaluating genetic algorithms. Berkeley: University of California.
- Weiss, S. M., & Kulikowski, C. A. (1991). *Computer Systems that Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems*. Morgan Kaufmann Publishers Inc.

- Willmot, C. J., & Matsuura, K. (2005). Advantages of the Mean Absolute Error (MAE) over the Root Mean Square Error (RMSE) in Assessing Average Model Performance. *Climate Research*, 30(1), 79–82.
- Woeginger, G. J. (2000). When Does a Dynamic Programming Formulation Guarantee the Existence of a Fully Polynomial Time Approximation Scheme (FPTAS)? *INFORMS Journal on Computing*, 12(1), 57–74.
- Woods, W. (1986). Important Issues in Knowledge Representation. *Proceedings of the Institute of Electrical and Electronics Engineers*, 74(10), 1322-1334. IEEE.
- Wu, Y., Hoi, S. C., & Mei, T. (2014). Massive-Scale Online Feature Selection for Sparse Ultra-high Dimensional Data. *eprint arXiv:1409.7794*, 12.
- Xing, E., Jordan, M. I., & Karp, R. M. (2001). Feature Selection for High-Dimensional Genomic Microarray Data. *ICML '01 Proceedings of the Eighteenth International Conference on Machine Learning* (pp. 601-608). San Francisco: Morgan Kaufmann Publishers Inc.
- Yang, H., & Moody, J. (1999). Feature selection based on joint mutual information. *International ICSC Symposium on Advances in Intelligent Data Analysis*, (pp. 22–2). Rochester, NY.
- Yeh, I.-C. (1998). Modeling of Strength of High Performance Concrete Using Artificial Neural Networks. *Cement and Concrete Research*, 28(12), 1797-1808.
- Yu, K., Wu, X., Ding, W., & Pei, J. (2014). Toward Scalable and Accurate Online Feature Selection for Big Data. *IEEE International Conference on Data Mining*. (pp. 660 - 669). IEEE.
- Yu, L., & Liu, H. (2004). Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research*, 5, 1205–1224.
- Yu, L., & Liu, H. (2003). Feature Selection for High-Dimensional Data: A Fast Correlation-Based Filter Solution. In T. Fawcett, & N. Mishra (Eds.), *Proceeding of the Twentieth International Conference on Machine Learning (ICML-2003)*, 2, 856-863. Washington DC.
- Zaffalon, M., & Hutter, M. (2002). Robust Feature Selection by Mutual Information Distributions. *UAI'02 Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence* (pp. 577-584). San Francisco: Morgan Kaufmann Publishers Inc.

Zheng, Z., Fred, M., Shashvata, S., Salem, A., Aneeth, A., & Huan, L. (2010). *Advancing Feature Selection Research – ASU Feature Selection Repository*. Arizona State University, Computer Science and Engineering. FEATURE SELECTIONat Arizona State University in Conjunction with the DMML: <http://featureselection.asu.edu/old/index.php>.

APPENDICES

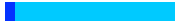
Appendix A: Runtime Complexity and Optimization

Generated 19-Aug-2016 10:40:52 using cpu time.

Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
ClassificationECOC>localFitECOC/loopBody	10	4.615 s	0.010 s	
SVMMain	1	1.227 s	0.058 s	
fitcecoc	1	0.740 s	0.000 s	
FitTemplate>FitTemplate.fit	11	0.711 s	0.019 s	
...t:ClassificationSVM.ClassificationSVM	10	0.331 s	0.000 s	
SVMImpl>SVMImpl.make	10	0.253 s	0.234 s	
templateSVM	1	0.214 s	0.097 s	
...>CompactClassificationECOC.predict	2	0.195 s	0.019 s	
smartForSliceout	5	0.175 s	0.029 s	
...ClassificationECOC.ClassificationECOC	1	0.156 s	0.019 s	
...OC>CompactClassificationECOC.score	2	0.156 s	0.010 s	
FitTemplate>FitTemplate.make	2	0.146 s	0.010 s	
...ectorOps>DisallowVectorOps.subsref	240	0.136 s	0.029 s	
FitTemplate>FitTemplate.fillIfNeeded	3	0.107 s	0.078 s	
...nSVM>ClassificationSVM.prepareData	10	0.107 s	0.029 s	
parseArgs	59	0.097 s	0.078 s	
parseArgs	59	0.097 s	0.000 s	
CompactClassificationECOC>localLoss	2	0.097 s	0.000 s	
...ssificationECOC>localLoss/loopBody	768	0.078 s	0.019 s	
...t:FullClassificationModel.prepareData	1	0.068 s	0.010 s	
ClassificationECOC>localFitECOC	1	0.068 s	0.000 s	
processParallelAndStreamOptions	3	0.058 s	0.010 s	
...mean(max(0,1-bsxfun(@times,M,f),2)/2	768	0.058 s	0.029 s	
...ficationModel.FullClassificationModel	11	0.049 s	0.000 s	
ModelParams>ModelParams.fillIfNeeded	11	0.049 s	-0.000 s	
...nModel>ClassificationModel.predict	20	0.049 s	0.019 s	
...M>CompactClassificationSVM.predict	20	0.049 s	0.000 s	

Appendix A1: Runtime Analysis for IFS SVM Implementation Only

Generated 19-Aug-2016 10:57:43 using cpu time.

Function Name	Calls	Total Time	Self Time *	Total Time Plot (dark band = self time)
main_multiReg	1	4.608 s	0.034 s	
gradientDescentMulti	1	4.548 s	0.283 s	
computeCostMulti	300	4.266 s	4.266 s	
close	1	0.009 s	0.009 s	
trainTestPartition	1	0.009 s	0.009 s	
predictReg	2	0.009 s	0.000 s	
mean	4	0.009 s	0.009 s	
close>getEmptyHandleList	1	0 s	0.000 s	
close>safegetchildren	1	0 s	0.000 s	
close>checkfigs	1	0 s	0.000 s	
close>request_close	1	0 s	0.000 s	
std	2	0 s	0.000 s	
var	2	0 s	0.000 s	

Self time is the time spent in a function excluding the time spent in its child functions. Self time also includes overhead resulting from the process of profiling.

Appendix A2: Runtime Analysis for IFS Multi-Reg. Implementation Only

Generated 19-Aug-2016 18:06:11 using cpu time.

Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
IFSMain	1	35.922 s	0.092 s	
IFSmethod	1	32.906 s	1.525 s	
computeMI	4532	21.834 s	1.871 s	
computeEntropyofX	13596	18.360 s	1.497 s	
simpleTrapofX	6996	13.756 s	13.756 s	
featureCount	1	8.982 s	8.982 s	
stoppingCrit	112	5.430 s	0.219 s	
pdfEstimator	6996	3.107 s	0.664 s	
computeJointEntropyXY	9064	2.761 s	0.982 s	
SVMmodel	1	2.690 s	0.014 s	
copdfExponent	6996	2.443 s	1.645 s	
fitcecoc	50	1.878 s	0.007 s	
FitTemplate>FitTemplate.fit	100	1.667 s	0.071 s	
ClassificationECOC>localFitECOC/loopBody	50	1.208 s	0.007 s	
pdfKernel	6996	0.798 s	0.798 s	
...t:ClassificationSVM.ClassificationSVM	50	0.628 s	0.035 s	
joinPreprocess	9064	0.621 s	0.621 s	
FitTemplate>FitTemplate.make	100	0.523 s	0.028 s	
...ClassificationECOC.ClassificationECOC	50	0.480 s	0.021 s	
FitTemplate>FitTemplate.fillIfNeeded	150	0.459 s	0.141 s	
SVMImpl>SVMImpl.make	50	0.424 s	0.177 s	
...>CompactClassificationECOC.predict	50	0.403 s	0.021 s	
parseArgs	1050	0.388 s	0.021 s	
templateSVM	50	0.388 s	0.071 s	
parseArgs	1050	0.367 s	0.120 s	
smartForSliceout	150	0.367 s	0.113 s	
redundancyCheck	156	0.311 s	0.113 s	

Appendix A3: Runtime Analysis for IFS + SVM Model before Redundancy Check Implementation

Generated 19-Aug-2016 18:17:28 using cpu time.

Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
IFSMain	1	209.235 s	0.063 s	
IFSmethod	1	148.579 s	6.383 s	
computeMI	10659	125.884 s	5.351 s	
computeEntropyofX	21318	118.166 s	3.469 s	
simpleTrapofX	11418	108.322 s	108.322 s	
MultiRegmodel	1	60.474 s	0.407 s	
gradientDescentMulti	50	59.926 s	1.833 s	
computeCostMulti	50000	58.093 s	58.093 s	
featureCount	1	11.327 s	11.327 s	
pdfEstimator	11418	6.376 s	1.215 s	
copdfExponent	11418	5.161 s	3.062 s	
stoppingCrit	69	4.845 s	0.147 s	
constr	7623	4.726 s	0.477 s	
redundancyCheck	7738	4.290 s	1.060 s	
ismember	7863	3.230 s	0.470 s	
ismember>ismemberR2012a	7863	2.760 s	1.131 s	
computeJointEntropyXY	10659	2.121 s	1.334 s	
pdfKernel	11418	2.100 s	2.100 s	
ismember>ismemberBuiltinTypes	7863	1.629 s	1.629 s	
joinPreprocess	10659	1.032 s	1.032 s	
rankedFeatureSelected	1	0.119 s	0.000 s	
GeneratePartion	69	0.112 s	0.112 s	
setdiff	67	0.077 s	0.014 s	
num2str	100	0.077 s	0.056 s	
setdiff>setdiffR2012a	67	0.063 s	0.014 s	
unique	67	0.049 s	0.021 s	
std	50	0.035 s	0.007 s	

Appendix A4: Runtime Analysis for IFS + Multi-Reg.

Generated 19-Aug-2016 22:24:11 using *cpu* time.

Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
HybridFSMain2	1	1186.490 s	0.059 s	
ACOIFSMethod	1	840.800 s	1.159 s	
moveAnts	1000	772.449 s	404.971 s	
RouletteWheelSelection	18000000	334.538 s	334.538 s	
IFSmethod	1	192.827 s	7.818 s	
computeMI	10692	164.684 s	7.328 s	
computeEntropyofX	21384	154.124 s	4.561 s	
MultiRegmodel	1	152.804 s	0.118 s	
gradientDescentMulti	50	152.534 s	17.710 s	
simpleTrapofX	11484	141.559 s	141.559 s	
computeCostMulti	50000	134.824 s	134.824 s	
updatePherome	1000	66.735 s	66.735 s	
ACOIFSMethod>@(x)computeCost(x,model)	300000	32.940 s	11.211 s	
computeCost	300000	21.729 s	21.729 s	
featureCount	1	13.902 s	13.902 s	
pdfEstimator	11484	8.004 s	1.836 s	
copdfExponent	11484	6.168 s	3.588 s	
constr	8088	6.168 s	0.702 s	
stoppingCrit	72	6.109 s	0.093 s	
redundancyCheck	8020	5.466 s	1.489 s	
ismember	8064	3.977 s	0.635 s	
ismember>ismemberR2012a	8064	3.342 s	1.227 s	
computeJointEntropyXY	10692	3.232 s	2.048 s	
pdfKernel	11484	2.581 s	2.581 s	
ismember>ismemberBuiltinTypes	8064	2.115 s	2.115 s	
joinPreprocess	10692	1.185 s	1.185 s	
num2str	2100	0.499 s	0.296 s	

Appendix A5: Runtime Analysis IFS_BACS + Multi-Reg.

Generated 26-Jun-2016 14:43:08 using cpu time.

Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
newMain4Redundancytest	1	674.375 s	1.458 s	
rankedFeatureSelected	1	618.972 s	76.320 s	
redundancyCheck	1884341	542.912 s	90.175 s	
ismember	1884369	452.737 s	51.603 s	
ismember>ismemberR2012a	1884369	401.134 s	112.647 s	
ismember>ismemberBuiltInTypes	1884369	288.487 s	288.487 s	
computeMI	2220	42.238 s	0.796 s	
computeEntropyofX	2220	41.441 s	0.662 s	
simpleTrapofX	2220	39.110 s	39.110 s	
stoppingCrit	110	16.024 s	0.403 s	
featureCount	1	10.564 s	10.564 s	
pdfEstimator	2220	1.670 s	0.278 s	
copdfExponent	2220	1.391 s	0.921 s	
pdfKernel	2220	0.470 s	0.470 s	
GeneratePartion	110	0.393 s	0.393 s	
constr	141	0.345 s	0.058 s	
setdiff>setdiffR2012a	1	0.029 s	0.019 s	
setdiff	1	0.029 s	0.000 s	
unique	1	0.010 s	0.010 s	
unique>uniqueR2012a	1	0 s	0.000 s	

Self time is the time spent in a function excluding the time spent in its child functions. Self time also includes overhead resulting from the process of profiling.

Appendix A6: Runtime Analysis for IFS before Optimizing the Redundancy Check Function

Generated 27-Jun-2016 13:26:23 using cpu time.

Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
newMain4Redundancytest	1	40.561 s	1.350 s	
computeMl	2196	30.844 s	0.606 s	
computeEntropyofX	2196	30.238 s	0.386 s	
simpleTrapofX	2196	28.819 s	28.819 s	
stoppingCrit	108	11.273 s	0.145 s	
featureCount	1	7.589 s	7.589 s	
pdfEstimator	2196	1.033 s	0.179 s	
copdfExponent	2196	0.854 s	0.523 s	
pdfKernel	2196	0.331 s	0.331 s	
constr	133	0.289 s	0.034 s	
redundancyCheck	129	0.262 s	0.090 s	
GeneratePartion	108	0.220 s	0.220 s	
ismember	158	0.172 s	0.041 s	
ismember>ismemberR2012a	158	0.131 s	0.083 s	
rankedFeatureSelected	1	0.124 s	0.028 s	
setdiff	7	0.090 s	0.014 s	
setdiff>setdiffR2012a	7	0.076 s	0.021 s	
unique	7	0.055 s	0.028 s	
ismember>ismemberBuiltinTypes	158	0.048 s	0.048 s	
unique>uniqueR2012a	7	0.028 s	0.028 s	

Self time is the time spent in a function excluding the time spent in its child functions. Self time also includes overhead resulting from the process of profiling.

Appendix A7: Runtime Analysis for IFS after Optimizing the Redundancy Check Function

Appendix B: IFS Experimental Results

Appendix B1: Result of Comparison between IFS_MI Estimator and Other MI Benchmarks

Dpoints	Kask@K=10	INFOTHEO	ITE, AT K=3	GNN3	IFS
3.00E-08	5.462384	2.557972	6.236029879	5.953756093	5.3526
6.00E-08	5.462384	2.557972	6.236029879	5.960113718	5.3526
9.00E-08	5.46238	2.557972	6.236029879	5.948256697	5.3526
3.00E-06	5.462116	2.557972	6.236029879	5.949816205	5.3526
6.00E-06	5.462086	2.560476	6.236029879	5.962602443	5.3526
9.00E-06	5.462045	2.557972	6.236029879	5.963622746	5.3526
0.0003	5.462807	2.557972	6.236029879	5.958816586	5.3526
0.0006	5.462786	2.557972	6.236029879	5.939177901	5.3526
0.0009	5.463861	2.562428	6.236029879	5.975097659	5.3526
0.03	5.465214	2.562428	6.236029879	5.944343201	5.348
0.06	5.462536	2.557972	6.236029879	5.976744133	5.348
0.09	5.463819	2.559924	6.236029879	5.978976661	5.3526
3	5.464406	2.559924	6.236029879	5.958907114	5.3527
9	5.463696	2.559924	6.236029879	5.961394207	5.3527
81	5.463696	2.559924	6.236029879	5.961946384	5.3527

Appendix B2: IFS Results for the Controlled Experiments

No of Runs		Selected Features									
	Feature	1	2	3	4	5	6	7	8	9	10
1	% of Appearance	0	1%	93%	6%	0%	0%	0%		0%	0%
2	% of Appearance	1%	2%	88%	8%	1%	0%	0%	0%	0%	0%
3	% of Appearance	1%	2%	83%	14%	0%	0%	0%	0%	0%	0%
4	% of Appearance	1%	3%	87%	9%	0%	0%	0%	0%	0%	0%
5	% of Appearance	1%	8%	76%	15%	0%	0%	0%	0%	0%	0%
6	% of Appearance	1%	7%	82%	8%	1%	0%	0%	1%	0%	0%
7	% of Appearance	3%	1%	86%	10%	0%	0%	0%	0%	0%	0%
8	% of Appearance	2%	5%	85%	6%	1%	0%	1%	0%	0%	0%
9	% of Appearance	2%	3%	86%	9%	0%	0%	0%		0%	0%
10	% of Appearance	3%	3%	88%	6%	0%	0%	0%	0%	0%	0%
Overall	% Appearance	1.5	3.5	85.4	9.1	0.3	0	0.1	0.1	0	0

Appendix B3: Results of IFS Redundancy Check Test

100% exact copy		100% rand copy		80% rand copy	
With Red.	Red. Eliminated	With Red.	Red. Eliminated	With Red.	Red. Eliminated
10	10	10	10	4	4
4	8	8	8	3	3
8	3	3	3	8	8
3	0	5	5	5	5
0	0	9	0	9	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
60% rand copy		50% rand copy		30% rand copy	
With Red.	Red. Eliminated	With Red.	Red. Eliminated	With Red.	Red. Eliminated
4	4	4	4	4	4
8	8	9	9	5	5
3	3	8	8	3	3
9	5	3	6	9	0
0	0	6	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
20% rand copy		10% rand copy			
With Red.	Red. Eliminated	With Red.	Red. Eliminated		
4	4	4	4		
3	3	8	8		
5	5	3	3		
8	8	5	5		
10	0	0	0		
0	0	0	0		
0	0	0	0		
0	0	0	0		
0	0	0	0		

Appendix B4: IFS Performance Evaluation Results Before/After IFS is Applied for Regression Tasks

SN	Dataset	No. of features	Method type	MAD test score	NMSE test score	RMSE test score
1	Breast cancer	33 / 7	ANN	0.0228 / 0.0148	0.9877/0.98755	0.9373655/0.9151360
			Multi-Reg.	0.0202 / 0.0083	0.84/0.9028	0.86112 / 0.83727
			SVR	0.0580 / 0.0092	0.9837/0.96019	0.94684 / 0.9126
2	Community	99 / 85	ANN	0.2491 / 0.1623	1.42187 / 1.3744	0.3754896 / 0.29998
			multi-Reg.	0.00086 / 0.00085	0.6847 / 0.61398	0.1647 / 0.15112
			SVR	0.20117 / 0.17623	1.25212 / 1.1108	0.2908 / 0.26406
3	Concrete	08 / 03	ANN	0.0358 / 0.0068	0.99825 / 0.9976	0.96861 / 0.9376
			MultiReg.	0.00697 / 0.00260	0.629396 / 0.6889	0.6633 / 0.699692
			SVR	0.03868 / 0.01114	0.4371 / 0.7315	0.4407 / 0.7100
4	Energy Efficiency (cooling loads)	08 / 04	ANN	0.174905 / 0.06253	1.01088 / 0.9989	1.04952 / 0.960107
			MultiReg.	0.031416 / 0.018233	0.3414 / 0.4552	0.34761 / 0.44578
			SVR	0.031416 / 0.018233	0.1595 / 0.28315	0.16097 / 0.2781

5	Energy Efficiency (heating loads)	08 / 04	ANN	0.0279692 / 0.00411458	0.9971 / 0.99676	0.98758 / 0.943086
			MultiReg.	0.01204746 / 0.0082479	0.2876 / 0.405	0.3013 / 0.40625
			SVR	0.000352 / 0.029168	0.09357 / 0.27438	0.0917 / 0.2753
6	Forest Fires	12 / 04	ANN	0.034768 / 0.00982	0.9956 / 0.9952	1.19275 / 1.15404
			MultiReg.	0.19209 / 0.00392	0.99656 / 0.997	1.7408 / 0.59867
			SVR	0.059358 / 0.005039	1.03044 / 1.0559	1.45685 / 1.0405
7	Parkinsons (UPDRS)	19 / 12	ANN	0.0187436 / 0.01078309	0.9996 / 0.9995	1.0102 / 0.98819
			MultiReg.	0.0124293 / 0.0001494	0.824 / 0.8936	0.81612 / 0.87428
			SVR	0.00476052 / 0.052048	0.5431 / 0.621876	0.5499 / 0.6123
8	Parkinsons (Total UPDRS)	19 / 04	ANN	0.024639 / 0.009691	0.9997 / 0.99949	0.99894 / 0.973008
			MultiReg.	0.03506 / 0.0005975	0.8432 / 0.908	0.86014 / 0.8867
			SVR	0.011569 / 0.0154379	0.5422 / 0.6856	0.5419 / 0.691

Appendix B5: IFS Performance Evaluation Results Before/After IFS is Applied for Classification Tasks

S N	Dataset	No of feature s	Method type	MAD test score	NMSE test score	RMSE test score
1	Breast cancer	33 / 10	ANN	0.025 / 0	1.29452 / 0.98897	0.5244 / 0.44721
			SVM	0.0167 / 0.0167	0.9322 / 0.8421	0.387298 / 0.288675
2	Glass	09 / 04	ANN	0.4186 / 0.2558	0.65488 / 0.5927	1.430563 / 1.1513
			SVM	0.328125 / 0.171875	0.51405 / 0.43248	1.05327 / 1.023169
3	Iris	04 / 03	ANN	0 / 0	0 / 0	0 / 0
			SVM	0.044 / 0	0.24269791 / 0	0.2108 / 0
4	Breast tumor	05 / 02	ANN	0.03448 / 0.0172	0.7403 / 0.46089	0.37139 / 0.2274
			SVM	0.023 / 0.01149	0.809798 / 0.73314	0.401148 / 0.3555795
5	Wine	13 / 06	ANN	0 / 0	0 / 0	0 / 0
			SVM	0.0185 / 0.0185	0.305736 / 0.29428	0.2357 / 0.2357
6	Energy efficiency (cooling load)	08 / 04	ANN	0.11688 / 0.04545	0.3875 / 0.4672	0.426401 / 0.528413
			SVM	0.004329 / 0.045455	0.3674689 / 0.41993	0.41089 / 0.48349
7	Energy efficiency (heating load)	08 / 04	ANN	0.103896 / 0.16234	0.4041886 / 0.5151	0.4413674 / 0.5976
			SVM	0.01732 / 0.004329	0.3341 / 0.4226479	0.37219 / 0.46987

Appendix C: IFS_BACS Experimental Results

Appendix C1: IFS_BACS Performance Evaluation Results - Comparison for Classification Tasks

Dataset	NMSE test score by IFS only	NMSE test score by IFS_BACS	RMSE test score IFS only	RMSE test score by IFS_BACS
Breast cancer	0.8421	0.817662173	0.288675	0.273861279
Glass	0.43248	0.392908853	1.023169	0.876037591
Iris	0	0	0	0
Breast tumor	0.73314	0.716598572	0.3555795	0.347404167
Wine	0.29428	0	0.2357	0
Energy utilization (cooling)	0.41993	0.333564643	0.48349	0.377964473
Energy utilization (Heating)	0.4226479	0.283804268	0.46987	0.322329186

Appendix C2: IFS_BACS Performance Evaluation Results – Comparisons for Regression Tasks

Dataset	NMSE test score by IFS only	NMSE test score by IFS_BACS	RMSE test score IFS only	RMSE test score by IFS_BACS
Breast cancer	0.9028	0.756214389	0.83727	0.6907486
Community	0.61398	0.600005604	0.15112	0.133052047
Concrete	0.6889	0.590574573	0.699692	0.601823867
Energy utilization (cooling)	0.4552	0.361137146	0.44578	0.407071057
Energy utilization (Heating)	0.405	0.359253339	0.40625	0.362557269
Parkinson (motor_UPDRS)	0.8936	0.851603182	0.87428	0.856584032
Parkinson (total_UPDRS)	0.908	0.845313314	0.8867	0.852119487

Appendix C3: IFS_BACS Detail Result Comparison for Regression Tasks

Dataset	Original No. of feature	No. of feature Selected (IFS only)	No. of feature selected (IFS_BACS)	NMSE test score (IFS only)	NMSE test score (IFS_BACS)	RMSE test score (IFS only)	RMSE test score (IFS_BACS)	% Acc. (IFS only)	% Acc. IFS_BACS
Breast cancer	33	7	9	0.9028	0.756214389	0.83727	0.6907486	42.5	55
Community	99	85	22	0.61398	0.600005604	0.15112	0.133052047	90.56603774	90.56603774
concrete	8	3	7	0.6889	0.590574573	0.699692	0.601823867	46.60194175	54.36893204
Energy utilization (cooling)	8	4	3	0.4552	0.361137146	0.44578	0.407071057	72.72727273	84.41558442
Energy utilization (Heating)	8	4	3	0.405	0.359253339	0.40625	0.362557269	70.77922078	75.32467532
Parkinson (motor_UPDRS)	19	12	10	0.8936	0.851603182	0.87428	0.856584032		41.52637486
Parkinson (total_UPDRS)	19	4	12	0.908	0.845313314	0.8867	0.852119487		39.73063973

Note: the percentage prediction accuracy for regression tasks is obtained after both the target output and predicted output were rounded up to the whole number.

Appendix C4: IFS_BACS Detail Results Comparison for Classification Tasks

Dataset	Original No. of feature	No. of feature Selected (IFS only)	No. of feature selected (IFS_BACS)	NMSE test score (IFS only)	NMSE test score (IFS_BACS)	RMSE test score (IFS only)	RMSE test score (IFS_BACS)	% Acc. (IFS only)	% Acc. (IFS_BACS)
Breast cancer	33	10	10	0.8421	0.817662173	0.288675	0.273861279	91.66666667	92.5
glass	9	4	5	0.43248	0.392908853	1.023169	0.876037591	65.625	62.79069767
iris	4	3	3	0	0	0	0	100	100
Breast tumor	5	2	4	0.73314	0.716598572	0.3555795	0.347404167	87.35632184	87.93103448
Wine	13	6	5	0.29428	0	0.2357	0	94.44444444	100
Energy utilization (cooling)	8	4	7	0.41993	0.333564643	0.48349	0.377964473	76.62337662	85.71428571
Energy utilization (Heating)	8	4	7	0.4226479	0.283804268	0.46987	0.322329186	77.92207792	89.61038961

Appendix C5: IFS_BACS Prediction Accuracy Comparison for Regression Tasks

Dataset	Prediction accuracy (IFS method) only	Prediction accuracy (IFS_BACS) method
Breast cancer	42.5	55
Community	90.56603774	90.56603774
concrete	46.60194175	54.36893204
Energy utilization (cooling)	72.72727273	84.41558442
Energy utilization (Heating)	70.77922078	75.32467532
Parkinson (motor_UPDRS)		41.52637486
Parkinson(total_UPDRS)		39.73063973

Note: the percentage prediction accuracy for regression tasks is obtained after both the target output and predicted output were rounded up to the whole number.

Appendix C6: IFS_BACS Prediction Accuracy Comparison for Classification Tasks

Dataset	Prediction accuracy (IFS method) only	Prediction accuracy (IFS_BACS) method
Breast cancer	91.66666667	92.5
glass	65.625	62.79069767
iris	100	100
Breast tumor	87.35632184	87.93103448
Wine	94.44444444	100
Energy utilization (cooling)	76.62337662	85.71428571
Energy utilization (Heating)	77.92207792	89.61038961

Appendix D: Sample of Optimal Parameter Settings for IFS and Best Performing Models

Appendix D1: Sample of w and size of partition settings for UCI datasets used in this Thesis

Dataset	Samples	Features	Size of Partitions	w (in %)
Breast cancer	197	33	2	3
Mammographic mass data	289	5	2	3
Communities and Crime	530	99	5	1
Concrete	1030	8	5	3
Energy (cooling load)	768	8	5	3
Energy (heating load)	768	8	5	3
Forest fire	517	12	5	20
Glass	213	9	3	3
Iris	149	4	3	3
Parkinson (motor_updrs)	4454	19	5	20
Parkinson (total_updrs)	4454	19	5	20
Wine	177	13	3	3

Appendix D2: Sample of Optimal Parameter Settings for best performing Regression models based on ANN

dataset	Feature Set	Lambda	MaxIter	Input layer	hidden layer
Bcancer	Selected	0.2	300	7	14
	Full	0.836	300	33	66
Communittee	Selected	-8270	300	85	170
	Full	-0.022	300	99	198
Concrete	Selected	0.2	300	3	6
	Full	0.2	300	8	16
Energy Utilization (cooling)	Selected	0.298	300	4	8
	Full	-0.596	300	8	16
Energy Utilization (heating)	Selected	-0.27	300	4	8
	Full	0.2	300	8	16
Forestfire	Selected	0.7	300	4	8
	Full	0.7	300	12	24
Perkison_UPDRS	Selected	0.5	300	12	24
	Full	0.1	300	19	38
Parkison_total_udpr	Selected	0.072	300	4	8
	Full	0.5	300	19	38

Appendix D3: Comparison of the Performance Rating of the Proposed IFS Method and Five Feature Selection Benchmarks

Performance rating	fsInfoGain	fsKruskal	fsMRMR	fsIFS	fsCFS	fsFCBF
Best	0	0	0	2	0	2
Jointly Best	3	3	0	2	2	2
Worst	0	2	1	0	2	0
Jointly Worst	0	0	2	1	1	1