Some Issues in Unsupervised Feature Selection Using Similarity

Partha Pratim Kundu

Thesis submitted to the Indian Statistical Institute
in partial fulfillment of the requirements
for the award of
Doctor of Philosophy.
August, 2014

Thesis supervisor: Prof. Sushmita Mitra

Indian Statistical Institute
203 B. T. Road, Kolkata - 700108, India.

To My Parents

Ranjita Kundu & Ganga Prosad Kundu

Acknowledgement

At first I would like to take this opportunity to express my sincere gratitude towards my supervisor, Prof Sushmita Mitra for her continuous support and encouragement during my Ph.D journey. Prof. Mitra has always allowed me complete freedom to define and explore my own directions in research. While this proved difficult and somewhat bewildering to begin with, I have come to appreciate the wisdom of her way – it encouraged me to think for myself.

Heartfelt thanks are due to Prof. C. A. Murthy for his great teaching skills. His expositions of several topics in mathematics, statistics and computer science will be of immense value to me throughout my research career. My indebtedness to Prof. Mandar Mitra is also beyond my words. His suggestions and moral encouragements helped me to overcome pitfalls during my journey of writing this thesis. I am also thankful to Prof. Witold Pedrycz for his support in my early Ph.D career.

My sincere thanks go to my friends and colleagues of ASU, CVPR, CSCR and MIU for their all round support during the period of my doctoral studies. I also thank the office staff of MIU for efficiently handling administrative issues. I acknowledge the CSSC for providing computing facilities, the ISI library for providing reference materials, and the authorities of ISI for extending various facilities.

I shall forever remain indebted to my parents for their constant encouragement, enthusiasm and support that has helped me throughout my academic career and specially during the research work.

ISI KOLKATA PARTHA PRATIM KUNDU

2014

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

Pattern recognition [9, 24, 28, 57, 61] is what humans do most of the time, without any conscious effort, and fortunately excel in. Information is received through various sensory organs, processed simultaneously in the brain, and its source is instantaneously identified without any perceptible effort. The interesting issue is that recognition occurs even under non-ideal conditions, *i.e.*, when information is vague, imprecise or incomplete. In reality, most human activities depend on the success in performing various pattern recognition tasks. Let us consider an example. Before boarding a train or bus, we first select the appropriate one by identifying either the route number or its destination on the basis of the visual signals received by the brain; this information is then speedily processed, followed by neurobiological implementation of template-matching [28].

The discipline of pattern recognition (PR) centers around the development of algorithms and methodologies/devices which enable automated implementation of various recognition tasks normally performed by humans. The motivation is to perform these tasks more accurately and/or faster, perhaps, in a more economical manner; and, in many cases, to relieve humans from the mundane activity of mechanically performing such routine recognition tasks. The scope of PR also encompasses tasks in which humans are not particularly good, like reading Quick Response codes. The goal of pattern recognition research is to prepare mechanisms to automate certain decision making processes that lead to classification and recognition. Though research in this domain has attained maturity over the past decades, it remains fertile to researchers due to the continuous interaction with other disciplines including biology, artificial intelligence, information theory, psychology and cognitive science. As a result, depending on the practical needs and demand, various applications like video surveillance, image retrieval, social media mining, have been initiated in order to supplement the classical techniques [24, 28].

The field of machine learning is concerned with the question of how to construct programs that gradually and automatically improve with experience. In recent years many successful machine learning applications have been developed, encompassing algorithms that learn to detect fraudulent financial transactions, information-filtering systems that learn users' reading preferences, and autonomous vehicles that learn to drive on public highways. Typically machine learning involves searching a very large space of possible hypotheses to determine the one that best fits the observed data, along with any prior domain knowledge. The learner's task is to search through the vast space of solutions, determined by the available evaluation functions, in order to locate the most consistent hypothesis [9, 82].

Over the last several years the availability of the internet and the decrease in cost of storage have resulted in databases become voluminous and, in some cases, heterogeneous. Such massive datasets generally consist of a combination of numeric, textual, symbolic, pictorial, video, as well as aural data. There may also be embedded a certain amount of redundancy, error, imprecision, etc. Traditional data cleaning, statistical data summarization and data management techniques are often just not adequate for handling such multimedia data [84]. This is where we need data mining in order to intelligently extract information or knowledge, that may be useful for exploring the domain in consideration and to provide support towards decision making.

Data mining [34, 47, 84] can be viewed as an integration of PR and machine learning in the context of large data. Here stress is more on the scalability of the number of features and instances, where scalability refers to the ability of an algorithm to efficiently handle large volumes of data. In effect data mining involves a multidisciplinary effort from the database, machine learning and statistics communities. Some of the major functionalities of data mining include association rule mining, clustering, classification, regression, sequence analysis, dimensionality reduction, rule generation, summarization or condensation. Data mining algorithms determine both the flexibility of a model in representing the data as well as its interpretability in human terms. Although a more complex model may fit the data in a better manner, often it may also be more difficult to understand [62, 84]. This pertains to the issues of generalization and overfitting.

There can exist various kinds of imperfection in the input data, mainly due to uncertainty, vagueness, and incompleteness. While incompleteness arises due to missing or unknown data, uncertainty (or vagueness) can be caused by errors in physical measurements due to incorrect measuring devices or the mixing of noisy and pure signals. Soft computing techniques are capable of effectively handling these issues. Soft computing is a consortium of paradigms like fuzzy sets, neural networks, and genetic algorithms, that work synergistically to provide flexible information processing capabilities for real life problems. Its aim is to exploit the tolerance for imprecision, uncertainty, approximate reasoning and partial truth in order to achieve tractability, robustness, low cost solution and close resemblance to human-like decision making [93]. The use of soft computing in pattern recognition and is reported in literature [84, 93].

With the discovery and/or growth of high-throughput technologies, like hyper-spectral imagery, radio frequency ID, high speed internet, and smart metering, there has been an asymptotic increase in the dimensionality and size of databases. As a result their storage and processing have become more challenging, with manual processing becoming impractical. Therefore, techniques integrating data mining and machine learning are being developed in order to efficiently automate the pattern recognition and knowledge discovery process. However application of these algorithms directly on raw data is mostly useless due to the high level of noise and redundancy associated with the samples. Noise usually comes from imperfection during data collection or from the source of the data itself. Redundancy may be incorporated during measurement of the same variable over different instances. Extracting nuggets of knowledge from such huge and noisy datasets is thus a difficult task, and data preprocessing is a necessary step towards achieving this goal [76, 78]. Feature selection plays a major role in this direction.

The objective of this thesis is to present development and design of some algorithms, along with their case studies, involving both theoretical and experimental studies in unsupervised feature selection. Extension to large data is also investigated, with a view to reducing the curse of dimensionality. Novel similarity measures, from statistical, classical, and soft computing domains, are introduced to identify reduced subsets of informative features. The similarity is mainly based on various internal characteristics of the data.

Before outlining the scope of the thesis, we provide a brief introduction to pattern recognition, data mining and soft computing. The rest of this chapter is organized as follows. Section 1.2 introduces the pattern recognition and data mining. Next we present genetic algorithms and its various constituents in Section 1.3. A short study of feature selection is provided in Section 1.4. The role of similarities, in clustering and feature selection, is highlighted in Section 1.4.3. Finally Section 1.5 deals with the scope of the thesis.

1.2 Pattern Recognition

Pattern recognition (PR) can be viewed as a two-fold task, consisting of learning the invariant and common properties of a set of samples (or patterns) characterizing a class or group, and of deciding an unknown pattern to be the possible member of a group by noting that it has properties common to those of its set of samples. PR can thus be described as a transformation from the measurement space MS to feature space FS, and finally to the decision space DS as

$$MS \rightarrow FS \rightarrow DS, \tag{1.1}$$

where $\rightarrow$ denotes a mapping from one space to another.

Patterns can be represented by arrays of numbers or characters obtained from a sequence of binary or logical tests, scanning of images, reading of texts, or acquiring information from any relevant source. Pattern classes can be depicted by one or several prototype patterns. A typical PR system consists of three phases namely, (i) data acquisition, (ii) feature selection or extraction, and (iii) decision making, i.e. classification or clustering. Its aim is to achieve robustness with respect to random noise, and to obtain output in real time. It is also desirable for the system to be adaptive to changes in environment [28].

Data is first gathered with a set of sensors during the data acquisition phase, depending on the environment within which patterns are to be classified or clustered. This is then passed on to the feature selection or extraction phase, where its dimensionality is reduced by either retaining a few characteristic features or properties or by mapping the information content into a space whose basis or features are formed using the characteristic features of the original data. In a broader perspective, this stage significantly influences the entire recognition process. Finally, the classification or clustering phase evaluates the information present among the selected or extracted features for learning a final decision. This phase basically establishes a transformation between the input features and the output clusters or classes [24, 28].

Learning can be broadly categorized into three categories, viz. supervised, semi-supervised and unsupervised. In supervised learning, the algorithm generates a learner by analyzing a training set made up of database tuples and their associated class labels. In the testing phase, the algorithm predicts the class labels of samples which it has not encountered during training. Generalization and scalability are two important properties of any learner. The generalization capability is estimated based on the performance over an unknown test set. Overfitting exists when a model is extensively complex, such as having too many parameters relative to the number of observations, and describes random error or noise instead of focusing on the underlying relationship. Underfitting occurs when a model is too simple, and is not flexible enough to capture the underlying trends in the observed data. Scalability refers to the ability to construct a learner or predictor efficiently, in the presence of a large set of data. Scalable approaches are thus capable of handling training data that are too large to fit in memory.

Supervised learning is also termed classification. It contrasts with unsupervised learning or clustering, in which neither the class label of a sample nor the total number of labels to be learned are available. In semi-supervised learning, on the other hand, partial class information of the training samples may be known in advance. In the following sections we describe classification and clustering in further detail, before moving on to large data.

1.2.1 Classification

A classifier partitions a feature space into regions, by assigning each input pattern to one of the possible output classes based on certain parameters. In real life, since most of these parameters may not be known a priori, they need to be estimated from a finite set of input patterns. This finite set of samples, which often provides partial information for the optimal design of a pattern recognition system, is termed the training set.

There are several approaches to classifier design. These include decision theoretic (both deterministic and probabilistic) approaches, connectionist approaches, and support vector machines, among others. A good classifier should possess characteristics like on-line adaptation, nonlinear separability, capability of handling overlapping classes, fast decision making and minimization of the number of tunable parameters in the system. Some of the well-known classifiers are outlined below.

k-nearest neighbors (k-NN)

Given a test point x_t, the k training points which are closest to x_t in terms of distance are identified. Then x_t is classified using majority voting among these k nearest neighbors. Ties are resolved arbitrarily [28]. The k-NN classifier is used in Sections 2.3, 3.4 and 4.4.

Discriminant analysis

The procedure attempts to determine several discriminant functions (linear combination of independent variables) that discriminates among the groups defined by the response variable [47].

Naïve Bayes (NB)

This is a probabilistic approach. In the Naïve Bayes (NB) [24] setting, the naive assumption of class conditional independence is made. Here the values of attributes of a sample are conditionally independent of one another, given the class label of the sample. In other words, there exists no dependence relationship among the attributes or features [47]. The NB classier is used in Sections 2.3, 3.4, 3.4.3 and 4.4.

Decision tree

A decision tree classifier uses, in most cases, an information theoretic measure, like entropy, for assessing the discriminating power of each attribute. A few important decision tree algorithms are Interactive Dichotomizer 3 (ID3), Classification and Regression Tree (CART), C4.5/C5.0, RainForest [84].

Support Vector Machine (SVM)

The SVM classifier is based on hyperplane learning. The idea is to map the training data into a higher dimensional feature space via a mapping function, and to construct a separating hyperplane with maximum margin. This yields a linear or nonlinear decision boundary in the input space. Using a kernel function k, it is possible to compute the separating hyperplane without explicitly mapping into the feature space. We have used SVM in Sections 2.3, 3.4.3 and 4.4, for classification.

1.2.2 Clustering

A cluster is comprised of a number of similar objects collected or grouped together [56]. It may be described as an aggregation of points in a test space, such that the within-cluster distance between any two points in a cluster is less than the between-cluster distance between any pair of points in different clusters. It can also be represented as connected regions in a multi-dimensional space, containing a relatively high density of points separated from other such dense regions by a region containing a relatively lower density of points [32]. The process of clustering usually consists of three steps. (1) Define a measure of dissimilarity or similarity between the objects or patterns. (2) Formulate an objective function for clustering given patterns or objects. (3) Design a methodology for obtaining the cluster satisfying the objective. Broadly clustering algorithms can be categorized into partitive, hierarchical and density-based approaches.

Given a database of objects or data tuples, a partitive method constructs partitions of data with each partition representing a cluster around a centroid. The most well-known member of this family is the k-means algorithm [24]. Although partitive algorithms are less expensive, in terms of time and space, yet the number of clusters need to be specified apriori. Moreover, the cluster structure is dependent on the choice of seed points. Another variant of conventional k-means algorithm is the Iterative Self-Organizing Data Analysis Technique (ISODATA), which employs splitting and merging operations on clusters based on a threshold [84]. Some other examples include Partitioning Around Medoids (PAM) and k modes [47, 84].

A hierarchical method creates a hierarchical decomposition of the given set of objects, and can be grouped as agglomerative or divisive depending on whether the process is bottom-up or top-down. A major weakness of these methods involves poor scalability, quadratic time complexity, and sensitivity to outliers. However the cluster structure remains the same over repeated executions. Popular algorithms of this category are single linkage, complete linkage, and Divisive Analysis (DIANA) clustering [47, 84].

The general idea of density-based methods is to continue growing a cluster, around a seed point, as long as the density of patterns in its neighborhood is above a user-defined threshold. The neighborhood region of each pattern in a cluster, within a user-defined radius, must contain a given minimum number of points (as defined by the density threshold). The major characteristics of density-based methods include the ability to effectively (i) discover clusters of arbitrary shape (convex and non-convex) and (ii) handle noise. But they are, generally, computationally more expensive than partitive methods. Few important examples are Density-Based Spatial Clustering of Applications with Noise (DBSCAN) and Ordering Points To Identify Clustering Structure (OPTICS) [47, 84].

1.2.3 Dimensionality reduction

Dimensionality reduction is an important preprocessing technique to remove noisy, irrelevant and redundant features or attributes from the data. It includes feature extraction and feature selection. Feature extraction involves the projection of data into a new transformed space of lower dimensionality, such that the attributes in this transformed space consist of linear or non-linear weighted combination of features from the original space. Examples of feature extraction techniques include Principle Component Analysis (PCA), Linear Discriminant Analysis (LDA), and Singular Value Decomposition (SVD) [9, 28].

On the contrary, feature selection approaches select subset(s) of features from the original space for maximizing their relevance to the target. Such selected features should also have minimum redundancy among themselves. Popular feature selection techniques include Sequential Forward Selection (SFS) [24], Sequential Backward Search (SBS) [24], Sequential Floating Forward Search (SFFS) [96], Step-Wise Clustering (SWC) [64], Information Gain [120], ReliefF [65], Chi Squares [120] and Minimal-Redundancy-Maximal-Relevance criterion (mRMR) [26].

Both these dimensionality reduction approaches improve learning performance, reduce computational complexity, build better generalizable models and decrease required storage space. However feature selection is superior in terms of improved understandability and interpretability, since it preserves the original feature values in the reduced space. Feature extraction, on the other hand, projects the feature values into a transformed space of lower dimension. Therefore, further semantic analysis in the new space becomes difficult as often no physical meaning can be assigned to the transformed features. Here we describe two of the well-known algorithms for feature extraction and selection.

Principle Component Analysis (PCA)

This is a popular technique for feature extraction technique [9], and is outlined below.

1. Compute $\vec{X}\vec{X}^T = \sum_{i=1}^{D} x_i x_i^T$ where D is the original dimension of the data. Let U be the eigenvectors of $\vec{X}\vec{X}^T$, corresponding to the top d eigenvalues.
2. Encode original data in $\vec{Y} = U^T\vec{X}$, where $\vec{Y}$ is a $d \times D$ matrix.
3. Reconstruct original data in the d dimensional space by $\vec{Z} = U\vec{Y} = UU^T\vec{X}$.

Sequential Forward Selection (SFS)

This is a suboptimal search procedure where one feature is added at a time to the current feature stage. The feature to be included in the feature set is selected, at each stage, from among the remaining available features. Thereby, the new enlarged feature set yields a maximum value of the criterion function used.

Let f_k be the set consisting of k already-selected features. Let $\xi_0 \in \{\vec{X} - f_k\}$ be the feature selected now, such that

$$\mathcal{F}(f_k \cup \xi_0) \geq \mathcal{F}(f_k \cup \xi);\;\; \forall \xi \in \{\vec{X} - f_k\}, \tag{1.2}$$

where $\mathcal{F}$ is the objective function to be maximized.

If ξ_0 satisfies eqn. (1.2), then $f_{k+1} \leftarrow \{f_k \cup \xi_0\}$. The feature selection method starts with $f_0 = \phi$ and ends after the desired d number of features are obtained [24].

The algorithm SFFS is a near-optimal SFS with provision for backtracking. SWC, on the other hand, is not a search-based algorithm which obtains a reduced subset by discarding correlated features.

This thesis deals with the development of four novel algorithms for feature selection. Therefore the concept of feature selection is elaborated in further detail in Section 1.4.

1.2.4 Extension to large data

Large data constitutes patterns having high dimension and/or size [47]. Handling such data involves extension of basic pattern recognition strategies in a scalable manner. Modern day research in data mining tries to achieve these goals [84]. Data mining is the non-trivial process of identifying valid, novel, potentially useful and ultimately understandable patterns in data [34]. Typically, it involves fitting models or determining patterns from available samples or objects. Data mining algorithms constitute some combination of 1) the model which contains parameters that are to be determined from the data, 2) the preference criterion which is usually some form of goodness-of-fit function of the model to the data, sometimes tempered by a smoothing term to avoid overfitting, and 3) the search algorithm [34].

The aim of data mining is to develop a unified framework which should be able to describe the probabilistic nature of the discovered patterns and models, be able to handle inductive generalizations of the data, accept different forms of data (viz. relational, sequential, textual, web) and recognize the interactive and iterative processes, with the comprehensibility of the discovered knowledge being of utmost importance. PR and machine learning algorithms seem to be the most suitable candidates for addressing these tasks [80, 98]. However, PR and data mining are not equivalent considering their original definitions. Development of new generation PR algorithms is expected to encompass more massive data sets involving diverse sources and types of data that will support mixed-initiative data mining, where human experts collaborate with the computer to form the hypotheses and test them. It should have capability to reduce the effect of spurious data points which misleads to overfit the model design [63].

Data mining, thus, is an attempt to make sense of the information explosion embedded in large volume of data. Its tasks are mainly of two types, viz. (i) descriptive, when it discovers interesting patterns or rules from the data, and (ii) predictive, when it predicts or classifies the behavior of the model based on available data. It uses automated tools that employ sophisticated algorithms to discover mainly hidden patterns, associations, anomalies, and/or structure from large amounts of data stored in data warehouses or other information repositories, by filtering necessary information from the dataset. It strives to develop architecture of an algorithm in such a way that it can be scalable in terms of the large numbers of features and instances [84].

Classification of large data is achieved by using decision trees like Serial PaRallelizable INduction of decision Trees (SPRINT) [84], support vector machines [9], neural networks [48], etc. Some popular clustering algorithms for handling large data are approximate kernel K-means [15], Balanced Iterative Reducing and Clustering Using Hierarchies (BIRCH) [84], spectral clustering [79], Clustering Large Applications based on RANdomized Search (CLARANS), Clustering Using Representatives (CURE), and Clustering in QUEest (CLIQUE) [84].

Big data is a popular term used to describe the exponential growth and availability of data, both structured and unstructured. It is important to business and society because, with the internet, the availability of more data leads to more accurate analyses and subsequently to better decision making. Eventually, it helps to achieve greater operational efficiencies, cost reductions and reduced risk. Big data has been used to convey all sorts of ideas, involving huge quantities of data, social media analytics, next generation data management capabilities, real-time data, and much more.

1.3 Genetic algorithms

Genetic algorithms (GAs) [107], based on powerful metaphors from the natural world, mimic some of the processes observed in natural selection and evolution, like selection, cross-over and mutation, towards stepwise optimization of mathematical problems. Since GAs consider multiple points in the search space simultaneously, they have less chance of converging to local optima. Thereby GAs offer a highly parallel, robust and adaptive search process, which generally leads to approximately global solutions guided by some heuristic function. GAs have been found to provide near optimal solutions to complex optimization problem in varied fields like operations research, VLSI design, pattern recognition, and machine learning [40, 84]. The design of the heuristic objective function can be of two types, viz. single objective and multi-objective, as described here.

1.3.1 Single objective GA (SGA)

SGAs, while simultaneously considering multiple solutions, use only one fitness function to provide a near optimal solution. A possible solution is encoded by a binary string, *i.e.* a finite set of '0' and '1' bits, and is called a chromosome. The length L of this string depends on the problem at hand, with the different subsets of bits being mapped to their corresponding domains. Increasing the length of a chromosome leads to high precision of the encoded variables. A collection of S_t such strings or chromosomes is called a population.

GAs typically start with a randomly generated population of size S_t. At every iteration, each chromosome of the population is evaluated in terms of a fitness function F signifying the suitability of the string (or solution) towards a given problem. A new population of the same size is produced in the next generation, using three basic operations viz. selection, crossover and mutation, on each chromosome. Since L and S_t are finite, therefore the number of possible populations is also finite. GAs are generally executed for a fixed number of generations, or terminated when there occurs no further improvement in the generated population over a certain number of iterations. In case of the elitist models, the knowledge about the best chromosome (generated so far) is preserved so that the population retains the good solutions. Typically the worst string of the offspring population gets replaced by the best string of the parent population.

The schematic diagram of the basic structure of an elitist GA model is provided in Fig. 1.1.

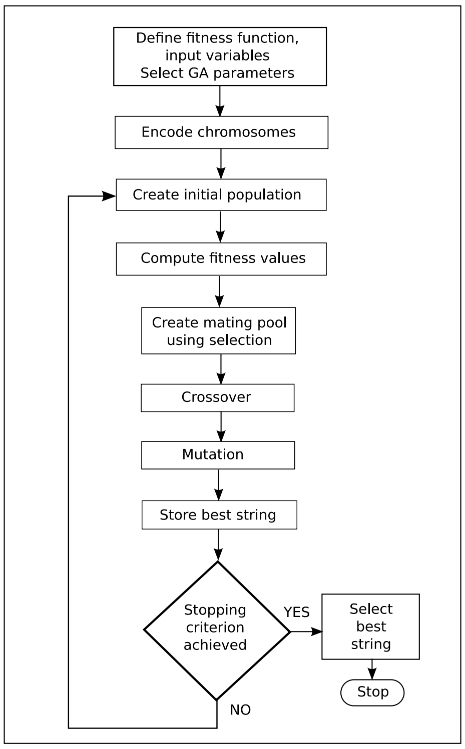

Define fitness function, input variables Select GA parameters
Encode chromosomes
Create initial population
Compute fitness values
Create mating pool using selection
Crossover
Mutation
Store best string
Stopping criterion achieved
YES
Select best string
Stop
NO

Figure 1.1: Basic steps of an elitist GA model

A given feature subset is typically represented in a GA framework as a binary string, also called as chromosome, with a “0” or “1” in position k specifying the absence or presence of the k-th feature in the set. The length of the chromosome is equal to the total number of available features in the data. It represents a prospective solution of the problem in hand, and a population of such chromosomes is evaluated by optimizing an objective function in order to enhance its fitness. GA proceeds to find a fit set of individuals (here, feature subsets) by reproducing new children chromosomes from older parents. In the process it employs the operators selection, crossover (where parts of two parent chromosomes are mixed to create an offspring) and mutation (where bit(s) of a single parent are randomly perturbed to create an offspring). Crossover probability p_c and mutation probability p_m are used. This repeats over multiple generations (or iterations) until a certain fitness level is achieved. The chromosome with the best fitness value is decoded to obtain the best feature subset.

1.3.2 Multi-objective optimization and GAs

Multi-objective optimization [16] trades off between a vector of objective functions

$$\vec{F}(\vec{x}) = F_1(\vec{x}), F_2(\vec{x}), \ldots, F_M(\vec{x}), \tag{1.3}$$

where M is number of objectives and $\vec{x}(\in \mathcal{R}^n)$ is a pattern vector of n decision variables. Unlike single objective optimization, here we try to optimize two or more conflicting characteristics represented by multiple objective functions. Modeling this situation in a single objective framework would amount to a heuristic determination of a number of parameters involved in expressing such a scalar-combination-type objective function. The multi-objective technique, on the other hand, is concerned with the simultaneous minimization or maximization of a vector of objectives $\vec{F}(\vec{x})$ that can be subject to a number of constraints or bounds. In other words, we have

$$\begin{aligned} \text{Minimize (or Maximize)} \quad & \vec{F}(\vec{x}) \qquad (1.4) \\ \text{subject to} \quad g_i(\vec{x}) \leq 0, \quad & i = 1, 2, \ldots, I; \\ h_k(\vec{x}) = 0, \quad & k = 1, 2, \ldots, K; \\ x_j^L \leq x_j \leq x_j^U, \quad & j = 1, 2, \ldots, n; \end{aligned}$$

where I and K are the inequality and equality constraints respectively. Each decision variable x_j takes a value within lower bound x_j^L and upper bound x_j^U, with the bounds constituting a decision variable space $\mathcal{D}$. The solution set of $\vec{x}$ that satisfies *all* $(I + K)$ constraints and *all* $2n$ variable bounds, forms the feasible solution space Ω. As these objective functions are competing with each other, there is no unique solution to this technique. Instead, the concept of nondominance [22] (also called Pareto optimality [13]) must be used to characterize the objectives. The objective function space Λ is defined as $\Lambda = f' \in \mathcal{R}^m$, where $f' = \vec{F}(\vec{x})_{\vec{x}\in\Omega}$. A mapping from the feasible solutions space into the objective function space, in two dimensions, is depicted in Fig. 1.2.

Multi-objective genetic algorithm (MOGA) simultaneously deals with such multiple conflicting objective functions to yield a family of solutions, which are not comparable. Each solution is equally good and can not be completely ordered with respect to the functions. While the goal of a single objective problem is to find the best solution from the solution space, the multi-objective framework optimizes several objectives to generate a set of solutions by making compromise in performance over all the concerned objectives [1, 22, 67]. Such a family of solutions is called the *Pareto Optimal Front* [13, 67], and contains those elements of a solution space which can not be simultaneously improved with respect to all the competing objectives under consideration.

The concept of optimality, in multi-objective optimization, deals with a set of solutions. The conditions for a solution to be *dominated* with respect to the other solutions are outlined here. A solution $\vec{x}^{(1)}$is said to dominate the other solution $\vec{x}^{(2)}$ if the following two conditions are true [22]:

1. The solution $\vec{x}^{(1)}$ is *no worse* than $\vec{x}^{(2)}$ in all M objectives, *i.e.*
$F_i(\vec{x}^{(1)}) \ntriangleright F_i(\vec{x}^{(2)})\ \forall i = 1, 2, \ldots M$.
2. The solution $\vec{x}^{(1)}$ is *strictly better* than $\vec{x}^{(2)}$ in *at least one* of the M objectives, *i.e.*
$F_{\bar{i}}(\vec{x}^{(1)}) \lhd F_{\bar{i}}(\vec{x}^{(2)})$ for at least one $\bar{i} \in \{1, 2, ...M\}$.

If any of the above conditions is not satisfied, then the solution $\vec{x}^{(1)}$ does not dominate the solution $\vec{x}^{(2)}$. So, the solution $\vec{x}^{(1)}$ and $\vec{x}^{(2)}$ form Pareto optimal front of these objective functions. A typical Pareto optimal front over two objective functions is shown in Fig. 1.3. Here we simultaneously optimize the conflicting requirements of the multiple objective functions. Multi-objective genetic algorithms (MOGAs) may thus be used as a tool for multi-objective optimization.

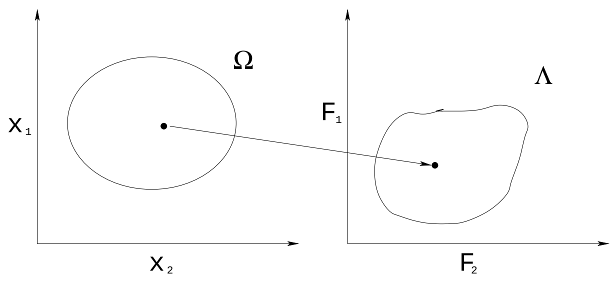


Figure 1.2: Mapping from feasible solutions space into objective function space

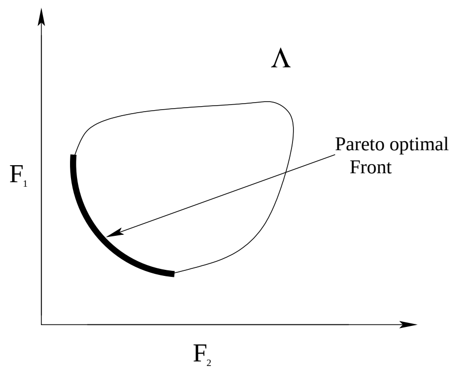


Figure 1.3: Pareto optimal front or non-dominated solutions of F_1 and F_2

The aim of MOGA is to converge to an archive which is a subset of Pareto optimal solutions and consist of diverse set of strings from the objective functions space. During the execution process a subset of the Pareto front, with respect to the present population, is created at each generation. In general, MOGAs consider two primary issues.

1) Selection of non-dominated solutions are preferred over dominated ones.

2) Good spread of solutions is maintained in a population, so that the archive represents (as close as possible) the true Pareto optimal set.

In this thesis we have used the Non-dominated Sorting Genetic Algorithm (NSGA-II), that converges to the global Pareto front while simultaneously maintaining the diversity of a population [22], for traversing the feature space (in Sections 2.2.3 and 3.3.3).

1.4 Feature Selection

In continuation to the discussion in Section 1.2.3, we recall that feature selection is a commonly used preprocessing technique for reducing high-dimensional data [77]. It helps to select a subset of attributes or features, from the original feature space, that can be used to construct a model describing a dataset. Its objectives encompass (i) reducing dimensionality, (ii) eliminating noisy, irrelevant and redundant features, (iii) reducing the amount of data needed for learning, (iv) improving the mining performance of algorithms, in terms of measures like predictive accuracy, and (v) enhancing the comprehensibility of constructed models [76, 78]. Feature selection has been widely applied to many fields such as pattern recognition [58, 83], text categorization [36, 42, 90, 120], image retrieval [21], stock market analysis [50], wireless sensor network analysis [2], face recognition [122], customer relationship management [89], intrusion detection [74], genomic analysis [3, 118] and social media analysis [112, 113].

1.4.1 Overview

Rapid development in computer engineering enabled collection of data at an unprecedented rate, thereby presenting new challenges to feature selection in ultrahigh dimensional data domains [33], stream data [39], multi-task data [91], multi-source data [124, 125] and high dimensional multi-view data [112]. Absence of class labels require unsupervised feature selection. Some strategies explore the intrinsic domain-dependent properties of datasets using statistics or information theory. In other words, there exists no alternative to good feature selection as a preprocessing strategy for any decision making task. As such, there is an ongoing volume of research [76, 78, 112] towards developing robust feature selection algorithms. We address some of these issues in this thesis.

The selected features are typically evaluated in terms of their performance in decision making. Feature selection algorithm thus constitute three steps, namely, feature subset generation, subset evaluation, and stopping criteria, as summarized in Fig. 1.4. Subset generation chooses feature subsets from the original feature space, based on certain search strategies. The evaluation criterion is used to judge the relevance of a selected subset of features. It may require either labeled or unlabeled data during the evaluation. A supervised feature selection method [95, 102, 117] determines feature relevance by computing the correlation or dependence of the subset with the class label, or by estimating its capability to predict the class labels in the dataset. In the absence of such labels, an unsupervised feature selection method [19, 30, 60] utilizes several internal characteristics of the data, like variance, distribution, or preservation of sample similarity, in order to evaluate the relevance of the selected subset. Research in feature selection is currently getting focused towards unsupervised learning [76, 78].

Feature selection strategies can be broadly categorized into filter, wrapper and embedded models, based on the degree of involvement of the learning algorithm in the evaluation criterion. Filter models do not utilize any particular learning algorithm during the feature subset evaluation process. Here the subset selection totally depends on the characteristics of the dataset as well as the class labels of samples (when available). Well-known filter algorithms include Information Gain, ReliefF, Fast Correlation-Based Filter (FCBF), mRMR, feature dependency, entropy-based distance and Laplacian score [77]. The wrapper mod-

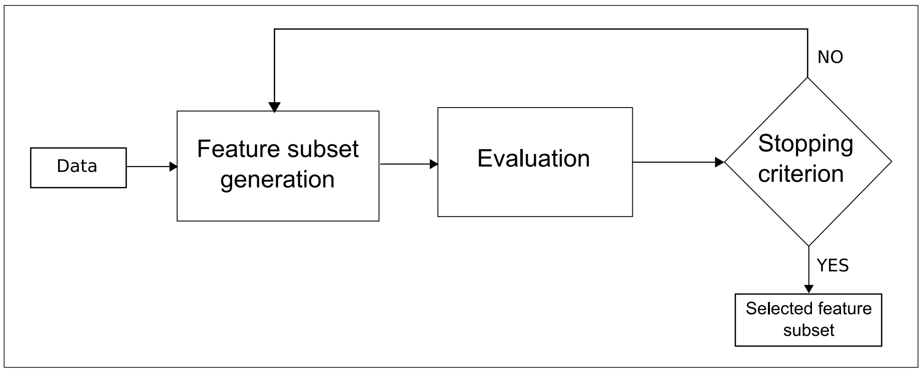


Figure 1.4: Typical view of a feature selection model

els, on the other hand, use a predetermined learning algorithm to compute the relevance of the features in a dataset [66, 77, 115]. Wrappers also tend to be more expensive than filters, from the aspects of both time and computational complexity [66, 73].Popular wrapper algorithms are Recursive Feature Elimination Support Vector Machine (RFE-SVM) [45], and Feature Subset Selection Wrapped around EM Clustering (FSSWEM) [29]. The embedded models incorporate feature selection as a part of their training schedule, while computing the relevance of features in terms of their effectiveness in optimizing a criterion function [77, 128].

Finding an optimal feature subset, based on a criterion, is usually intractable [66] for large number of features. This is because exhaustively traversing the entire search space is NP-hard in nature [10]. Therefore researchers use sequential, incremental, or random search strategies to generate feature subsets for high-dimensional data [78]. A complete search traverses the total search space of a D-dimensional data, while evaluating all 2^D feature subset combinations. Thereby, it is guaranteed to find an optimal feature set. However, a search need not be exhaustive in order to guarantee completeness. Heuristic functions have been introduced to minimize the size of the search space, without jeopardizing the chances of finding the optimal result. Random search either starts with a randomly selected subset for performing sequential search, or proceeds to generate the next subset in a completely random manner.

The use of soft computing is an interesting proposition along this direction [5, 108], in order to arrive at an acceptable solution at a lower cost. This is of particular interest towards the efficient mining and analysis of large data. We can utilize the uncertainty handling capacity of fuzzy sets and the search potential of genetic algorithms for efficiently traversing large search spaces [84].

Since the problem of feature selection involves an exponential search space, therefore GAs become naturally applicable [100, 119] due to their heuristic nature. The use of GA in feature selection already exists in literature [12, 58, 106, 107], where it is employed as an optimization technique to select a minimal set of features. Feature subsets were selected [106] using GA, involving an objective function based on the capability of preserving the correspondence between pairwise inter-pattern distances (relative to the original feature set) in terms of Sammon's stress function. Here GA was used to randomly traverse the feature subset space.

When there occur two or more conflicting characteristics to be optimized, often the single objective optimization function requires an appropriate formulation in terms of an additive combination of the different criteria involved. In such cases multi-objective optimization becomes more appropriate. Feature selection can be formulated as a minimization of the number of features and maximization of the information content in unsupervised learning (or predictive accuracy in supervised learning) over the selected subset. MOGAs were employed [5] over a population of candidate strings to select multiple non-dominated solutions representing strings of feature subsets. We have used both single objective and multi-objective GA for evaluating the fitness of a population of encoded chromosomes for feature selection in Sections 2.2.3, 3.3 and 3.3.3.

1.4.2 Evaluation of subspaces

The partitioning in different feature subspaces is evaluated both internally and externally. While the external measures compare the resultant partitioning with the correct classification of the (known) data, the internal measures compute a relationship involving the inter- and intra-cluster separability. There exist many measures of this type in literature [37, 38, 54, 56, 104]. Some of these are discussed below, and are used in Sections 2.3, 3.4, and 3.4.3.

The **Silhouette statistic** [103] offers a way of internally validating the generated clusters. Though computationally more intensive, it is another way of estimating the number of clusters in a distribution. The Silhouette index, S, computes for each point a width depending on its membership in any cluster. This silhouette width is then an average over all observations. This is expressed as

$$S_k = \frac{1}{N_k} \sum_{i:\vec{x}_i \in U_k} \frac{b_i - a_i}{\max(a_i, b_i)}, \tag{1.5}$$

where N_k is the total number of points of cluster U_k, a_i is the average distance between pattern $\vec{x}_i$ and all other points in its own cluster U_k , and b_i is the minimum of the average dissimilarities between $\vec{x}_i$ and patterns in other clusters. Finally, the global silhouette index, S, of the clustering is given by

$$S = \frac{1}{k} \sum_{j=1}^{k} S_j. \tag{1.6}$$

The partition with the highest value of S is considered to be optimal.

Redundancy rate (RED**)** assesses the average linear correlation among all feature pairs in a subset of $\mathcal{G}$ features, and is measured as [127]

$$RED(\mathcal{G}) = \frac{1}{d(d-1)} \sum_{f_i, f_j \in F, i>j} \rho_{i,j}. \tag{1.7}$$

Here $\rho_{i,j}$ is the Pearson correlation between feature pairs f_i and f_j, and the cardinality of $\mathcal{G}$ is d. A larger value of this measure indicates that more features are strongly correlated, thereby implying that greater redundancy in $\mathcal{G}$. A smaller value of $RED(\mathcal{G})$ corresponds to the selection of a better feature subset.

The F**-measure** Fm is an external validation technique, using class labels as external information. It combines precision and recall [101], expressed as

$$Recall(i,j) = \frac{n_{ij}}{n_i}, \tag{1.8}$$

$$Precision(i,j) = \frac{n_{ij}}{n_j}, \tag{1.9}$$

where n_{ij} is the number of patterns belonging to class i that fall in cluster j, and n_i, n_j are the cardinalities of class i cluster j respectively. The $Fm(i,j)$ of cluster j and class i is computed as

$$Fm(i,j) = \frac{2 \times Recall(i,j) \times Precision(i,j)}{Recall(i,j) + Precision(i,j)}. \tag{1.10}$$

No one-to-one mapping exists between a class and a cluster. The $Fm(i)$ for a particular class i is given as

$$Fm(i) = \max_{0<j<k} Fm(i,j). \tag{1.11}$$

Finally, the F-measure is evaluated as

$$Fm = \sum_i \frac{n_i}{N} Fm(i), \tag{1.12}$$

with values lying in the range $[0,1]$, and a larger value of Fm indicating improved quality of clustering.

Next we describe a few external measures for comparing different sets of partitioning, over the same or different feature spaces. Let U be a set of clusters $U_1, U_2, \ldots, U_k$. **Jaccard**

Index (JI) [6], between two sets of clusters (partitioning) U and U', is defined as

$$JI(U, U') = \frac{n_{11}}{n_{11} + n_{10} + n_{01}}. \tag{1.13}$$

Here n_{11} is the number of pattern pairs lying in the same cluster under both sets of partitions U and U', n_{10} is the number of pattern pairs falling in the same cluster under U but not in U' and n_{01} is the number of pattern pairs that belong to the same cluster under U' but not in U. A value of JI nearer to 1 indicates a better match between the clusters from the two different partitioning spaces U and U'.

Rand Index (RI) [99] is used to compare the partitioning sets U and U' as

$$RI(U, U') = \frac{n_{11} + n_{00}}{N(N-1)/2}, \tag{1.14}$$

where n_{00} is the number of pattern pairs that belong to different clusters under partitioning U and U'. A value of RI nearer to 1 indicates a better matching between U and U'. We have $0 \leq JI, RI \leq 1$.

An information theoretic measure **Variation of Information** (VI) [81] is also used to compare the partitioning spaces U and U'. It is defined as

$$VI(U, U') = H(U) - H(U') - 2I(U, U'), \tag{1.15}$$

where

$$H(U) = -\sum_{j=1}^{k} P(j) log(P(j)) \tag{1.16}$$

is the entropy associated with clustering U and

$$I(U, U') = \sum_{j=1}^{k} \sum_{j'=1}^{k'} P(j, j') log \frac{P(j, j')}{P(j)P(j')} \tag{1.17}$$

is the mutual information between clustering U and U'. Here $P(j) = n_j/N$ is the probability of a pattern belonging to cluster U_j, where n_j is the number of patterns in the cluster

U_j, and $P(j, j') = \frac{|U_j \cap U'_{j'}|}{N}$ is the probability that a pattern belongs to the jth cluster in both U and U'. We have $0 \leq VI \leq logN$. A value of VI nearer to 0 implies better matching of the partitions in the spaces U and U'.

The **Jaccard Score (**JAC**)** evaluates the proficiency of a selected feature subset in preserving pairwise sample similarity, and is computed as [127]

$$JAC(M_{\mathcal{G}}, M, m) = \frac{1}{N} \sum_{i=1}^{N} \frac{NN(i, m, M_{\mathcal{G}}) \cap NN(i, m, M)}{NN(i, m, M_{\mathcal{G}}) \cup NN(i, m, M)}. \tag{1.18}$$

Here $M_{\mathcal{G}} = X_{\mathcal{G}} X_{\mathcal{G}}^T$ is a similarity matrix computed over the selected feature set $\mathcal{G}$ (using the inner product), $X_{\mathcal{G}}$ is the pattern set with these $\mathcal{G}$ features, and M is the similarity matrix computed in the original feature space; $NN(i, m, M)$ and $NN(i, m, M_{\mathcal{G}})$ denote the m-nearest neighbors of the ith sample according to M and $M_{\mathcal{G}}$ respectively. JAC measures the average overlapping of the neighborhoods specified by $M_{\mathcal{G}}$ and M, with a higher score indicating a better preservation of sample similarity.

1.4.3 Role of similarity

The concept of similarity is basic to human experience. In everyday life it implies some degree of closeness between two physical objects or ideas or patterns, with the metric being often used as a standard for measurement. Effective solutions for data indexing and data mining often require that appropriate measure of pattern-to-pattern similarity be provided. Let X be a set. A function $s : X \times X \rightarrow \mathcal{R}$ is called similarity or proximity in X if, for $\forall\ x, y \in X$, we have

1. $s(x, y) \geq 0$ (non-negativity);
2. $s(x, y) = s(y, x)$;
3. $s(x, y) < s(x, x)\ \forall x, y \in X : x \neq y$ and $s(x, y) = s(x, x)$ if and only if $x = y$.

If any distance satisfies the triangular property then it is called a metric. Such sets of distances or similarities are of importance in pattern recognition, as they help in projecting patterns or objects closely if they are in the same cluster or group, and far apart if they belong to different clusters or groups. Any use of such similarity measures involves implicit assumption that the data objects or patterns naturally form groups, which can be regarded as arising from different generation mechanisms while sharing common statistical characteristics [25, 38].

Unsupervised learning aims to group objects based on similarities, with the measure being highly dependent on the features representing the data. Many learning algorithms assume the domain expert to have determined the relevant features. Since all features are not equally important, there can exist redundancy, irrelevance or noise – which can again misguide learning. The challenge is to identify and eliminate unimportant features from the datasets, thereby increasing comprehensibility under the curse of dimensionality.

The concept of preservation of sample similarity has been used to identify irrelevant features [49, 123] as well as to remove redundant features [127]. Zhao and Liu [123] (SPEC framework) ranked each feature based on their alignment to the leading eigenvectors of the pairwise similarity matrix of samples, thereby preserving the geometric structure of data. This was employed in Sections 3.4.2 and 3.4.3. He *et al.* [49] evaluated features individually, depending on their capability for preserving the locality in terms the nearest neighbors of sample points. Another popular feature selection algorithm, based on nearest neighbor approach, is ReliefF [65, 102]. This supervised algorithm ranks each individual feature depending on how well it can distinguish all neighbouring same-class data points in the training set from those belonging to different classes. This has been used in Sections 2.3.4, 3.4.2 and 3.4.3.

Since these algorithms handle each feature individually, while neglecting possible correlation between different features in the set, therefore there exists a chance of redundant feature(s) being retained in the reduced subset; such that eventually the selected feature set may not be optimal [11]. Zhao *et al.* [127] overcame this limitation by collectively evaluating a set of features, and solved the combinatorial optimization formulation using sequential forward selection (SPFS-SFS) approach. This is employed in Sections 3.4.2 and 3.4.3.

Feature selection using feature similarity is not new in literature. Maximal information compression index (fsfs) has been used in Ref. [83], to measure the similarity between features based on their linear dependence. This feature selection method initially partitions the original feature set into distinct subsets or clusters, such that all features within a cluster are highly similar to each other and vice versa. A single representative feature from each such cluster is then selected, based on the nearest neighbors of the features, to constitute the resulting reduced subset. The maximal information compression index

$$\lambda_2 = 1/2[vr(\vec{f_i}) + vr(\vec{f_j}) - \sqrt{(vr(\vec{f_i}) + vr(\vec{f_j}))^2 - 4vr(\vec{f_i})vr(\vec{f_j})(1 - \rho(\vec{f_1}, \vec{f_2}))^2}] \tag{1.19}$$

is used for feature clustering, where $vr(\vec{f_i})$ is the variance of the feature vector $\vec{f_i}$, $\rho(\vec{f_i}, \vec{f_j}) = \frac{cov(\vec{f_i},\vec{f_j})}{\sqrt{vr(\vec{f_i})vr(\vec{f_j})}}$ is the Pearson correlation coefficient between $\vec{f_i}$ and $\vec{f_j}$, and $cov(\vec{f_i}, \vec{f_j})$ is the covariance between $\vec{f_i}$ and $\vec{f_j}$. Here λ_2 becomes zero when the features are linearly dependent, and it increases as the amount of dependency decreases. The computational complexity of this scheme is $O(D^2 * N)$ [83], with N being the cardinality of the samples or instances of a dataset. However any variation in the set of nearest neighbors can influence the cluster of features, and thereby affect the final feature set. This is used in Section 4.4.

The Hilbert-Schmidt independence criterion (HSIC) [41] has also been used to measure the similarity between features [18]. It maps the feature vectors into a Reproducing Kernel Hilbert Spaces (RKHS) to calculate the norm between them. The algorithm computes

$$HSIC(\vec{f_i}, \vec{f_j}) = \frac{1}{N(N-3)}[trace(\widetilde{K}\widetilde{L}) + \frac{\mathbf{1}^T\widetilde{K}\mathbf{1}\mathbf{1}^T\widetilde{L}\mathbf{1}}{(N-1)(N-2)} - \frac{2}{N-2}\mathbf{1}^T\widetilde{K}\widetilde{L}\mathbf{1}], \tag{1.20}$$

where $\widetilde{K}$ and $\widetilde{L}$ are kernel matrices with diagonal elements equal to 0, *i.e* $\widetilde{K} = K - diag(K)$ and $\widetilde{L} = L - diag(L)$. Here $K(p,q) = \phi_1(f_{ip}, f_{iq})$ and $L(p,q) = \phi_2(f_{jp}, f_{jq})$ are the kernel matrices with $p,q = 1,\dots N$. ϕ_1 and ϕ_2 are the Gaussian mapping functions [18, 41]. Note that the choice of the kernel affects the measurement of dependence or similarity between the features. The computational complexity of HSIC (computed between a pair of features) is $O(N^2)$ [109]. It is employed in Section 4.4.

Mutual information, in terms of minimal-redundancy-maximal-relevance (mRMR) criterion [26] has been used [95] to measure the maximal statistical dependency or similarity between features. The algorithm proceeds by selecting features incrementally, *i.e.*, it includes a feature into an already generated subset when the inclusion improves the overall mutual information of the subset. The supervised mRMR scheme [95] selects features incrementally by optimizing

$$\max_{\vec{f_j} \in \{\vec{\mathcal{G}}_D - \vec{\mathcal{G}}_{m-1}\}} [MI(\vec{f_j}, \vec{w}) - \frac{1}{m-1} \sum_{\vec{f_i} \in \{\vec{\mathcal{G}}_{m-1}\}} MI(\vec{f_j}, \vec{f_i})]. \tag{1.21}$$

Here $MI(\vec{f_i}, \vec{w})$ is the mutual information between feature vector $\vec{f_i}$ and target class vector $\vec{w}$, $MI(\vec{f_j}, \vec{f_i})$ is the mutual information between $\vec{f_i}$ and $\vec{f_j}$, $\{\vec{\mathcal{G}}_{m-1}\}$ is the subset of $(m-1)$ selected features, and $\{\vec{\mathcal{G}}_D\}$ is the original feature space. The feature $\vec{f_i}$ from the set $\{\vec{\mathcal{G}}_D - \vec{\mathcal{G}}_{m-1}\}$, which maximizes eqn. (1.21), is selected at the mth step to generate a feature subset $\{\vec{\mathcal{G}}_m\}$ of size m. The Parzen window method is used to approximate the mutual information. However, in the process, the algorithm may also happen to miss the best subset. This measure is used in Section 4.4.

Related literature on feature subset evaluation include Category Utility score [23], Fisher's feature dependency measure [27, 111], and entropy-based unsupervised feature ranking [20]. These proceed by selecting the subset(s) of features while trying to preserve the inherent characteristics of the data. Authors have used an unsupervised method [116] that assumes a linear model to choose a subset of features while approximating the original data. Zhao *et. al*. [126] developed an embedded model which evaluates a feature subset based on its capability of preserving sample similarity.

The role of soft computing in efficiently handling similarity, from the perspective of feature selection, is one of the major thrusts of this thesis. We use fuzzy proximity to quantify topological neighborhood information, followed by its incorporation in dimensionality reduction. A secondary distance measure is then introduced to preserve sample similarity, and is used to identify optimal feature set(s) from the data. We also employ a relatively new statistic called distance correlation to measure feature dependence, and propagate this information in a belief propagation network to perform clustering of a feature space for deriving meaningful feature subset(s). Evaluation indices demonstrate that our algorithms produce more informative, less redundant feature subset(s) over related methods existing in literature. The selected subsets also resulted in comparatively higher predictive accuracy.

1.5 Scope of the Thesis

The objective of this thesis is to present some investigations, both theoretical and experimental, addressing certain aspects of unsupervised feature selection using similarity, involving structural, neighborhood and affinity between pattern pairs, and passing messages between feature pairs. Quantitative evaluation of the selected reduced feature subset(s) is also performed, and these results are compared with other state-of-the-art feature selection techniques.

Some of the issues covered in this thesis include concepts from structural similarity, shared nearest neighbors, and distance correlation, towards improved feature selection. The effectiveness of the different algorithms is demonstrated on one synthetic, along with fifteen sets of publicly available real data viz. Iris[1], Spambase[1], Ionosphere[1], Multiple Features (MF)[1], Isolet[1], ORL[2], COIL20 [88], USPS[3], NSL-KDD[4], Colon[5], Leukemia[6], Prostate[6], DLBCL[6], MLL[6]. The outline of the investigations is summarized below, under different chapter headings.

1.5.1 Feature selection using structural similarity

A new method of feature selection is developed, based on structural similarity [72, 85]. The topological neighbourhood information about pairs of objects (or patterns), to partition(s), is taken into consideration while computing a measure of structural similarity. This is termed proximity, and is defined in terms of membership values. Multi-objective evolutionary optimization is employed to arrive at a consensus solution in terms of the contradictory criteria pair involving fuzzy proximity and feature set cardinality. Results on Iris, Ionosphere, Spambase, Isolet and Colon, and a synthetic dataset, show that the method led to a correct selection of the reduced feature subset from data having low, medium as well as high dimensionality. Comparative study is also provided, and quantified in terms of accuracy of classification and clustering validity indices.

1.5.2 Feature selection using SNN distance

In this chapter, we use the concept of Shared Nearest Neighbor (SNN) distance [53] to design a novel feature selection strategy. The algorithm strives to preserve the pairwise sample similarity in the selected feature subspace [71]. The similarity is measured in

[1]http://archive.ics.uci.edu/ml/datasets.html

[2]http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html

[3]http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multiclass.html#usps

[4]http://nsl.cs.unb.ca/NSL-KDD/

[5]http://microarray.princeton.edu/oncology

[6]http://www.biolab.si/supp/bi-cancer/

terms of the number of patterns common to the fixed size neighborhoods of a pair of sample points, as determined by primary distance measures like Euclidean, City block, Cosine. This is a filter model which collectively evaluates a set of features. A secondary similarity between pattern pairs is computed, based on a ranking of the nearest neighbors of each sample as induced by the primary distance (or similarity). Genetic algorithm (GA) is used to traverse the search space to find an optimal feature set.

This is then extended to improve the scalability to data with larger numbers of samples. In order to overcome the bottleneck of generating a large pairwise similarity matrix, we adopt a divide-and-conquer strategy. The data is randomly partitioned into nearly equal subsets, followed by a merger of the sample pairs having an SNN distance measure below some user-defined threshold within each such subset. Finally a feature subset is selected from this merged set of patterns, while preserving the pairwise sample similarity based on SNN distance. Results are provided on five publicly available datasets viz. MF, USPS, ORL, Spambase, and COIL20, along with comparative study involving related methods.

This work further extended in a multi-objective framework, which tries to preserve pairwise sample similarity while reducing the feature size [70]. A multi-objective framework is employed for the preservation of sample similarity, along with dimensionality reduction of the feature space. A reduced set of samples, chosen to preserve sample similarity, serves to reduce the effect of outliers on the feature selection procedure while also decreasing computational complexity. Experimental results on four sets of publicly available datasets viz. MF, USPS, ORL, and COIL20 demonstrate the effectiveness of this feature selection strategy. Comparative study with related methods is based on classification accuracy in the reduced space and evaluation indices.

1.5.3 Feature selection through message passing

A novel similarity-based feature selection algorithm is developed, using the concept of distance correlation. A feature subset is selected in terms of distance correlation between pairs of features, without assuming any underlying distribution of the data [69]. The pairwise similarity is then employed in a message passing framework, to select a set of exemplars features involving minimum redundancy and reduced parameter tuning. The algorithm does not need an exhaustive traversal of the search space.

The methodology is next extended to handle large data, using an inherent property of R. The effectiveness of the algorithm is demonstrated on nine sets of publicly-available data viz. Colon, Leukemia, DLBCL, Prostate, MLL, NSL KDD, Isolet, COIL20, and MF. The algorithm starts by simultaneously considering all the features as potential exemplars, and gradually updating messages on the basis of simple formulae that search for the minima of an appropriately chosen energy function. The magnitude of each message reflects the current affinity that one feature has for choosing another feature as its exemplar.

1.5.4 Conclusions and scope for further research

The concluding remarks with future scope of research are presented in Chapter 5.

Chapter 2

Feature Selection using Structural Similarity

2.1 Introduction

An interesting way of looking at feature selection is to aim at preserving the structural similarity of data clusters, while mapping a high-dimensional feature space to a lower-dimensional one. In other words, a pair of objects (or patterns) belonging to the same partition in the original high-dimensional space is expected to be retained in the same partition in the reduced domain as well. By considering such similarity or proximity between all object pairs as a guideline [94], one can hope to eliminate some of the less important features. The aim is to retain those features which allow the similarity between the partitioning, in the original and reduced spaces, to be high. This can also help in improving the computational efficiency in the lower dimensional space, given that the mapping is nearly lossless as measured in terms of the similarity measure used.

The chapter introduces a new method of feature selection, based on structural similarity.

The topological neighbourhood information about pairs of objects (or patterns), to partition(s), is taken into consideration while computing a measure of structural similarity. This is termed proximity, and is defined in terms of membership values of the corresponding patterns. For a dataset with N input patterns we can define an $N \times N$ symmetric matrix, referred as proximity matrix $\mathcal{P}$, whose (i, j)th entry represents the similarity (or dissimilarity) measure between the ith and jth patterns for $i, j = 1, \ldots, N$. Typically distance functions are used for the purpose. The proximity matrix is a pertinent construct that allows us to deal with structural information inherent in the data. In the fuzzy perspective the concept of similarity boils down to the membership value μ.

We focus on the use of proximity relationship, as a similarity measure, from the viewpoint of fuzzy sets. This is used as one of the objective functions, during multi-objective optimization, for evaluating the fitness of the feature subsets of varying cardinality. The use of fuzziness allows us to efficiently model uncertainties and ambiguities inherent in real life overlapping data. The proximity of a pair of patterns in the original feature space is compared with that in the reduced subspace of selected features. If they are similar, as measured in terms of their belonging to the same cluster (both before and after feature selection), then this implies that the eliminated feature(s) are not so relevant to the decision making process.

The second criterion is the cardinality of the selected feature subset. This is sought to be minimized, and serves as a penalty to the objective function. A close observation reveals that these two criteria are of a conflicting nature. A smaller subset of features is likely to result in a reduced proximity, and hence reduced classification accuracy (as compared to the original feature space).

Multi-objective optimization is employed to arrive at a consensus solution in terms of this contradictory criteria pair, involving fuzzy proximity and feature set cardinality. Here MOGA is used as a tool for the multi-objective optimization, and any other technique could also have sufficed [17]. The user does not need to specify the desired number of features, as it is embedded in the optimization process. The algorithm terminates when an optimal subset of features is obtained, according to the fitness criteria of the multi-objective genetic optimization. Experimental results indicate correct selection of the reduced feature subset. Validation of the selected set of features is reported in terms of classification accuracy using WEKA [46] implementation of several well-known classifiers, as well as internal and external clustering validity indices.

The rest of the chapter is organized as follows. In Section 2.2 we present the proximity-based methodology for feature selection and outline the background on multi-objective optimization. The experimental results and comparative study are described in Section 2.3, on Iris, Ionosphere, Spambase, Isolet and Colon, and a synthetic dataset. Finally, Section 2.4 concludes the chapter.

2.2 Proximity-based Feature Selection

Let us consider Fig. 2.1 to explain the concept of structural similarity between clusters in the context of feature selection. Using this crude example, we describe how that the idea of preserving cluster structure of original feature space in a feature subset actually leads to feature selection. Removing irrelevant feature(s) does not significantly affect the internal characteristics of the data. Three patterns $X1$, $X2$ and $X3$ are seen to be partitioned into the same cluster in the three-dimensional feature space of part (a). The three features are aligned with three reference axes *i.e.* x-axis, y-axis and z-axis of this dataset.

If the least important feature *i.e.* the feature aligned with y-axis is eliminated, the cluster structure is expected to remain unaltered; implying that the single cluster would still contain the same distribution of pattern points as depicted in part (b) of the figure. Here the three- to two-dimensional mapping is said to be almost lossless, such that the clustering structures in the two subspaces remain very similar. The clustering structure is thus said to be preserved in the transformation between these two subspaces.

On the other hand, if an important feature *e.g.* the feature aligned with z-axis is eliminated then the mapping is bound to disrupt the cluster structure, since important information gets lost in the process. From part (c) of the figure we observe that the similarity between the partitioning, in the two subspaces, is now no longer high. In other words, the distance between the partitioning is higher; with the pattern points getting redistributed into two different clusters *i.e.* cluster structure of original space is not preserved here..

2.2.1 Concept of proximity

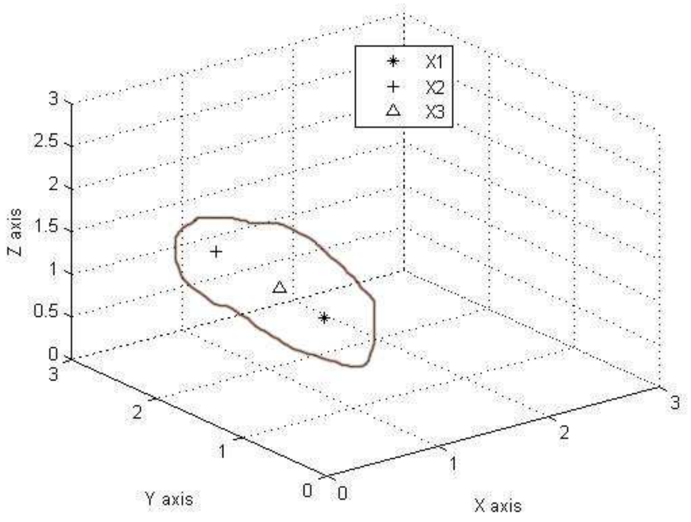
Proximity is used as a way of determining the similarity between clustering structures, while mapping from a high- to a low-dimensional feature space. In the process, we aim to retain the important features. Such preservation of structural similarity between clusters is expected to lead to the selection of important features. Let there be k subsets of data located in different feature subspaces, with the number of patterns in each subspace being equal to N.

Fuzzy c-Means (FCM)

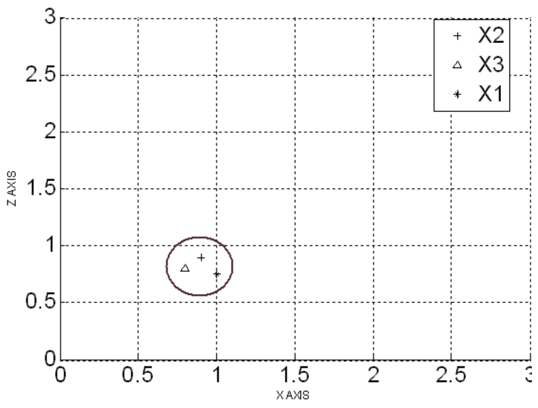
[8] The FCM is outlined here for our reference.

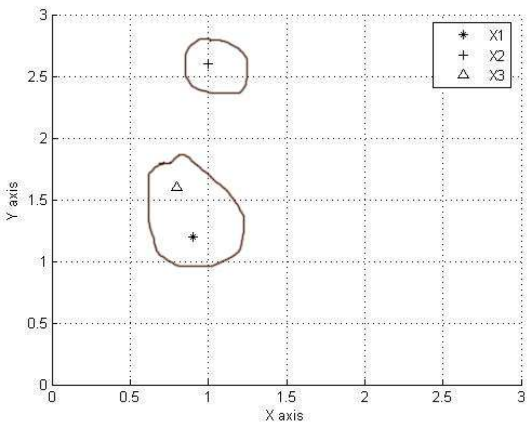
1. Assign c initial means $\vec{m}_j s$. Choose the value of fuzzifier ff and the number of iterations $iter$.

2. Repeat Steps 3 to 4 until there is no significant change in partitions, or upto a specific number of iterations $iter$.



(a)





(b) (c)

Figure 2.1: Mapping of patterns from (a) three-dimensional space, to a pair of two-dimensional spaces having cluster structure (b) preserved, and (c) not preserved

3. For $i = 1, 2, \ldots N$ and $j = 1, 2, \ldots, c$, compute the membership μ_{ij} of the ith pattern $\vec{x}_i$ to the jth cluster U_j by

$$\mu_{ij} = \frac{1}{\sum_{k=1}^{c} \left[\frac{dis(\vec{x}_i, \vec{m}_j)}{dis(\vec{x}_i, \vec{m}_k)}\right]^{\frac{2}{ff-1}}} \; : \; \sum_{j=1}^{c} \mu_{ij} = 1 \, \forall i \; and \; 0 < \sum_{i=1}^{N} \mu_{ij} < N \, \forall j. \tag{2.1}$$

   Put $\vec{x}_i$ in U_j if $\mu_{ij} > \mu_{ij_1}, \forall j_1 \neq j$.

   Resolve ties arbitrarily.

4. Update mean by

$$\vec{m}_j = \frac{1}{\sum_{i=1}^{N} \mu_{ij}^{ff}} \sum_{i=1}^{N} \mu_{ij}^{ff} * \vec{x}_i, \forall j. \tag{2.2}$$

Typically, $ff > 1$.

We form a $k \times N$ partition matrix $\mathcal{PR}$ consisting of membership values μ_{ij}. This membership value is updated by minimizing the objective function $\mathcal{J}_f$ of eqn. (2.3).

$$\mathcal{J}_f = min \sum_{j=1}^{c} \sum_{i=1}^{N} \mu_{ij}^{ff} * (\vec{x}_i - \vec{m}_j)^2. \tag{2.3}$$

We compute $\mu_{ij} \in [0, 1]$ as the membership of the jth pattern to the ith mean m_i, where $||.||$ is the distance norm and $1 \leq ff < \infty$ is the fuzzifier [8]. Note that the dimensionality d of the patterns in each subset could be different. However, in each subset, the distance of a pattern is computed from the fuzzy cluster prototypes [of eqn. (2.2)] over the same set of features.

The partition matrix is used to evaluate proximity px, which measures the extent to which a pair of patterns are regarded as similar or dissimilar in different subspaces [94]. This incorporates a mechanism of partial supervision in the process of navigating a structure in the data. The proximity matrix $\mathcal{P}$ contains the proximity values for all possible pairs of patterns. The fuzzy partitions generated by FCM, using eqn. (2.1), are directly related to the proximity relation. The proximity between pattern pair k_1 and k_2 is computed as

$$px(k_1, k_2) = \sum_{i=1}^{k} (\mu_{ik_1} \wedge \mu_{ik_2}), \tag{2.4}$$

where $\wedge$ denotes the minimum operation, $px(k_1, k_2) \in [0, 1]$, and $k_1, k_2 = 1, \ldots, N$. Evidently $px(k_1, k_2) = 1$ for $k_1 = k_2$, such that membership is evaluated with respect to FCM, and $px(k_1, k_2) = px(k_2, k_1)$.

The aim is to reduce the number of features, subject to maintaining the structural similarity between patterns. For this purpose multi-objective optimization is employed to handle the conflicting requirements of dimensionality reduction along with proximity preservation. We use MOGA [here NSGA-II] [22], as a tool to efficiently traverse the feature subspaces, subject to fulfilling the above objectives.

2.2.2 Proximity between feature subspaces

Let the cardinality of the original and reduced feature spaces be D and d, respectively. Let the proximity matrices in these two spaces be denoted by $\mathcal{P}$ and $\mathcal{P}'$. The similarity between the two matrices is represented by a scalar value

$$\mathcal{P}_s = \sum_{k_1=1, k_2>k_1}^{N} [px(k_1, k_2) \wedge px'(k_1, k_2)], \tag{2.5}$$

where $px'(k_1, k_2)$ is computed by eqn. (2.4) in the reduced feature space and $\wedge$ denotes the minimum operation.

Note that the membership value μ_{ik} at each stage is computed based on the FCM objective function, using eqns. (2.1)-(2.3). This becomes inherent in the proximity matrix in eqn. (2.4). Moreover, as the MOGA updates the encoded cluster means over the generations it has to continuously refer to the FCM based membership computations.

We retain only those pattern pairs which belong to the same cluster in both the original and the reduced feature space, in an attempt to reduce the ambiguity of the resultant clustering.

For such cases we use

$$\mathcal{P}_{s_0} = \sum_{k_1=1,k_2=k_1+1}^{N} [(px(k_1,k_2) \geq \theta) \wedge (px'(k_1,k_2) \geq \theta)], \tag{2.6}$$

such that $\mathcal{P}_{s_0}$ takes the minimum of the values of $px(k_1,k_2)$ and $px'(k_1,k_2)$ only when both $px(k_1,k_2)$ and $px'(k_1,k_2)$ are greater than a threshold θ. This implies that both μ_{ik_1} and μ_{ik_2} are greater than or equal to θ in the original and reduced feature spaces by eqn. (2.4).

2.2.3 Optimization tool

The multi-objective optimization is implemented using NSGA-II. We encode the problem as a real string of length L, with the first d bits corresponding to the d features in the original space. Here, in the bit representation, a "1" implies that the corresponding attribute is present while "0" indicates that it is not. The desired number of features need not be pre-specified, since it is automatically determined during the optimization. Let the size of a chromosome be

$$L = d + k \times d = d \times (k+1). \tag{2.7}$$

The k cluster centers (or prototypes) are encoded in real form in the subsequent $k \times d$ bits. Only those features of the centers in the second part of the string, corresponding to a "1" in the first part, are considered during clustering. Fig. 2.2 depicts such an encoding in a chromosome, representing a sample set of cluster prototypes in a feature subspace. Initially all the bits are set randomly.

The objective is to optimize a conflicting set of requirements; *i.e.*, select a minimal number of features that enable us to arrive at an acceptable structure-preserving mapping. We employ MOGA with $\mathcal{P}_{s_0}$ of eqn. (2.6) as the fitness function

$$F_1 = \mathcal{P}_{s_0}. \tag{2.8}$$

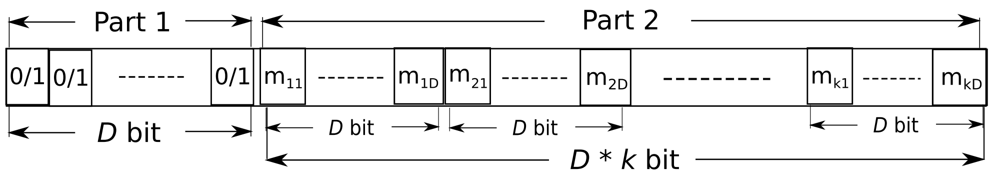


Figure 2.2: An encoded chromosome representing a feature subspace with the cluster prototypes

The second fitness function corresponds to the cardinality of the feature set under consideration, and is defined as

$$F_2 = d. \tag{2.9}$$

While F_2 is minimized to give credit to a candidate string containing less attributes, the function F_1 maximizes the extent to which all pairs of patterns belong to the same cluster in the two feature spaces, *viz.*, original and reduced subspace. These two fitness functions are optimized in the framework of MOGA. Clustering is done by FCM to update the prototypes $\vec{m}_i$, in the different subspaces.

2.2.4 The algorithm

The objective is to preserve the proximity relationship between pattern pairs, which is a measure of their structural similarity, while reducing the number of features. The main steps of the algorithm PR, outlined below, are repeated over several generations.

1. Initialize the population randomly, with real numbers.
2. Select a pair of chromosomes randomly for single-point crossover.
3. Perform two-point mutation simultaneously on the two parts of the string. In the first part, the value of the randomly chosen bit (signifying presence or absence of the corresponding attribute) is flipped. In case of the second part, the value $m_{ij_{old}}$

corresponds to the randomly chosen attribute j of the ith cluster center; this is mutated as

$$m_{ij_{new}} = \sigma \times x + m_{ij_{old}}, \tag{2.10}$$

where the perturbation $x(\sim Norm(0,1)$ is drawn from a Gaussian distribution, the variance σ^2 determines the magnitude of this perturbation at position $m_{ij_{old}}$, and $m_{ij_{new}}$ is its new value (at the corresponding attribute j of the ith cluster center) after mutation.

4. Compute the fitness values of different feature sets based on their proximity and cardinality, using eqns. (2.8)-(2.9).

5. Rank the population using dominance criteria. Compute the crowding distance of the chromosome, to maintain diversity in the population [22].

6. Combine parent and offspring population. Replace the parent population by the best members of the combined population.

Note that the cluster centers are initially set randomly. During crossover and mutation the centers get modified. Their effect is reflected through the proximity function [eqn. (2.8)] into the fitness evaluation. The features present in a chromosome, as indicated by the "1"s in the first part, determine the reduced feature subspace. They affect the computation of proximity in terms of cluster prototypes, using eqns. (2.1)-(2.3) and (2.4). Finally the selected feature sets are validated in terms of cluster validity indices [eqns. (1.6) and (1.12)], and the classification accuracy.

2.3 Experimental Results

The performance of the algorithm was tested on various synthetic and real datasets. These include (i) a synthetic dataset and the benchmark *Iris* flower (low-dimensional), (ii) *Iono-*

sphere and *Spambase* (medium-dimensional), and (iii) *Isolet* and *Colon* cancer microarray gene expression data (high-dimensional). All results were averaged over several (3-5) runs involving different random seeds. No significant change was observed in the performance, using different seeds. The choice of θ in eqn. (2.6) was taken to be 0.5, so that the membership of pattern pair k_1, k_2 became simultaneously high in the same cluster. The crossover and mutation probabilities, in the MOGA, were selected as 0.85 and 0.05 respectively after several experiments. The clustering was evaluated in terms of clustering validity indices S and Fm [of eqns. (1.6) and (1.12)]. The selected feature subsets were externally validated on their predictive accuracy, using the publicly available WEKA implementation [46] of different classifiers like k-nearest neighbors (k-NN), Naive Bayes' (NB) and support vector machine (SVM) [described in Section 1.2.1], involving ten-fold cross validation. The clustering structures of reduced and original feature spaces are compared using JI, RI and VI of eqns. (1.13)-(1.15).

2.3.1 Data description

The synthetic data contains three clusters, each with 100 randomly generated patterns. The two-dimensional scatter plot of Fig. 2.3 depicts the patterns lying within circles of unit radii, each having different centers. A lot of overlapping is artificially introduced. We included a third attribute having completely random values, to evaluate the effectiveness of the algorithm in identifying the significance of the first two features. The *Iris* data consists of 150 pattern points with four input features corresponding to measurements of *sepal length, sepal width, petal length, petal width* on fifty flowers from each of the three species *setosa, versicolor, virginica* (representing the three output classes).

The *Ionosphere* data represents autocorrelation functions of radar measurements. There are 351 instances, each having 34 (continuous) features and belonging to two classes, *viz.*

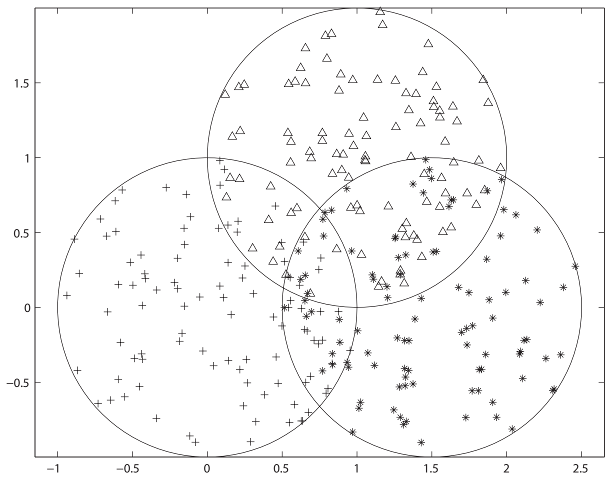


Figure 2.3: Synthetic data

“good” or “bad” – indicating the passage or obstruction of free electrons in the ionosphere. We considered a total of 32 features (attributes 3 to 34) as input to the algorithm. The *Spambase* data consists of 4601 instances of emails, to be classified into spam or nonspam categories. There are 57 continuous attributes denoting word frequencies.

The *Isolet* data consists of several spectral coefficients from the utterance of English alphabets by 150 subjects. There are 617 real features (having values in the range [0,1]) with 7797 instances and 26 classes. The above-mentioned three datasets were taken from the UCI Machine Learning Repository, as indicated in Section 1.5.

The *Colon Cancer* data is a collection of 62 gene expression measurements from colon biopsy samples. There are 22 normal and 40 colon cancer samples, having 2000 genes (features). Typically, microarray gene expression data involves a larger number of features (genes) as compared to the samples (time points). In other words, the features correspond to gene expression values that indicate the abundance of mRNA in a sample (or tissue) for a number of patients; with the objective being to separate cancer patients from healthy ones based on their gene expression profiles. Many of these features are redundant and adversely affect the output decision. Hence preprocessing is often needed [44] to initially eliminate some of the irrelevant features. Some initial preprocessing [5] was done, to reduce the large number of redundant genes to 943, before applying our algorithm.

2.3.2 Low- and medium-dimensional data

The performance of the algorithm for strings generated in the non- dominated Pareto front, for the four datasets (having low and medium number of features), are presented in Tables 2.1-2.2. The second column (in both tables) indicates the selected attributes, marked by a "1" in the first part of the chromosome, with the string corresponding to feature positions $1, 2, \ldots, D$. The two fitness functions are evaluated by eqns. (2.8)-(2.9). However for the cases where the original feature space did not figure in the Pareto optimal front, this is still included as the last row for each dataset in the table for comparison (without any F_1). The external validation performance of the selected feature subsets is provided, along with that of the original set, in terms of classification accuracy involving ten-fold cross-validation using different classifiers. The algorithm was run for 100 generations with a population size of 50 chromosomes. The last two columns indicate the Silhouette index (S) [eqn. (1.6)] and F-measure (Fm) [eqn. (1.12)] values.

We know that the synthetic data is represented with the first two attributes, and the third feature was inserted randomly. As evident from the results, the selection of the first two features (only) generally results in the best overall accuracy, as well as S and Fm, due to the elimination of this unimportant third feature. The feature set {1,2} also produces better clustering in reduced space according to JI [eqn. (1.13)], RI [eqn. (1.14)] and VI [eqn. (1.15)].

In case of the *Iris* data, it is observed that the choice of feature 3 occurs in all the three cases, with feature 4 being selected the second-most frequently. Together they result in the

Table 2.1: Performance of selected feature subsets, of low cardinality, from Pareto-optimal front

Dataset	Feature subspace	F_1 prox. $(\times 10^4)$	F_2 card.	Validation accuracy (%) by				Silh. stat. S	Fm-meas.	JI	RI	VI
				k-NN		NB	SVM					
				1	77.4							
	$\{1, 2, 3\}$	2.10	3	3	73.9							
	(Original)			5	76.0	78.0	78.7	0.055	0.395	–	–	–
				7	77.6							
Synthetic				1	60.3							
	$\{1\}$	0.74	1	3	59.5							
$N = 300$				5	60.2	62	61.3	0.044	0.622	0.21	0.55	2.13
$D = 3$				7	60.3							
$k = 3$				1	**79.2**							
	$\{1, 2\}$	1.53	2	3	**79.1**							
				5	**80.4**	**80**	**80.3**	**0.088**	**0.801**	0.20	0.56	2.17
				7	**80.8**							
				1	94.7							
	$\{2, 3, 4\}$	0.58	3	3	94.2							
				5	93.8	96.0	**97.3**	0.176	0.940	0.80	0.93	0.38
				7	93.0							
Iris				1	93.0							
	$\{3\}$	0.34	1	3	92.9							
$N = 150$				5	92.1	**96.7**	95.3	0.214	0.933	**0.82**	**0.93**	**0.37**
$D = 4$				7	92.4							
$k = 3$				1	**94.8**							
	$\{3, 4\}$	0.36	2	3	**94.6**							
				5	**94.2**	96.0	96.7	**0.219**	**0.950**	0.79	0.92	0.42
				7	**94.3**							
				1	93.3							
	(Original)	–	4	3	92.6							
				5	91.4	96.0	96.7	0.156	0.677	–	–	–
				7	89.7							

second highest proximity and second lowest cardinality. It is well-known that these are the two features most important for discriminating between the classes in this benchmark data. Interestingly, the performance of the k-NN in the reduced space (involving attributes 3 and 4) is found to be the overall best – inspite of the elimination of two features. The same holds for the validity indices S and Fm. The SVM provides best accuracy with three features while NB performs best with only feature 3. The JI, RI and VI also demonstrate that the cluster structure is best preserved along feature 3.

The results from Table 2.2 exhibit better average classification performance by k-NN and SVM, for *Spambase*, with a smaller set of features *viz.* 13 and 15. The values of both S and Fm are also the best with 15 features. Although NB provides a better score of 79.3% in the original space, yet its performance with 15 features is comparable at 79.0%. The JI, RI and VI provide best result with 11 features *i.e* the original cluster structure is preserved in this feature space.

Results for the *Ionosphere* data demonstrate that out of the 32 initial attributes our algorithm selected a cardinality of 5 and 7 for the best performance in terms of mean recognition accuracy (%) by k-NN. In Fig. 2.4 we depict a visually understandable, three-dimensional projection, in terms of attributes 4, 5, 6 of the 32-dimensional data. Incidentally, this corresponds to the best performance by classifier NB. It is observed here that our algorithm selected a reasonably good set of features, which captured the structural similarity between the two classes in the original feature space (at the best values of S and Fm). The best feature subset in terms of structure preservation is a set of 16 features according to the JI, RI and VI.

Next the scope of the algorithm was extended to incorporate a variation in the number of clusters. We determined the optimum number of clusters k_o (varying k from 2 to 12), in both the original and reduced feature spaces, by maximizing the Silhouette index of eqn. (1.6). FCM is used to determine the fuzzy partitioning corresponding to k_o clusters, for

Table 2.2: Performance of selected feature subsets, of medium cardinality, from Pareto-optimal front

Dataset	Feature subspace	F_2 card.	Validation accuracy (%) by				Silh. stat. S	Fm-meas.	JI	RI	VI
			k-NN		NB	SVM					
	{27, 28, 29, 47,		1	71.0							
	48, 49, 53, 54,	11	3	69.9	61.2	77.4	0.099	0.664	**1.00**	**1.00**	**0.00**
Spambase	55, 56, 57}		5	70.0							
			7	69.4							
$N = 4601$	{3, 4, 9, 10,		1	77.9							
$D = 57$	11, 22, 23, 24,	15	3	78.1	79.0	**85.5**	**0.131**	**0.742**	0.55	0.58	0.85
$k = 2$	35, 36, 37, 38,		5	78.2							
	52, 53, 54}		7	77.9							
$F_1 =$	{6, 7, 8, 12,		1	**80.4**							
105.8×10^5	13, 14, 15, 26,	13	3	**80.3**	65.5	81.2	0.039	0.672	0.82	0.82	0.38
	27, 28, 36, 44,		5	**80.3**							
	45}		7	**79.7**							
			1	72.5							
	(Original)	57	3	72.0	**79.3**	83.7	0.098	0.664	–	–	–
			5	71.6							
			7	71.1							
	{6, 7, 8, 9,		1	90.8							
	10, 11, 16, 17,	16	3	90.6							
	18, 22, 23, 24,		5	90.2	74.9	90.3	0.077	0.724	**0.94**	**0.97**	**0.16**
	25, 29, 30, 31}		7	90.4							
			1	90.9							
	{4, 5, 6, 33, 34}	5	3	**92.9**							
Ionosphere			5	**92.6**	88.6	90.6	0.108	0.731	0.54	0.69	0.96
			7	**92.3**							
$N = 351$			1	86.2							
$D = 32$ (2-34)	{4, 5, 6}	3	3	90.1							
$k = 2$			5	91.9	**89.7**	90.3	**0.156**	**0.836**	0.45	0.57	1.08
			7	92.3							
$F_1 =$			5	**91.7**							
0.61×10^5	{14, 21, 22, 23,	7	3	92.0							
	24, 25, 26}		5	92.2	70.4	85.8	0.070	0.733	0.75	0.85	0.51
			7	92.0							
			1	91.1							
	(Original)	32	3	91.3	81.8	**94.0**	0.078	0.700	–	–	–
			5	91.7							
			7	92.0							

Table 2.3: Performance of selected feature subsets, from Pareto-optimal front, allowing variation in number of clusters

Dataset	Feature subspace	F_1 prox. ($\times 10^4$)	F_2 card.	Validation accuracy (%) by		Silh. stat. S	Fm-meas.	JI	RI	VI
				NB	SVM					
Iris $N = 150$	$\{3\}$	0.549	1	96.7	95.3	0.311	0.933	0.82	0.93	0.37
$D = 4$	$\{3, 4\}$	0.556	2	96.0	96.7	0.311	0.950	0.79	0.92	0.42
$k = 3$	$\{1,2,3,4\}$ (Original)	0.564	4	96.0	96.7	0.311	0.677	–	–	–
Ionosphere	$\{3,5,8,13,15,17,19,21,31\}$	2.739	9	82.6	89.5	0.150	0.740	0.85	0.92	0.29
$N = 351$ $D = 32$	$\{3,5,8,15,17,21,31\}$	2.733	7	86.9	89.7	0.150	0.728	0.85	0.92	0.29
$k = 2$	$\{3,5,8,13,15,17,21,31\}$	2.737	8	85.7	89.2	0.150	0.746	0.85	0.91	0.29

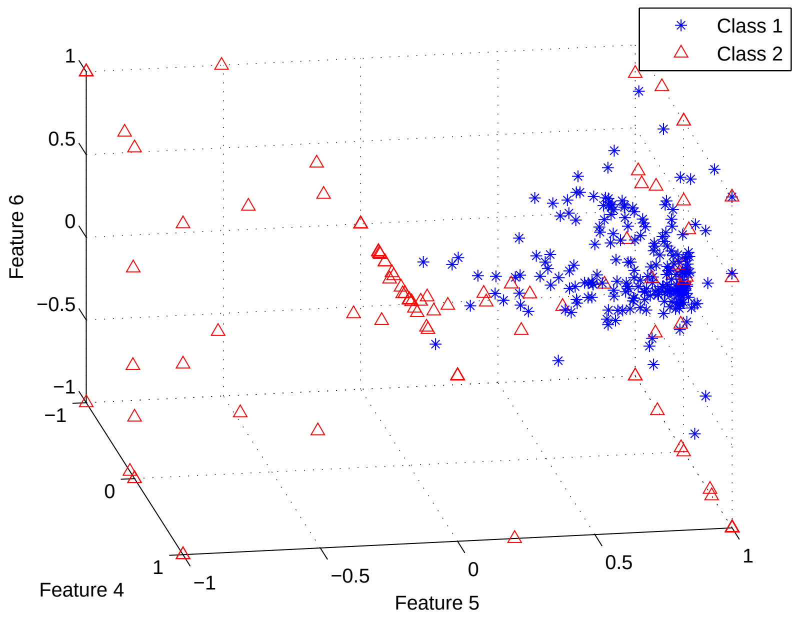


Figure 2.4: Projection of *Ionosphere* data in three-dimensional space

each generated feature subspace. Multiobjective optimization in terms of maximization of proximity [eqn. (2.8)] and minimization of cardinality of the feature space [eqn. (2.9)] ensures the selection of those feature subsets that retain structural similarity among the clusters. The encoded chromosome of Fig. 2.2 now involves only the first D bits. However, the computational complexity gets enhanced and adversely affects the processing of large data.

Table 2.3 depicts the results for the *Iris* and *Ionosphere* data. In all the cases the optimum number of clusters converged to $k_o = 2$. Incidentally the corresponding value of S was found to be better here, as compared to Tables 2.1-2.2. The algorithm, in this modified framework, generated the same subsets of reduced features in Table 2.3 as in Table 2.1. In case of *Synthetic* data the algorithm failed to eliminate the random third feature. The *Spambase* data was found to be too large to be processed, upon varying the number of clusters. With the *Ionosphere* data we obtained a different set of reduced feature subsets, that were generally comparable in terms of predictive accuracy and F-measure. The result of the JI, RI and VI shown that cluster structure is also preserved when we allow variation in the number of clusters.

2.3.3 High-dimensional data

Table 2.4 presents the average performance of the algorithm (over ten runs), corresponding to strings generated in the non-dominated Pareto front, for the high-dimensional *Isolet* and the microarray *Colon* cancer data. The algorithm was run for 100 generations with a population size of 40 chromosomes. There were 15,000 generations, with a population size of 200. The 10-fold cross validation was used to compute the classification accuracy in both the cases.

With the *Isolet* data we observe that the performance of the classifiers is, in general, better in the original feature space. However, both NB and SVM provide comparable classification accuracy with less than half the number of features. The value of S is found to be better in the reduced space. The values of JI, RI and VI indicate that the clustering obtained in the reduced space preserves the structure present in the original space.

In case of the *Colon* microarray data we observe that the performance of NB and k-NN (for k = 1, 3) is better with reduced features. The same is true for Fm. Keeping in mind that the reduction in feature set cardinality is almost ten times, as compared to the original set of 2000 features, the overall performance can be said to be reasonably good in the reduced space. The JI, RI and VI values show that our algorithm succeeded in preserving the cluster structure over the reduced subset, when compared to the original as well as preprocessed feature spaces.

Table 2.4: Performance of some selected feature subsets, of large cardinality, from Pareto-optimal front

Dataset	F_2 cardinality	Validation accuracy (%) by				Silhouette statistic S	Fm-measure	JI	RI	VI
		k-NN		NB	SVM					
	275	1	77.6	84.9	94.9	2.4 $\times 10^{-3}$	0.344	0.88	0.94	0.28
Isolet		3	79.1							
		5	80.2							
$N = 7797$		7	80.7							
$D = 617$	274	1	77.5	84.8	94.8	2.4 $\times 10^{-3}$	0.336	**0.89**	**0.94**	**0.26**
$k = 26$		3	79.1							
		5	80.3							
$F_1 = 2.91 \times 10^7$		7	80.7							
	617 (Original)	1	**92.7**	**85.1**	**95.5**	1.5 $\times 10^{-3}$	**0.365**	–	–	–
		3	**93.7**							
		5	**94.1**							
		7	**94.1**							
	261	1	**83.9**	54.8	64.5	1.5 $\times 10^{-2}$	0.704	1.0	1.0	0.0
Colon		3	**80.7**							
$N = 62$		5	71.0							
$D = 2000$		7	71.0							
$D_{preproc} = 943$	264	1	83.9	54.8	64.5	1.5 $\times 10^{-2}$	0.704	1.0	1.0	0.0
$k = 2$		3	80.6							
		5	71.0							
$F_1 = 1.25 \times 10^3$		7	71.0							
	943 (Preproc.) [5]	1	77.7	53.2	64.5	1.2 $\times 10^{-2}$	0.704	–	–	–
		3	79.7							
		5	**75.8**							
		7	**74.8**							
	2000 (Original)	1	77.1	53.2	**82.3**	2.4 $\times 10^{-2}$	0.687	–	–	–
		3	77.7							
		5	75.2							
		7	73.9							

2.3.4 Comparative study

The performance of our algorithm (model PR) for *Iris* data was compared with that of some of the existing techniques, considered as benchmark in this study. These are

1. the statistical method of Devijver and Kittler [24] (model DK), which uses probabilistic distance measure to assess discriminatory information conveyed by a set of features,
2. the fuzzy entropy based method of Pal and Chakraborty [92] (model PC), which is defined in terms of interclass and intraclass distances of patterns,
3. the neural network based method of Ruck and Rogers [105] (model R*), which used MLP and saliency measure for feature selection, and
4. the model of Ishibuchi [55] (model IM), which used MLP and a variant of class separability.

Table 2.5 demonstrates a comparative study of the feature subsets selected by different algorithms for the *Iris* data. As *Iris* data is typically studied by researchers (in the pattern recognition field), an extensive comparison has been provided for this data. The overall study shows that the results tally with each other. The features 3 and 4 were always found to be more important than the features 1 and 2 for classifying *Iris* data.

Next the average classification performance of the feature set selected by algorithm PR was compared (on some of the datasets) over a test set (90% of the data) with the performance of those selected by certain existing unsupervised techniques, averaged on 10 runs, using a training set size of 10%. The well-known feature selection algorithms considered were SFS [24] (described in Section 1.2.3), SFFS [96], SWC [64] and BB [87]. We also compared the performance of the supervised Relief-F [65] in a similar manner.

Table 2.5: Comparative study on *Iris* data

Algorithm	Features providing best performance
PR	{3, 4}
DK	{3, 4}
PC	{4, 3}
IM	{3, 4}
$R*$	{3, 4}

Table 2.6 presents a comparison of the average classification performance, by the k-NN over $k = 1, 3, 5, 7$, for sample feature subsets selected by all these algorithms for datasets *Iris, Spambase, Ionosphere* and *Isolet*. In each case the initial D features were reduced to d (for uniformity of comparison with PR). In general, our algorithm PR was better than the supervised *Relief-F* for data *Isolet* and comparable for data *Iris*. As compared to the other algorithms, PR was always found to be better.

For *Iris* $d = 2$ corresponds to the minimal subset selected by our algorithm in Table 2.1. However with *Spambase* and *Ionosphere* we observed that a lower cardinality of 13 (row 4) and 5 (row 6) by PR in Table 2.2, respectively, provided a higher classification accuracy as compared to that generated by the larger subsets, $d = 27$ and 16 respectively (d as reported in [86]) in Table 2.6.

Since BB and $SFFS$ algorithms required infeasibly high computation time for high-dimensional data, we did not include them for the comparison involving *Isolet*. The performance was best with PR for $d = 309$ (as reported in [86]). On the other hand, Table 2.4 indicates the lowest cardinality of 274 with a poorer average classification accuracy (as compared to that using $d = 309$). The computational complexity of the algorithm PR is

Table 2.6: Comparative study with k-NN classifier on some data

Dataset	Algorithm	Accuracy (%)	
		Mean	SD
Iris $D = 4$ $d = 2$, $k = 3$	PR	94.49	1.34
	BB	92.29	2.57
	SFS	92.29	2.57
	$SFFS$	92.29	2.57
	SWC	93.48	2.03
	$Relief-F$	95.68	0.65
Spambase $D = 57$ $d = 27$, $k = 2$	PR	79.75	0.99
	BB	70.93	0.70
	SFS	70.73	0.77
	$SFFS$	70.73	0.77
	SWC	76.40	1.05
	$Relief-F$	89.00	0.28
Ionosphere $D = 32$ $d = 16$, $k = 2$	PR	78.67	1.81
	BB	75.96	0.35
	SFS	69.94	0.32
	$SFFS$	74.73	0.37
	SWC	62.03	0.32
	$Relief-F$	89.90	1.30
Isolet $D = 617$ $d = 309$, $k = 26$	PR	94.60	0.38
	SFS	74.45	1.20
	SWC	78.25	1.22
	$Relief-F$	90.40	0.30

Table 2.7: Execution time of Algorithm PR on different datasets

Dataset	Execution time (Second)
Iris	10
Synthetic	55
Spambase	6300
Ionosphere	39
Colon	18000
Isolet	86052

$O(gS_tDN^2)$, where S_t is population size, N corresponds to the number of pattern points, and g is the number of the generations. Now complexity of ReliefF algorithm is $O(t_nND)$, where t_n is the number of training samples used for finding nearest neighbour [102]. It has higher time requirements for datasets containing large number of samples. The computational complexity of BB, SFS, and SFFS algorithms are infeasibly high for large data sets [86].

Incidentally, we also explored the use of k-means [24] clustering during proximity computation. This resulted in the generation of binary values in the proximity matrix, instead of values lying in the range [0,1]. The presence of a number of "ones" in the matrix perhaps lead to a greater homogeneity between the chromosomes of the population, as evaluated by the first objective function of eqn. (2.8). Thereby, during multi-objective optimization this objective function plays a less significant role as compared to the cardinality of the feature space [eqn. (2.9)]. Hence k-means almost always resulted in a minimum cardinality of feature space, typically one, with no emphasis on the cluster structure. This highlights the utility of fuzzy clustering in our algorithm PR. The execution time of PR over different set was reported on Table 2.7. The execution time was computed on a HP Z800 workstation with Xeon(R) 2.67 GHz CPU and 16 GB RAM.

2.4 Conclusion

A new feature selection algorithm, based on structural similarity, has been described. Fuzzy proximity was used to evaluate the similarity between the original and reduced feature subspaces. The cardinality of the feature subset was simultaneously minimized. The optimal number of features was automatically determined during the multi-objective optimization. This algorithm preserves the performance of the benchmark classifiers as well as cluster structure in the reduced space. Comparative study demonstrated the effectiveness of the developed method.

The topological neighbourhood information, pertaining to the inherent cluster structure in the data, was utilized while achieving reduction in feature subspace cardinality. This is expected to have wide ramifications in data mining, data analysis and retrieval, with particular emphasis on visualization.

Here the basic objective was to investigate how preservation of structural similarity, as measured by proximity, could help in the selection of appropriate features. Multi-objective genetic algorithm was a tool used during optimization. Any other tool could also have served the purpose. However, in the MOGA framework the size of the chromosome in eqn. (2.7) gets limited by the cardinality D while matrix $\mathcal{P}$ is dependent on the number of patterns N. This constrains the algorithm for large data, with a complexity of $O(gS_tDN^2)$. That is one of the reasons why we used preprocessing for the high-dimensional and redundant gene expression data.

In the following chapter we present a new algorithm for feature selection based on shared nearest neighbors (SNN) distance between patterns. While here we focused on structural similarity between clusters in different feature spaces, at a global level, the SNN distance is computed between pattern pairs based on their neighborhood regions. Thereby we are able to incorporate more local information during computation of the similarity measure.

Chapter 3

Feature Selection using SNN Distance

3.1 Introduction

Similarity measures based on distance are often sensitive to the dimensionality of the pattern space [7]. The relative contrast is found to decrease, with increase in dimensionality, for a broad range of data distribution and distance measures. This, in turn, reduces the discriminatory ability of the measures [53]. As an alternative, researchers have devised a simple and common secondary similarity measure involving shared nearest neighbor (SNN) information. The SNN measure has been used in the merge step of agglomerative clustering [43,59], for clustering high dimensional data sets [31,52], and in finding outliers in subspaces of high dimensional data [68]. It is less prone to the distance concentration effect, that occurs in higher dimensions. It is also found to be more robust than primary distances, while providing better separability in the presence of several irrelevant and redundant features [53]. This observation motivated us to use the SNN distance measure as a novel evaluation criterion for our feature selection algorithm.

The algorithm aims to preserve the pairwise sample similarity in the selected feature subspace. The similarity is measured in terms of the number of patterns common to the fixed size neighborhoods of a pair of sample points, as determined by a primary distance measure like Euclidean, City block, Cosine, etc. It is a filter model which collectively evaluates a set of features. A secondary similarity between pattern pairs is computed based on a ranking of the nearest neighbors of each sample, as induced by the primary distance (or similarity). Genetic algorithm (GA) is used to traverse the search space to find an optimal feature set. GA employs an inductive learning strategy to produce a solution which is unaffected by local peaks caused by noise or interdependencies among features.

This is then extended to improve scalability in larger data. In order to overcome the bottleneck of generating a large pairwise similarity matrix, we adopt a divide-and-conquer strategy. The data is divided into nearly equal subsets, followed by a merger of those sample pairs having an SNN distance measure below some user-defined threshold within each such subset. This is followed by the selection of a feature subset, while preserving pairwise sample similarity based on SNN distance, from this merged set of patterns.

The rest of chapter is organized as follows. The concept of shared nearest neighbor (SNN) distance between points is described in Section 3.2. In Section 3.3 we present the new feature selection algorithm using the SNN distance, and its extension to accommodate the divide and merge strategy for improving scalability. The experimental results are provided in Section 3.4 on five sets of publicly available real data, *viz. MF, USPS, ORL, Spambase* and *COIL20*, along with related comparison. Finally, Section 3.5 concludes the chapter.

3.2 Shared Nearest Neighbor Distance

The most basic form of shared nearest-neighbor similarity measure is the ‘overlap’ [59]. Given a data set X consisting of $N = |X|$ sample points and $s \in \mathbb{N}^+$, where $NN_s(x) \subseteq X$ is the set of s-nearest-neighbors of $x \in X$ as determined using some specified primary distance or similarity measure, like Euclidean, city block, or cosine distance. A primary similarity measure is any function which determines a ranking of patterns relative to a query. It is not necessary for the data points to be represented as vectors. The query pertains to the s-nearest neighbors of a sample.

The 'overlap' between sample points x and y is defined to be the intersection size $SNN_s(x,y) = |NN_s(x) \cap NN_s(y)|$. It is an alternative to the traditional similarity measure, and is sometimes called as secondary similarity measure as it is based on the rankings induced by a specified primary similarity measure. The similarity measure, used here, is based on this 'overlap'. It is equivalent to the cosine of the angle between the zero-one set membership vectors for $NN_s(x)$ and $NN_s(y)$, and is defined as [31, 51]

$$simcos_s(x,y) = \frac{SNN_s(x,y)}{s}. \tag{3.1}$$

Transforming to the distance form [53], we have

$$dacos_s(x,y) = arccos(simcos_s(x,y)). \tag{3.2}$$

This distance is symmetric and satisfies the triangular inequality. Therefore, this distance is a metric [53]. There also exist other similar distance forms like linear inversion $dinv_s(x,y) = 1 - simcos_s(x,y)$ and the logarithmic form $din_s(x,y) = -\ln(simcos_s(x,y))$ [53]. However, these distances do not satisfy the triangular inequality property. All these distance functions decrease monotonically with respect to the similarity value present between the points x and y.

The feature selection algorithm based on shared nearest neighbor distance (FSSNN), outlined in the following section, employs the $dacos_s(x,y)$ distance to evaluate feature subsets. It selects a subset of features which are able to preserve the pairwise common natural grouping present in the s-size neighborhood of the original feature space.

3.3 Feature Selection

Let PDM_s be the pairwise secondary distance matrix of $N \times N$ dimension, where N is the number of patterns in a data set. We have

$$PDM_s(i,j) = dacos_s(i,j). \tag{3.3}$$

It is obvious from the definition that PDM_s is symmetric and the principal diagonal elements of PDM_s are always zero. Hence the upper triangular part of matrix PDM_s contains information about the pairwise common natural groupings of all N data points in the original feature space.

In order to reduce the computational complexity, we select those pattern pairs having $dacos_s$ values below a threshold θ (as chosen in Algorithm 3.2). Next a set X_{sel} is created, using these selected pairs. We calculate PDM_{s_1}, using s_1-nearest neighbors on set X_{sel}, with its dimension being $n \times n$ $(n = |X_{sel}|)$. Let $PDM_{s_1fsub}(i,j)$ be the pairwise secondary distance, with s_1 nearest neighbors being evaluated on set X_{sel} over f_{sub} subset of features. The f_{sub} is a reduced subset of original features and is considered for evaluation. The set of features is selected by minimizing objective function

$$F_1 = \sum_{i=1,j>i}^{i=n-1,j=n} abs(PDM_{s_1}(i,j) - PDM_{s_1 f_{sub}}(i,j)), \tag{3.4}$$

such that the similarity between pattern pair (i,j) in the original feature space gets preserved in the reduced feature space f_{sub}. Here $abs(Z)$ is the absolute value of the elements of Z. We employ GA for heuristically exploring this search space.

3.3.1 Using SNN

The feature selection Algorithm 3.1 is presented here. Its time complexity largely depends on the cost of building the dissimilarity matrix PDM_s as well as on the optimization tech-

Algorithm 3.1: **FSSNN**

```
Input: Pattern set $X$, with $N$ sample points and $D$ features.
    Neighborhoods $s$ and $s_1$.
Output: A feature subset, $f_{final}$.
1: Construct pairwise dissimilarity matrix $PDM_s$ using eqn. (3.2).
2: Choose threshold $\theta$ using Algorithm 3.2.
3: Select the pairs of points lying at a distance less than $\theta$. Construct a set $X_{sel}$ using
   these points.
4: Calculate $PDM_{s_1}$ with $s_1$-nearest neighbors on the set $X_{sel}$.
5: Select the feature subset by optimizing eqn. (3.4).
   Here GA is used as an optimization technique.
```

nique. We need $O(sN^2D)$ floating point operations for constructing PDM_s. Next we generate PDM_{s_1} for each randomly selected feature subset from the set X_{sel}. It requires $O(s_1n^2D)$ floating point operations. This is followed by optimization using eqn. (3.4). When GA is used with a population size S_t over g generations, the complexity of the optimization process becomes $O(S_tgs_1n^2D)$. Hence the overall time complexity of Algorithm 3.1 is $O((sN^2 + S_tgs_1n^2)D)$.

It may be noted that a threshold θ is used in Steps 2-3 of the algorithm. This user-defined parameter helps in reducing the computational burden on the optimization procedure, while also retaining the influential points (*hubs*) [97] which affect the reasoning procedure of nearest-neighbor computation. The heuristic for choosing θ is outlined in Algorithm 3.2.

Algorithm 3.2: **Choosing the threshold**

```
Input: The pairwise dissimilarity matrix $PDM_s$.
Output: Threshold $\theta$.
1: Find the minimum entry ($> 0$) of each row of $PDM_s$, and store as $min\_row_i$, for $i \in 1, \ldots, N-1$.
2: Choose the first percentile of these $(N-1)$ values of $min\_row_i$ as $\theta$.
```

3.3.2 Improving scalability

For datasets with large numbers of patterns we have $N >> n$, such that the overall complexity of Algorithm 3.1 tends to $O(sN^2D)$. Therefore the construction of PDM_s becomes the bottleneck. To overcome this problem the original dataset can be randomly divided into $\mathcal{T}$ number of disjoint subsets X^i of nearly equal size, with i varying from 1 to $\mathcal{T}$. Each new pairwise dissimilarity matrix PDM_s^i is generated from X^i with an appropriate θ^i. This can also be done in parallel. The proposed extension, involving divide and merge procedures, is listed as Algorithm 3.3.

The time complexity of the optimization step in Algorithm 3.3 is the same as that of Algorithm 3.1, *i.e.* $O(S_t g s_1 n^2 D)$. Now, let the size of the largest subset be n_l. So the generation of PDM_s^i involves $O(sn_l^2 D)$ floating point operations. Hence the overall time complexity of this algorithm becomes $O(\mathcal{T} sn_l^2 + S_t g s_1 n^2)D)$. Thereby, we have a gain in time complexity. This gain is enhanced if each subset X^i and its PDM_s^i are processed in parallel, such that it now becomes $O(sn_l^2 + S_t g s_1 n^2)D)$.

Algorithm 3.3: **Using Divide and Merge**

```
Input: Pattern set $X$ with $N$ sample points and $D$ features.
       Neighborhoods $s$ and $s_1$.
       Number of subsets $\mathcal{T}$.
Output: A feature subset $f_{final}$.
1: Randomly divide the data set $X$, into $\mathcal{T}$ disjoint subsets, $X^i$ such that $X = \cup_i^{\mathcal{T}} X^i$ and
   $X^{i_1} \cap X^{i_2} = \phi$ if $i_1 \neq i_2$.
2: for $i \leftarrow 1$ to $\mathcal{T}$ do
a.     Construct pairwise dissimilarity matrix $PDM_s^i$ using eqn. (3.2) on $X^i$.
b.     Select threshold $\theta^i$ using Algorithm 3.2.
c.     Select the pair of points whose distance value is less than $\theta^i$. Construct a set $X_{sel}^i$,
       of cardinality $n_i$, using these points.
3: end for
4: Construct $X_{sel}$ such that $X_{sel} = \cup_{i=1}^{\mathcal{T}} X_{sel}^i$.
5: Calculate $PDM_{s_1}$ with $s_1$-nearest neighbors on the set $X_{sel}$.
6: Select the feature subset by optimizing eqn. (3.4).
   GA is used as an optimization technique.
```

3.3.3 Using Multi-objective Optimization

Determining an appropriate formulation of a single objective function in terms of an additive combination of conflicting fitness criteria is a difficult task. Therefore, multi-objective optimization becomes necessary when it is required to arrive at a consensus solution in terms of two or more contradictory criteria. In this chapter we employ the SNN distance for feature selection in a multi-objective framework. It chooses a reduced set of features while preserving the pairwise sample similarity. A feature evaluation criterion is formulated in terms of the SNN distance, and is simultaneously optimized with the feature set cardinality. The multi-objective genetic algorithm NSGA-II of Section 1.3.2 is used, as an optimization technique, to traverse the search space and generate a non-dominated set of features. NSGA-II is a randomized search, guided by the principle of evolution and natural genetics, with a large amount of implicit parallelism.

In Section 3.3 we optimized the single objective function F_1 of eqn. (3.4). In the multi-objective framework the algorithm simultaneously reduces the size of the feature subset while preserving the pairwise topological neighbourhood information present in the s-size neighborhood in the original feature space. The second objective function is the cardinality of the reduced feature set f_{sub}, and is expressed as

$$F_2 = |f_{sub}|. \tag{3.5}$$

We employ NSGA-II [5, 22] for heuristically exploring this search space by minimizing both fitness functions.

A population of chromosomes, representing the selected feature subset, is evaluated by simultaneously optimizing the two objective functions F_1 and F_2, in order to enhance their fitness; and thereby perform feature selection.

Multi-objective GA proceeds to find a fit set of individuals (here, feature subsets) by reproducing new children chromosomes from older parents. In the process it employs the operators selection, crossover and mutation. This repeats over multiple generations (or iterations) until a stopping criterion is met. The chromosomes associated with the non-dominated solutions with respect to the fitness functions are decoded to obtain the reduced feature subsets.

The multi-objective feature selection algorithm based on shared nearest neighbors (MF-SSNN) is outlined as Algorithm 3.4.

Algorithm 3.4: **MFSSNN**

```
Input: Pattern set $X$, with $N$ sample points and $D$ features.
    Size of neighborhoods $s$ and $s_1$, cardinality of reduced set $N_{sel} = |X_{sel}|$.
Output: A feature subset, $f_{final}$.
1: Construct pairwise dissimilarity matrix $PDM_s$ using eqn. (3.2).
2: Construct a reduced set $X_{sel}$ of samples using Algorithm 3.5.
3: Calculate $PDM_{s_1}$ with $s_1$-nearest neighbors on the set $X_{sel}$.
4: Select feature subset(s) by simultaneously optimizing eqns. (3.4) and (3.5) in a multi-
   objective framework.
```

3.4 Experimental Results

The two feature selection algorithms, which preserve similarity between samples based on the shared nearest neighbors concept, were implemented on five sets of public domain data viz. *MF, USPS, ORL, Spambase,* and *COIL20*. The effectiveness of the algorithms was evaluated by externally validating selected feature subsets in terms of their predictive accuracy, as measured by well-known classifiers, like k-nearest neighbors (k-NN) and Naive Bayes (NB) [described in Section 1.2.1] with 10-fold cross validation. The process was repeated 20 times and the results were averaged for the final result. The paired Student's t-test for unequal mean and variance [4,75] was used to compute the statistical significance of the obtained results, and the threshold for rejecting the null hypothesis was set at 0.05. The feature subsets were also evaluated in terms of sample similarity and the presence of redundancy, as measured by JAC [eqn. (1.18)] and RED [eqn. (1.7)].

Since $s << N$, the SNN is found to be reasonably robust to the choice of s [53]. Here we selected $s = 50$ and $s_1 = s$ when n (*i.e.* $|X_{sel}|$) $> s$; otherwise s_1 was set to the nearest

Algorithm 3.5: **A heuristic for constructing** X_{sel}

```
Input: The pairwise dissimilarity matrix $PDM_s$ and $N_{sel}$.
Output: Reduced sample set $X_{sel}$
1: Find the minimum entry ($> 0$) of each row of $PDM_s$ and store as $min\_row_i$, with $i \in 1, \dots, N-1$.
2: Sort $min\_row_i$ in ascending order along with indices.
3: Select top $N_{sel}$ index values of $min\_row_i$.
4: Generate sample set $X_{sel}$ with these selected points.
```

integer value of 60% of n. In Algorithm 3.3 parameter $\mathcal{T}$ was selected as 10 after several experiments. Results were generated using cosine distance [eqn. (3.2)] as the primary measure. The Euclidean and City block distances were also explored, but their ranking of the patterns was found to be nearly similar.

GA has been used to optimize the evaluation criterion of eqn. (3.4) for selecting a minimal set of features. The parameter settings used were crossover probability $p_c = 0.8$, size of population $S_t = 150$, and number of generations 100, with the mutation probability p_m being varied over the generations based on a Gaussian function. The best feature subsets were selected over 30 runs.

3.4.1 Data description

The datasets used are listed here with their characteristics. *Multiple Features (MF)* dataset consists of 2000 samples from 10 classes of handwritten numerals (‘0’–‘9’), having 649 real-valued features. These are extracted from a collection of Dutch utility maps. *USPS* is a handwritten digit database. It contains 9298 handwritten images over 16 $\times$ 16 pixels, and has 10 classes. The *ORL* database consists of a total of 400 face images of 40 subjects The original images are subsampled to a size of 56 $\times$ 46 pixels, with 256 grey levels per pixel. Thus each face image can be represented by a 2576-dimensional feature vector. *COIL20* is a database of grey scale images of 20 objects, each having 72 images. The original images are subsampled down to 32 $\times$ 32 pixels, with 256 grey levels per pixel. *Spambase* data is outlined in Section 2.3.1.

3.4.2 Performance using FSSNN

The results with Algorithm 3.1, for the five datasets, is presented in Table 3.1. The cardinality of the selected feature subsets, in each case, is listed in column 2. The third and fourth column indicate the average classification accuracy involving ten-fold cross-validation, using k-NN and NB classifiers respectively. The values within parenthesis represent the standard deviations over twenty independent runs. The last two columns depict the effectiveness of the selected feature subset in preserving pairwise sample similarity, in terms of JAC, and the feature subset redundancy, in terms of RED, respectively. The last row for each dataset contains the average, cross-validated classification accuracy over the original feature space for the data.

It is observed that our algorithm provides better performance in the reduced space in most cases, as compared to that in the original feature space, while involving only about half the number of features. This is true for both classifiers k-NN and NB. Table 3.2 represents the performance of Algorithm 3.3, in a similar format, over three sets of data. Here also the classification accuracy is generally better in the reduced space. Hence the computational complexity of the resultant classifier gets further reduced.

Finally the performance of the two algorithms was compared to that of SPFS-SFS, ReliefF and SPEC, as described in Section 1.4.3. Though ReliefF is a supervised method *i.e.* it uses class label information to rank features, but it uses nearest neighbors of training samples to evalute a feature. So we consider this method to be relevant to our approach. The results are presented in Tables 3.3 and 3.5, with the best outputs marked in bold. Both Algorithms 3.1 and 3.3 provide better performance in terms of classification accuracy and the validity measures. In the case of *USPS* data, Algorithms 3.1 and 3.3 have comparable scores. It may be noted that our feature selection algorithms are unsupervised, in the sense (unlike ReliefF) that they do not use class label information during feature subset selection. Statistical significance of the classification performance of the algorithms compared was also tested. The comparative study of the execution time of Algorithms 3.1 and 3.3 was mentioned on Table 3.4 and Table 3.6 respectively. The execution time was computed on a HP Z800 workstation with Xeon(R) 2.67 GHz CPU and 16 GB RAM.

3.4.3 Performance using MFSSNN

As in the above, the algorithm was evaluated by externally validating selected feature subsets in terms of their predictive accuracy, as measured by well-known classifiers, like k-nearest neighbor (k-NN), Naive Bayes (NB) and Support Vector Machine (SVM) [described in Section 1.2.1], using 10-fold cross validation. The process was repeated 50 times and the results were averaged for the final result. The paired Student's t-test for unequal mean and variance was used to compute the statistical significance of the obtained results, and the threshold for rejecting the null hypothesis was set at 0.05. The feature subsets were also evaluated in terms of sample similarity and the presence of redundancy, as measured by JAC [eqn. (1.18)] and RED [eqn. (1.7)]. We used four sets of real data viz. *MF, USPS, ORL* and *COIL20*, whose characteristics are outlined in Subsection 3.4.1.

Since $s << N$, the performance of SNN is found to be reasonably robust to the choice of s. Here we selected s as 50 and $s_1 = s$, for $s_1 < N_{sel} < 2 * s_1$, in our experiments.

Table 3.1: Performance evaluation of Algorithm 3.1

Data Set	d	Classification Accuracy (%)		JAC	RED
		k-NN	NB		
MF	305	96.1 (0.18)	95.8 (0.12)	0.65	0.0047
$N = 2000$	313	96.7 (0.13)	96.1 (0.09)	0.67	0.0040
$D = 649$	315	96.1 (0.15)	94.7 (0.08)	0.69	0.0050
$s = 50$	317	95.2 (0.19)	96.0 (0.10)	0.68	0.0025
$s_1 = 16$	318	96.3 (0.21)	94.3 (0.09)	0.71	0.0039
$C = 10$	D	95.1	95.9	–	–
ORL	1267	97.6 (0.16)	94.7 (0.22)	0.94	0.0652
$N = 400$	1272	97.8 (0.14)	93.8 (0.36)	0.94	0.0654
$D = 2576$	1283	98.1 (0.19)	93.3 (0.39)	0.94	0.0654
$s = 50$	1287	98.1 (0.11)	93.8 (0.36)	0.93	0.0664
$s_1 = 27$	1305	97.9 (0.19)	94.5 (0.29)	0.95	0.0655
$C = 40$	D	97.8	94.1	–	–
USPS	144	96.1 (0.10)	83.4 (0.04)	0.66	0.0348
$N = 9298$	151	96.2 (0.08)	81.4 (0.06)	0.66	0.0395
$D = 256$	159	96.8 (0.06)	83.4 (0.05)	0.70	0.0366
$s = 50$	164	96.5 (0.07)	83.0 (0.05)	0.69	0.0329
$s_1 = 50$	171	96.6 (0.07)	83.1 (0.04)	0.71	0.0339
$C = 10$	D	96.9	82.5	–	–
Spambase	27	82.5 (0.27)	89.4 (0.08)	1.00	0.0099
$N = 4601$	34	81.6 (0.22)	86.6 (0.08)	1.00	0.0117
$D = 57$	35	82.1 (0.24)	87.8 (0.10)	1.00	0.0094
$s = 50$	36	81.3 (0.19)	87.2 (0.10)	1.00	0.0088
$s_1 = 50$	37	81.9 (0.26)	87.7 (0.08)	1.00	0.0055
$C = 2$	D	82.5	89.4	–	–
COIL20	480	99.9 (0.08)	92.3 (0.37)	0.81	0.0672
$N = 1440$	502	99.9 (0.09)	93.4 (0.19)	0.80	0.0674
$D = 1024$	512	99.8 (0.08)	91.7 (0.24)	0.80	0.0705
$s = 50$	541	99.9 (0.09)	92.6 (0.16)	0.84	0.0664
$s_1 = 50$	550	99.8 (0.06)	92.3 (0.30)	0.87	0.0693
$C = 20$	D	99.7	92.6	–	–

Table 3.2: Performance evaluation of Algorithm 3.3

Data Set	d	Classification Accuracy (%)		JAC	RED
		k-NN	NB		
MF	321	94.6 (0.19)	96.5 (0.09)	0.70	0.0041
$N = 2000$	323	95.5 (0.18)	95.2 (0.08)	0.71	0.0040
$D = 649$	324	93.9 (0.20)	95.3 (0.08)	0.68	0.0059
$s = 50$	326	96.3 (0.14)	94.5 (0.11)	0.70	0.0046
$s_1 = 50$	336	96.3 (0.16)	95.5 (0.09)	0.71	0.0042
$C = 10$	D	95.1	95.9	–	–
USPS	131	96.9 (0.05)	83.5 (0.04)	0.63	0.0332
$N = 9298$	144	96.5 (0.06)	83.7 (0.06)	0.64	0.0330
$D = 256$	153	96.1 (0.09)	82.4 (0.04)	0.66	0.0321
$s = 50$	155	96.6 (0.07)	83.2 (0.05)	0.68	0.0327
$s_1 = 50$	167	96.6 (0.05)	82.9 (0.05)	0.71	0.0338
$C = 10$	D	96.9	82.5	–	–
COIL20	509	99.7 (0.09)	92.8 (0.22)	0.84	0.0693
$N = 1440$	512	99.7 (0.09)	91.0 (0.43)	0.85	0.0694
$D = 1024$	527	99.8 (0.09)	92.5 (0.24)	0.85	0.0695
$s = 50$	529	99.7 (0.18)	91.9 (0.37)	0.86	0.0718
$s_1 = 50$	531	99.8 (0.09)	92.4 (0.31)	0.86	0.0686
$C = 20$	D	99.7	92.6	–	–

Table 3.3: Performance comparison of Algorithm 3.1

Data Set	d	Algorithm	Classification Accuracy (%)		JAC	RED
			k-NN	NB		
MF	317	SPFS-SFS	94.2 (0.18)	95.7 (0.08)	0.30	0.0150
		ReliefF	93.2 (0.17)	95.9 (0.08)	0.69	0.0047
		SPEC	85.3 (0.35)	94.9 (0.07)	**1.00**	0.0046
		Algorithm 3.1	**95.2** (0.19)	**96.0** (0.10)	0.68	**0.0025**
ORL	1283	SPFS-SFS	97.4 (0.16)	90.3 (0.32)	0.57	0.0774
		ReliefF	97.1 (0.14)	88.8 (0.35)	0.83	0.0669
		SPEC	96.8 (0.17)	88.4 (0.44)	0.49	0.0808
		Algorithm 3.1	**98.1** (0.19)	**93.3** (0.39)	**0.94**	**0.0652**
USPS	144	SPFS-SFS	91.4 (0.11)	67.8 (0.06)	0.30	0.0521
		ReliefF	96.6 (0.07)	**84.1** (0.03)	0.65	0.0433
		SPEC	96.7 (0.06)	84.0 (0.06)	0.66	0.0428
		Algorithm 3.1	**96.9** (0.10)	83.4 (0.04)	**0.66**	**0.0348**
Spambase	27	SPFS-SFS	82.2 (0.20)	86.6 (0.06)	0.95	0.0115
		ReliefF	80.5 (0.31)	86.1 (0.06)	0.95	0.0241
		SPEC	83.1 (0.21)	66.7 (0.12)	0.02	0.0377
		Algorithm 3.1	**82.5** (0.27)	**89.4** (0.08)	**1.00**	**0.0099**
COIL20	480	SPFS-SFS	95.0 (0.20)	78.7 (0.31)	0.23	**0.0664**
		ReliefF	99.8 (0.08)	89.0 (0.29)	0.62	0.1030
		SPEC	94.7 (0.22)	78.8 (0.47)	0.28	0.0715
		Algorithm 3.1	**99.9** (0.08)	**92.3** (0.37)	**0.81**	0.0672

Table 3.4: Execution time comparison over different datasets, using Algorithm 3.1

Data	Algorithm	Execution time (Second)
MF	SPFS-SFS	17264
	ReliefF	64
	SPEC	13
	Algorithm 3.1	633
ORL	SPFS-SFS	7930
	ReliefF	28
	SPEC	2
	Algorithm 3.1	874
USPS	SPFS-SFS	37316
	ReliefF	410
	SPEC	352
	Algorithm 3.1	179077
Spambase	SPFS-SFS	1038
	ReliefF	35
	SPEC	96
	Algorithm 3.1	2800
COIL20	SPFS-SFS	25663
	ReliefF	54
	SPEC	8
	Algorithm 3.1	1451

Table 3.5: Performance comparison of Algorithm 3.3

Data Set	d	Algorithm	Classification Accuracy (%)		JAC	RED
			k-NN	NB		
MF	321	SPFS-SFS	94.3 (0.16)	96.0 (0.07)	0.29	0.0153
		ReliefF	93.2 (0.19)	95.9 (0.06)	0.70	0.0048
		SPEC	85.4 (0.23)	94.9 (0.12)	**1.00**	0.0044
		Algorithm 3.3	**94.6** (0.19)	**96.5** (0.09)	0.70	**0.0041**
USPS	131	SPFS-SFS	88.8 (0.08)	65.3 (0.08)	0.26	0.0543
		ReliefF	96.8 (0.08)	84.0 (0.07)	0.59	0.0428
		SPEC	96.4 (0.09)	**84.4** (0.03)	0.61	0.0411
		Algorithm 3.3	**96.9** (0.05)	83.5 (0.04)	**0.63**	**0.0332**
COIL20	509	SPFS-SFS	95.3 (0.09)	80.1 (0.96)	0.25	**0.0676**
		ReliefF	**99.8** (0.09)	89.5 (0.26)	0.65	0.1017
		SPEC	95.0 (0.24)	80.2 (0.41)	0.29	0.0718
		Algorithm 3.3	99.7 (0.09)	**92.8** (0.22)	**0.84**	0.0693

Table 3.6: Execution time comparison over different datasets, using Algorithm 3.3

Data Characteristics	Algorithm	Execution time (Second)
MF	SPFS-SFS	59561
	ReliefF	64
	SPEC	13
	Algorithm 3.3	199
USPS	SPFS-SFS	29948
	ReliefF	410
	SPEC	352
	Algorithm 3.3	52053
COIL20	SPFS-SFS	26432
	ReliefF	54
	SPEC	8
	Algorithm 3.3	1266

Table 3.7: Performance comparison with related algorithms

Dataset & parameters	d	Algorithm	Accuracy (%)			JAC	RED
			k-NN	NB	SVM		
MF	155	SPFS-SFS	91.2 (0.25)	**94.9** (0.10)	83.1 (0.20)	0.90	0.0253
$N = 2000$, $D = 649$		ReliefF	83.1 (0.28)	94.0 (0.12)	84.7 (0.16)	0.90	0.0122
$C = 10$, $N_{sel} = 80$		SPEC	**96.5** (0.14)	93.3 (0.10)	**94.1** (0.11)	0.32	**0.0053**
$s = 50$, $s_1 = 50$		Algorithm 3.4	93.8 (0.21)	93.0 (0.11)	87.4 (0.15)	**0.96**	0.0107
	D	——	95.1 (0.24)	95.9 (0.15)	88.9 (0.10)	—-	—-
USPS	89	SPFS-SFS	69.5 (0.17)	52.1 (0.07)	64.0 (0.07)	0.09	0.0619
$N = 9298$, $D = 256$		ReliefF	94.8 (0.07)	82.1 (0.07)	90.2 (0.06)	0.41	0.0405
$C = 10$, $N_{sel} = 80$		SPEC	94.4 (0.09)	81.4 (0.06)	90.4 (0.06)	0.43	0.0388
$s = 50$, $s_1 = 50$		Algorithm 3.4	**95.7** (0.07)	**83.8** (0.05)	**91.4** (0.05)	**0.55**	**0.0387**
	D	——	98.3 (0.06)	82.5 (0.06)	96.2 (0.03)	—-	—-
COIL20	253	SPFS-SFS	89.8 (0.24)	63.7 (0.51)	80.2 (0.41)	0.11	**0.0633**
$N = 1440$, $D = 1024$		ReliefF	98.4 (0.22)	78.3 (0.32)	93.4 (0.29)	0.30	0.1390
$C = 20$, $N_{sel} = 80$		SPEC	84.4 (0.35)	61.9 (0.36)	73.6 (0.45)	0.12	0.0665
$s = 50$, $s_1 = 50$		Algorithm 3.4	**99.7** (0.08)	**90.7** (0.30)	**95.5** (0.14)	**0.67**	0.0737
	D	——	99.8 (0.08)	92.6 (0.40)	95.9 (0.19)	—-	—-
ORL	756	SPFS-SFS	95.6 (0.11)	89.4 (0.39)	92.5 (0.36)	0.48	0.0953
$N = 400$, $D = 2576$		ReliefF	95.4 (0.17)	81.6 (0.38)	92.6 (0.26)	0.30	0.1312
$C = 40$, $N_{sel} = 60$		SPEC	94.9 (0.17)	87.0 (0.35)	92.3 (0.39)	0.42	0.1009
$s = 50$, $s_1 = 50$		Algorithm 3.4	**97.6** (0.15)	**94.0** (0.30)	**98.0** (0.25)	**0.77**	**0.0649**
	D	——	97.8 (0.14)	94.1 (0.30)	98.1 (0.18)	—-	—-

Table 3.8: Execution time comparison over different datasets, using Algorithm 3.4

Data Characteristics	Algorithm	Execution time (Second)
MF	SPFS-SFS	6739
	ReliefF	64
	SPEC	13
	Algorithm 3.4	1009
USPS	SPFS-SFS	29214
	ReliefF	410
	SPEC	352
	Algorithm 3.4	199076
COIL20	SPFS-SFS	15624
	ReliefF	54
	SPEC	8
	Algorithm 3.4	75105
ORL	SPFS-SFS	12035
	ReliefF	28
	SPEC	2
	Algorithm 3.4	57583

Results of Table 3.7 were generated using cosine distance [eqn. (3.2)] as the primary measure. Multi-objective NSGA-II has been used to optimize the evaluation criteria of eqns. (3.4) and (3.5), for selecting the minimal set of features. As before, the parameter values were set at crossover probability $p_c = 0.8$, size of population $S_t = 100$, and number of generations $g = 300$, with the mutation probability p_m being varied over the generations based on a Gaussian function.

Fig. 3.1 depicts the Pareto optimal front, for the four datasets, using MFSSNN (Algorithm 3.4). The two objective functions F_1 and F_2, by eqns. (3.4) and (3.5), are plotted along the two axes.

We provide in Fig. 3.2 the classification accuracy (%) with respect to the cardinality of the feature subsets, over the four datasets, for the classifiers k-NN ($k = 1, 3, 5$), NB and SVM. The feature subsets from the Pareto front (of Fig. 3.1) are found to provide comparable accuracies with respect to each classifier, as depicted in the figure.

The performance of Algorithm 3.4, for the four datasets, was also compared to that of SPFS-SFS, ReliefF, and SPEC, as described in Section 1.4.3. The results are presented in Table 3.7 for a subset of features, chosen from the plots of Figs. 3.1 and 3.2. The cardinality of the reduced feature subsets, in each case, is listed in column 2. The third, fourth and fifth columns indicate the average classification accuracy involving 10-fold cross-validation, using k-NN (for $k = 1$), Naive Bayes' (NB) and Support Vector Machine (SVM) classifiers respectively. The values within parentheses represent the standard deviations over 50 independent runs. The last two columns depict the effectiveness of the selected feature subset in preserving pairwise sample similarity, in terms of JAC, and the feature subset redundancy, in terms of RED. The last row, corresponding to each dataset, contains the average, cross-validated classification accuracy over the original feature space (of cardinality D) for the data.

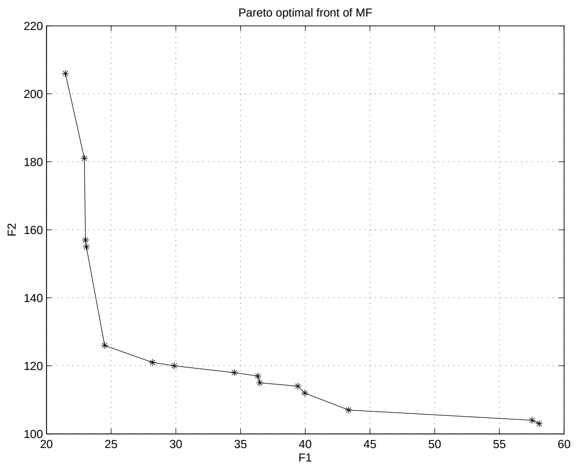
Algorithm 3.4 performs the best (as highlighted in bold in the table) over all classifiers, both in terms of accuracy and sample similarity for datasets *USPS*, *COIL20* and *ORL*. Interestingly the classification accuracy (%) is found to be comparable for feature cardinality d even with respect to the original feature space having cardinality D ($d < D$). This serves to highlight the utility of our algorithm in reducing computational complexity while providing comparable output performance. Moreover, since our algorithm is unsupervised, the efficacy of its performance becomes even more apparent (particularly in comparison to ReliefF). The comparative study of the execution time of Algorithm 3.4 was mentioned in Table 3.8. The execution time was computed on the platform mentions in Section 3.4.2.

3.5 Conclusion

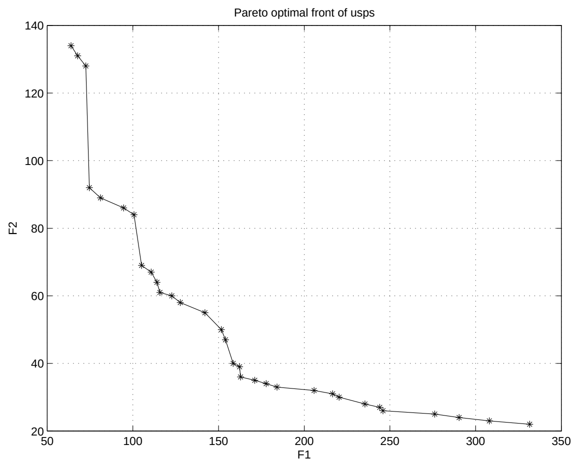
In this chapter we have developed a new unsupervised feature selection algorithm which preserves sample similarity in a reduced feature space based on the concept of shared nearest neighbor distance. The novelty of our approach lies in the effective use of SNN secondary distance (or similarity) for feature selection. The divide and merge strategy was incorporated in order to improve the scalability of the algorithm.

The results demonstrate that the selected feature subsets could not only preserve the predictive accuracy of the classifiers in the reduced feature space, but also improved a little in some of the cases. The validation indices indicated that sample similarity was also preserved in the reduced space. Both the algorithms could effectively handle the redundancy present in the feature set. Comparative study with SPFS-SFS, ReliefF and SPEC demonstrated the suitability of our algorithms.

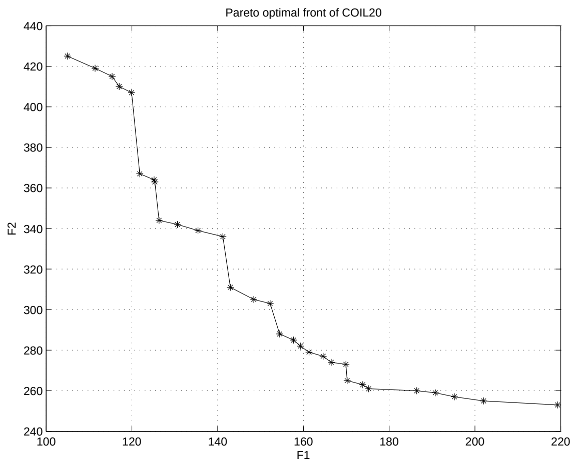
Comparing row 1 of Table 2.2 (for *Spambase* data) with row 4 of Table 3.1, we find that the SNN concept helps producing improved classifier accuracy (with both k-NN and NB) over structural similarity. This is perhaps due to the incorporation of local information from the neighborhood concept, implicit in the shared nearest neighbor distance.



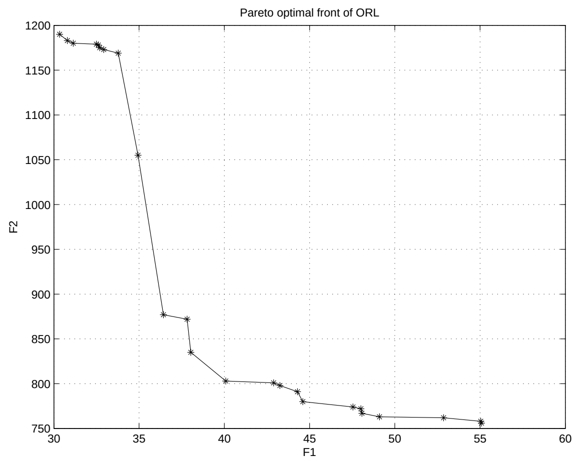
(a)



(b)

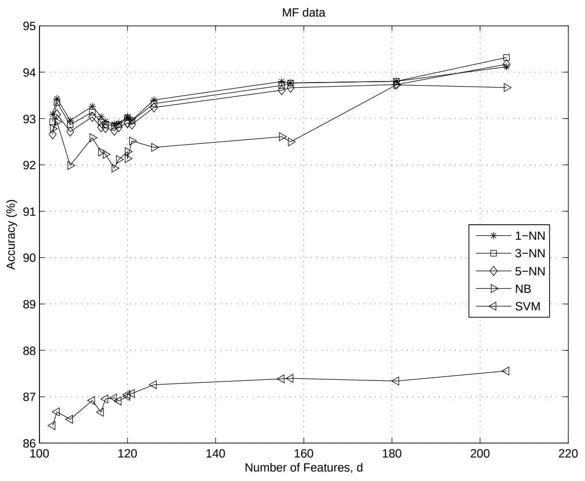


(c)

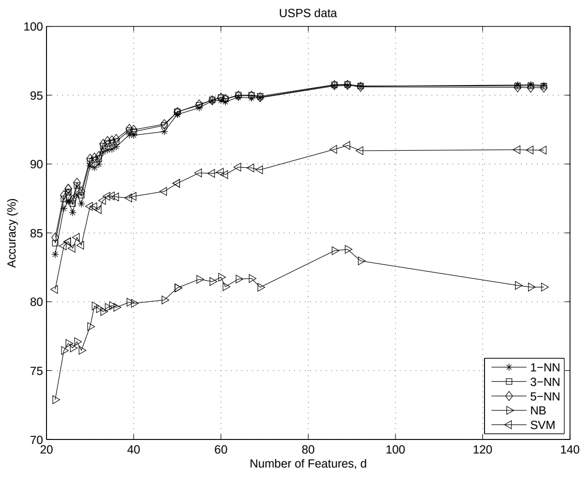


(d)

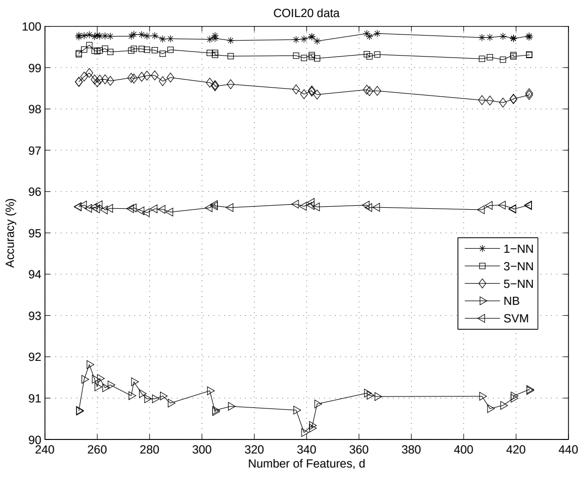
Figure 3.1: Pareto optimal front for Algorithm 3.4 over datasets (a) MF, (b) USPS, (c) COIL20, and (d) ORL



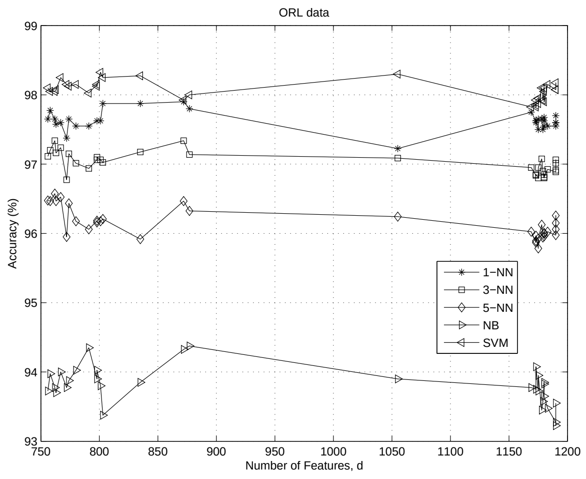
(a)



(b)



(c)



(d)

Figure 3.2: Classification performance over different feature subsets, selected from the Pareto optimal front, for datasets (a) MF, (b) USPS, (c) COIL20, and (d) ORL

This work was further extended with a multi-objective framework. The results of MF-SSNN demonstrate that the reduced feature subsets could not only preserve the predictive accuracy of the classifiers in the reduced feature space, but also improved a little (in some of the cases) with respect to the original feature space. The validity indices indicated that sample similarity was also preserved in the reduced space, with the selected features exhibiting very little correlation amongst them. This algorithm also compared with related algorithms like SPFS-SFS, ReliefF and SPEC and it demonstrated the suitability of this algorithm.

Comparing Tables 3.1, 3.7, and Fig. 3.2, we observe that the multi-objective framework of Algorithm 3.4 resulted in the selection of feature subsets having reduced cardinality, while generating comparable performance in terms of predictive accuracy and sample similarity.

Chapter 4

Feature Selection through Message Passing

4.1 Introduction

Unlike the concept of SNN distance (described in Chapter 3), here we develop an unsupervised feature selection scheme based on their similarity. It stems from the classical concept of comparing different objects to detect any hidden linear or nonlinear relationship between them. The algorithm selects a subset of features based on the internal characteristics of the data, to improve the generalization capability of a classifier. Compared to the proximity and SNN frameworks of previous chapters, the message passing scheme involves a lower time complexity. It also attaches a weight or significance for each feature with respect to the entire feature set. Features are ranked based on these weights.

The novelty of the message passing framework is that it computes the pairwise similarity between features in terms of distance correlation R, which measures the degree of all possible relationships (linear and non-linear) [110] between feature pairs without assuming any underlying distribution. This pairwise similarity is then fed into a “message passing” scheme, which selects a subset of representative features (or a set of exemplars) from the original feature set without exhaustively traversing the entire space. In this case, the cardinality of the search space is 2^D with D being the cardinality of the original feature set. It results in minimum redundancy amongst the selected set of features, coupled with reduced parameter tuning.

For datasets involving large number of instances, the computation of pairwise similarity between features using R becomes computationally intensive. In order to alleviate this problem, a dataset is randomly divided into a number of subsets and the distance covariance is independently computed on each of these subsets. These are subsequently merged, using a characteristic property of the covariance, to estimate a value of R [110]. These similarity values are again used in a message passing framework for selecting the feature subset.

The local message passing scheme starts by simultaneously considering all the features as potential representative or exemplars, and then gradually updating messages on the basis of simple formulae which search for the minima of an appropriately chosen energy function. The magnitude of each message reflects the current affinity that one feature has for choosing another feature as its exemplar. The idea is motivated by the affinity propagation algorithm for clustering in sample space [35]. Following repetitive message exchange, the feature vectors get weighted according to their representative capability. Eventually the features are selected based on these weights. The advantage of the scheme is that it does not require any exhaustive traversal of the entire search space in order to find the best subset of features. It also automatically reduces redundancy within the selected subset.

The rest of chapter is organized as follows. Section 4.2 introduces some basic concepts of distance correlation. The feature selection algorithm, based on message passing, is explained in Section 4.3. This is next extended to work on large data. The experimental results, including comparative study on nine sets of publicly-available data, *viz. Colon, Leukemia, Prostate, DLBCL, MLL, NSL-KDD, Isolet, COIL20* and *MF*, are presented in Section 4.4. Finally Section 4.5 concludes the chapter.

4.2 Distance Correlation

The distance correlation R is a relatively new approach [110]. It provides an extension to Pearson correlation [114], and measures dependence between a pair of random vectors in different types of applications. Let $\vec{f_i}$ and $\vec{f_j}$ be two feature vectors in a dataset $\vec{X}$. The distance dependence statistic is defined as follows.

Let a dataset $\vec{X}$ consist of a set of feature vectors $\vec{f_1}, \vec{f_2}, \ldots, \vec{f_D}$, with each $\vec{f_i}$ having N instances such that $\vec{f_i} = f_{i1}, f_{i2}, \ldots, f_{iN}$. The Minkowski distance matrix, of norm r, is computed for each feature vector $\vec{f_i}$ using

$$c_{gh} = |f_{ig} - f_{ih}|_r \;\; g, h = 1, \ldots, N. \tag{4.1}$$

For each $\vec{f_i}$, we have

$$C_{igh} = c_{gh} - \overline{c_{g.}} - \overline{c_{.h}} + \overline{c_{..}}, \quad g, h = 1, \ldots, N, \;\; i = 1, \ldots, D, \tag{4.2}$$

where $\overline{c_{g.}} = \frac{1}{N}\sum_{h=1}^{N} c_{gh}$, $\overline{c_{.h}} = \frac{1}{N}\sum_{g=1}^{N} c_{gh}$, $\overline{c_{..}} = \frac{1}{N^2}\sum_{g,h=1}^{N} c_{gh}$.

The distance covariance $V(\vec{f_i}, \vec{f_j})$ and distance correlation $R(\vec{f_i}, \vec{f_j})$, between feature vectors $\vec{f_i}$ and $\vec{f_j}$, are defined as [110]

$$V(\vec{f_i}, \vec{f_j}) = \frac{1}{N^2}\sum_{g,h=1}^{N} C_{igh}C_{jgh}, \tag{4.3}$$

and

$$R(\vec{f_i}, \vec{f_j}) = \begin{cases} \frac{V(\vec{f_i}, \vec{f_j})}{\sqrt{V(\vec{f_i})V(\vec{f_j})}}, & V(\vec{f_i})V(\vec{f_j}) > 0 \\ 0, & V(\vec{f_i})V(\vec{f_j}) = 0, \end{cases} \tag{4.4}$$

where the distance covariance of a feature vector $\vec{f_i}$, with itself, is given as

$$V(\vec{f_i}) = \frac{1}{N^2} \sum_{g,h=1}^{N} C_{igh}^2. \tag{4.5}$$

For all distributions with finite first moments, the coefficient $R(\vec{f_i}, \vec{f_j})$ is a standardized version of the distance covariance $V(\vec{f_i}, \vec{f_j})$. The value of R lies in the interval [0,1], such that $R = 0$ if only if $\vec{f_i}$ and $\vec{f_j}$ are independent or have no similarity between them.

This statistic is consistent for linear as well as non-linear dependence between vectors having finite second moments. It also measures nonlinear or non-monotone dependence between two feature vectors. This works well when (i) the dimension (or the number of features) D of a dataset exceeds the sample size N, or (ii) when the distributional assumptions of a dataset do not hold.

Some of the properties of distance covariance $V(\vec{f_i}, \vec{f_j})$ and distance correlation $R(\vec{f_i}, \vec{f_j})$ are listed below [110].

1. $V(\vec{f_i}, \vec{f_j}) \geq 0, \forall i, j.$
2. $V(\vec{f_i}) = 0$, if and only if every instance of the feature vector $\vec{f_i}$ is identical.
3. $0 \leq R(\vec{f_i}, \vec{f_j}) \leq 1.$
4. $R(\vec{f_i}, \vec{f_j})$ is symmetric.
5. If $(\vec{f_i^a}, \vec{f_j^a})$ and $(\vec{f_i^b}, \vec{f_j^b})$ are pairs of vectors (corresponding to the ith and jth features), over independent subsets a and b of samples, then the distance covariance follows

$$V(\vec{f_i^a} + \vec{f_i^b}, \vec{f_j^a} + \vec{f_j^b}) \leq V(\vec{f_i^a}, \vec{f_j^a}) + V(\vec{f_i^b}, \vec{f_j^b}). \tag{4.6}$$

The equality holds if and only if (i) $\vec{f_i^a}, \vec{f_j^a}, \vec{f_i^b}, \vec{f_j^b}$ are mutually independent, or (ii) $\vec{f_i^a}$ and $\vec{f_j^a}$ are both constants, or (iii) $\vec{f_i^b}$ and $\vec{f_j^b}$ are both constants.

4.3 Message Passing between Features

The algorithm initially considers a collection of real-valued similarities $sim(i, j)$ between the ith and jth feature vectors ($\vec{f_i}$, $\vec{f_j}$), $i, j \in \{1, \ldots, D\}$, where the similarity value is computed based on the distance correlation $R(\vec{f_i}, \vec{f_j})$ of eqn. (4.4). The objective is to find a set of D hidden labels $l_i; i \in 1, \ldots, D$, to map each feature vector $\vec{f_i}$ such that l_i becomes its exemplar or representative based on the similarity. The idea follows from the affinity propagation algorithm for clustering in sample space [35], and is extended here to develop a novel way of feature selection.

Each feature $\vec{f_i}$ is thus mapped to its most similar feature, on the basis of R. During message update, a feature $\vec{f_i}$ can not directly choose itself as its own representative. Only when some other feature $\vec{f_{i'}}$ has already chosen $\vec{f_i}$ as its representative, can feature $\vec{f_i}$ choose itself as its representative; and such a set of choices is called a valid mapping. Let a set of valid mappings for the D features be $\mathbf{L} = \{l_1, \ldots, l_D\}$, and $sim(i, l_i)$ be the similarity of the ith feature vector $\vec{f_i}$ to its representative label l_i. The energy (or cost) of a set of valid mappings is $E(\mathbf{L}) = -\sum_{i=1}^{D} sim(i, l_i)$. The goal of the feature selection algorithm is to search for the minima of this energy (or cost) function. The exact minimization of the function being computationally intractable [14], researchers have developed update rules [35] for searching the minima of the energy (or cost) function based on the Bethe free energy approximation [121].

4.3.1 Selection of features

The update rules for feature selection are based on marginal probabilities (or belief), and proceed recursively by exchanging two kinds of messages between a feature and its representative. The responsibility $res(i, k)$, sent from feature vector $\vec{f_i}$ to candidate representative feature vector $\vec{f_k}$, reflects the accumulated evidence regarding how well-suited

Algorithm 4.1: **Similarity between features**

```
Input: $N \times D$ data matrix $\vec{X}$.
Output: $D \times D$ similarity matrix $sim$.
1: for $i \leftarrow 1$ to $D$ do
2:     for $j \leftarrow i$ to $D$ do
           Compute distance covariance $V(\vec{f_i}, \vec{f_j})$ between features $\vec{f_i}$ and $\vec{f_j}$ using eqns. (4.1)-(4.3).
           Evaluate distance correlation $R(\vec{f_i}, \vec{f_j})$ using eqn. (4.4), and store in $sim(i,j)$ and $sim(j,i)$.
3:     end for
4: end for
5: $sim = -1 * sim$.
```

$\vec{f_k}$ is to serve as the representative for feature $\vec{f_i}$ (taking into account its other potential representatives). The availability $avl(i,k)$, sent from representative feature vector $\vec{f_k}$ to feature vector $\vec{f_i}$, measures the accumulated evidence about how appropriate it is for $\vec{f_i}$ to choose $\vec{f_k}$ as its representative (taking into account the support from other features regarding whether it should be the representative).

The procedure starts with the pairwise similarity between features sim as the input preference, for a feature to select another feature as its representative. The pairwise similarities are calculated using Algorithm 4.1.

The responsibility $res(i,k)$ and availability $avl(i,k)$ are defined as [35]

$$res(i,k) \leftarrow sim(i,k) - max_{k' \in \{1,\dots,D\}; k' \neq k}\{avl(i,k') + sim(i,k')\}, \tag{4.7}$$

and

$$avl(i,k) \leftarrow min\{0, res(k,k) + \sum_{i'; i' \notin \{i,k\}} max\{0, res(i',k)\}\}. \tag{4.8}$$

The self-availability $avl(k,k)$ of a feature k is updated as

$$avl(k,k) \leftarrow \sum_{i';i' \notin \{i,k\}} max\{0, res(i',k)\}. \tag{4.9}$$

To avoid numerical oscillation, the messages (res or avl) are damped before moving to the next phase (or iteration). We use

$$msg \leftarrow (1-\lambda) * msg + \lambda * msg_{old}, \tag{4.10}$$

where $0 < \lambda < 1$ is a damping factor, msg represents either $res(i,k)$ or $avl(i,k)$ [eqns. (4.7)-(4.9)] in the current message, and msg_{old} is the corresponding message computed in the previous iteration. The energy is computed by summing the diagonal elements $res(k,k)$ and $avl(k,k)$ of the matrices. The procedure for feature selection through message passing (FSMP) is summarized in Algorithm 4.2.

The complexity of Algorithms 4.1 and 4.2 are $O(D^2 * N^2)$ and $O(iter * D^2)$ respectively. While for $N \ll D$ the complexity of Algorithm 4.1 tends to $O(D^2)$, it approaches $O(N^2)$ for $N \gg D$. Therefore, for large values of N this can lead to a bottleneck.

4.3.2 Extension to large data

To apply distance correlation R on a data with large samples, the property #5 of this measure [from eqn. (4.6)] is used. It provides an upper-bound on the actual value of the distance covariance $V(\vec{f_i}, \vec{f_j})$ of $\vec{X}$.

In the feature selection scenario, the data is randomly divided into div disjoint subsets such that (i) $\vec{X} = \cup_t^{div} \vec{X}^t$ and (ii) $\vec{X}^{t_1} \cap \vec{X}^{t_2} = \phi$ if $t_1 \neq t_2$. The distance covariance $V(\vec{f_i^t}, \vec{f_j^t})$, over each subset $\vec{X}^t; t = 1, \ldots, div$, can now be computed parallelly using eqn. (4.3). The final value of distance covariance $V(\vec{f_i}, \vec{f_j})$ over $\vec{X}$ becomes

$$V(\vec{f_i}, \vec{f_j}) = \sum_{t=1}^{div} V(\vec{f_i^t}, \vec{f_j^t}). \tag{4.11}$$

The procedure to compute similarity between features is outlined in Algorithm 4.3. The similarity matrix sim is then fed into Algorithm 4.2.

The complexity of Algorithm 4.3 becomes $O(D^2 * N_t^2 * div)$, where N_t is the sample cardinality of the subset $\vec{X}^t$. Therefore for large data, using $N_t \ll N$ with a moderately large value of div, this algorithm has an advantage over Algorithm 4.1. Moreover, the complexity can be further reduced by modeling each disjoint subset in a parallel framework when it tends to $O(D^2 * N_t^2)$.

4.4 Experimental Study

The feature selection algorithms were implemented on nine real life datasets, *viz. Colon, Leukemia, DLBCL, Prostate, MLL, NSL-KDD, Isolet, COIL20,* and *MF,* whose characteristics are listed below.

Leukemia dataset contains the gene expression information of 72 acute leukemia samples. There are 25 human acute myeloid leukemia (AML) and 47 acute lymphoblastic leukemia (ALL) cases, each with 7129 features. *Prostate* data includes the gene expression measurements for 52 prostate tumors and 50 adjacent normal prostate tissue samples, over 12626 features. *DLBCL* data contains 77 gene expression levels. Among them, 58 are of diffuse large B-cell lymphoma (DLBCL) type while 19 are of follicular lymphoma (FL) type. This data consists of 7070 features. *MLL* data is a collection of 72 gene expression measurements. There are 24 examples of acute lymphoblastic leukemia (ALL), 20 examples of mixed-lineage leukemia (MLL) and 28 examples of acute myeloid leukemia (AML), over 12533 features. *NSL-KDD* is a benchmark data representing intrusion related information for network-based IDs. The data has 148517 instance from two classes (*viz.* normal and anomaly) over 42 features. The *Colon* and *Isolet* datasets are described in Section 2.3.1, while *MF* and *COIL20* are presented in Section 3.4.1.

As before, the efficacy of the feature selection algorithm was demonstrated by externally validating the selected feature subsets in terms of their predictive accuracy, as measured by well-known classifiers like k-nearest neighbors (k-NN), Naive Bayes (NB), and Support Vector Machine (SVM) [described in Section 1.2.1], using 10-fold cross validation. The results were averaged over 50 runs. The paired Student's t-test for unequal mean and variance was used to compute the statistical significance of the obtained results, and the threshold for rejecting the null hypothesis was set at 0.05.

The experiments were conducted in two parts - first with the datasets having smaller sample size, followed by those with relatively larger cardinality (N). Results were compared with those from related feature selection algorithms, *viz.* fsfs and mRMR, and independence criterion HSIC (as described in Section 1.4.3). The distance correlation R between every feature pair was computed using eqn. (4.4) on the entire dataset, and stored in the similarity matrix sim. Here $r = 1$ was used in eqn. (4.1). The values of damping factor λ and iteration $iter$ were chosen as 0.5 and 100, respectively, after several experiments.

4.4.1 Algorithms 4.1 and 4.2

The algorithms were tested on the five datasets, *Colon*, *Leukemia*, *Prostate*, *DLBCL* and *MLL*. The classification accuracy of the classifiers k-NN $(k = 1, 3, 5)$, NB and SVM, over the reduced sets of features, was plotted in Fig. 4.1. It is observed, in most cases, that for a small number of features $(d \sim 50)$ we obtain around 90% accuracy for *Colon*, *DLBCL* and *MLL* data. In case of *Prostate* data the classification performance is found to decrease as $d \to D$.

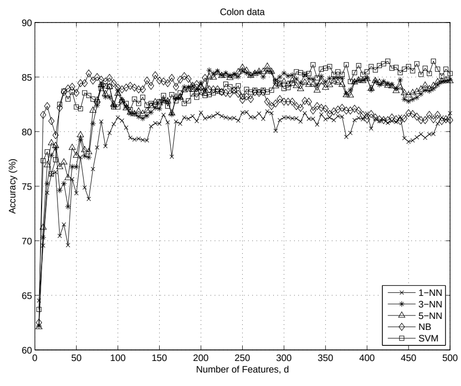
A comparative study of the classification performance, over the feature set extracted by Algorithms 4.1 and 4.2, was made with that by fsfs, mRMR and HSIC. Note that HSIC was used in the framework of Algorithm 4.2. The results are provided in Table 4.1 for classifiers k-NN $(k = 1)$, NB and SVM. The cardinality of the selected feature subsets, in each case, is listed in column 2. The last row (corresponding to each data) indicates the performance over the entire feature set of cardinality D. The best results (among the feature selection methods compared) are indicated in bold. The reduced set of features selected by Algorithms (4.1, 4.2) [FSMP] provides the best results in most cases. With respect to the performance over the original feature space (D), the accuracy is found to be comparable (over the reduced set d) in most cases. Specifically, our algorithm obtains the best overall results (at $d \ll D$) with classifiers NB (for *Colon*, *DLBCL*, *MLL*), and with k-NN and SVM (for *Prostate*). This highlights the usefulness of the FSMP scheme in selecting an appropriate set of reduced features for good decision making.

The comparison of classifier accuracy for FSMP, HSIC and mRMR, over *Prostate, MLL,* and *Colon* datasets, is depicted in Figs. 4.2 - 4.4 respectively. In general, FSMP is found to provide better performance over different feature sets of the data. However for *Colon* data FSMP shows higher classifier accuracy with SVM, only over smaller feature subsets.

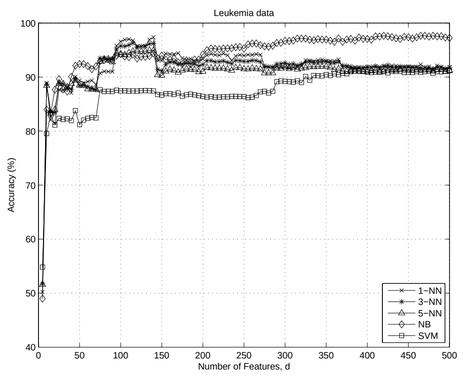
A comparative study of the execution time, for selection of a set of 500 features by Algorithms 4.1 and 4.2 and with that by fsfs, mRMR and HSIC, was listed in Table 4.2. FSMP took less time with respect to mRMR and HSIC over all data sets. FSMP performed faster in comparison with fsfs on Prostate Data. Over other datasets, it consumed comparable time. The execution time was computed on a HP Z800 workstation with Xeon(R) 2.67 GHz CPU and 16 GB RAM.

4.4.2 Algorithms 4.3 and 4.2

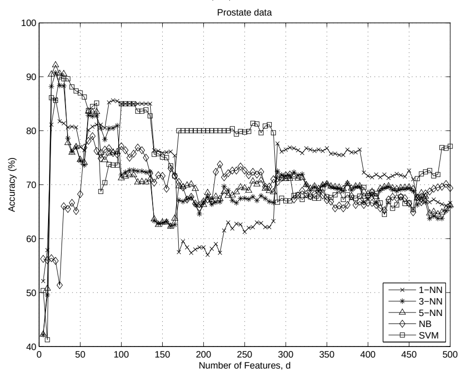
Next the four larger datasets, *NSL-KDD, Isolet, COIL20* and *MF*, were handled in the framework of Algorithms 4.2 and 4.3. The plots in Fig. 4.5 depict the results, over different choices of div, for these data. In most cases a larger div, signifying smaller subsets, gives



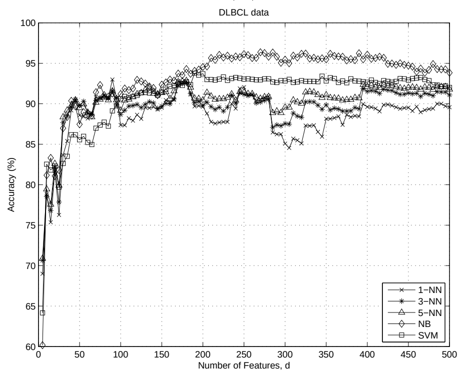
(a)



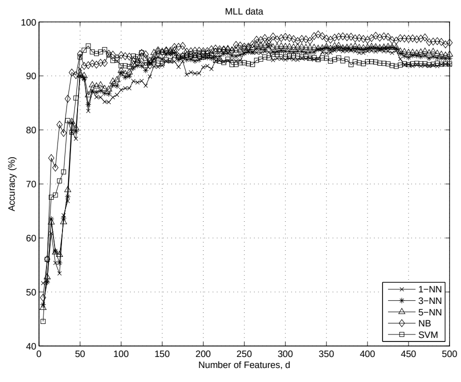
(b)



(c)

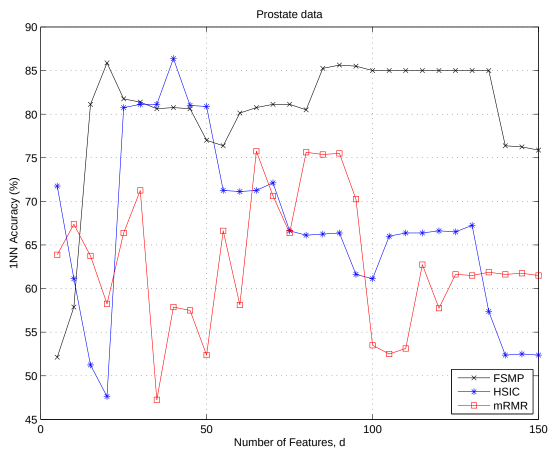


(d)

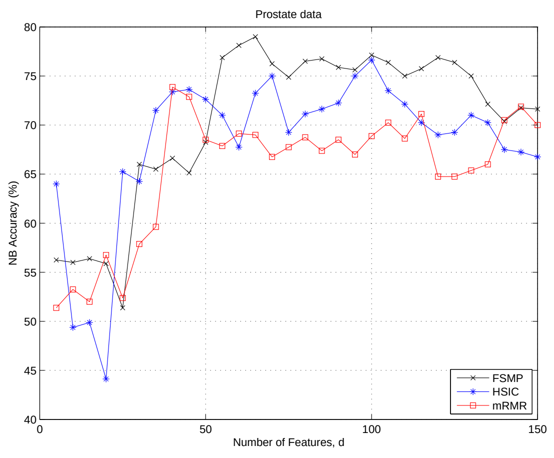


(e)

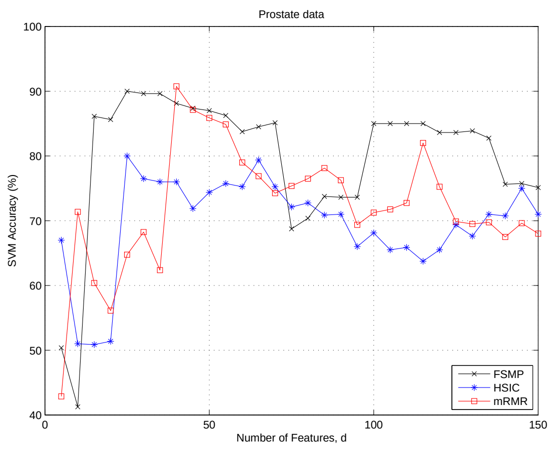
Figure 4.1: Classification performance over different feature subsets, selected using FSMP (4.1, 4.2), for datasets (a) Colon, (b) Leukemia, (c) Prostate, (d) DLBCL, and (e) MLL



(a)

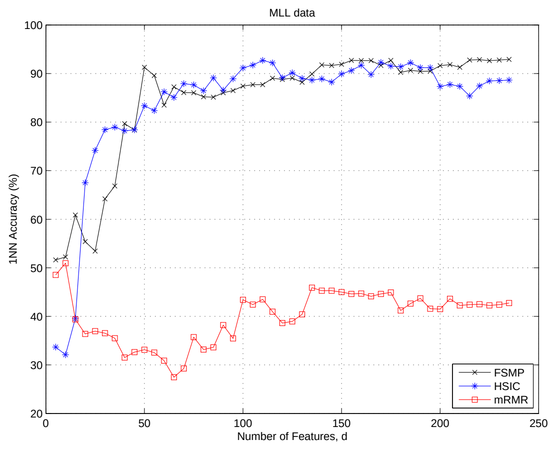


(b)

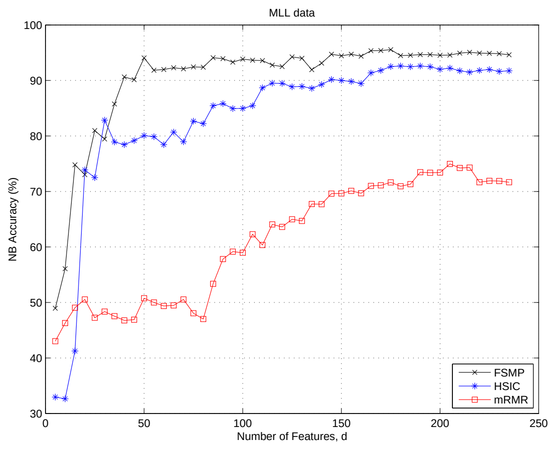


(c)

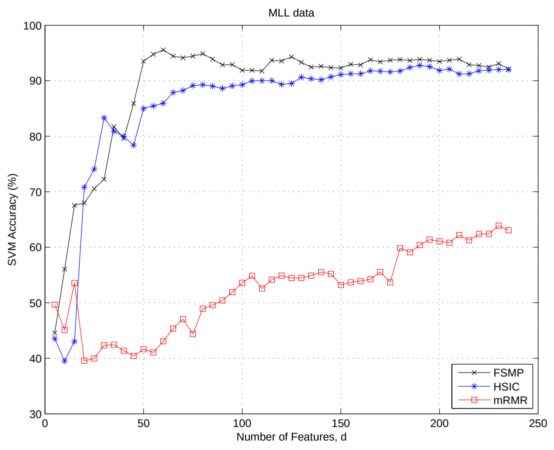
Figure 4.2: Performance comparison of FSMP (4.1, 4.2), mRMR and HSIC, over Prostate dataset, using classifiers (a) k-NN, (b) NB, and (c) SVM



(a)

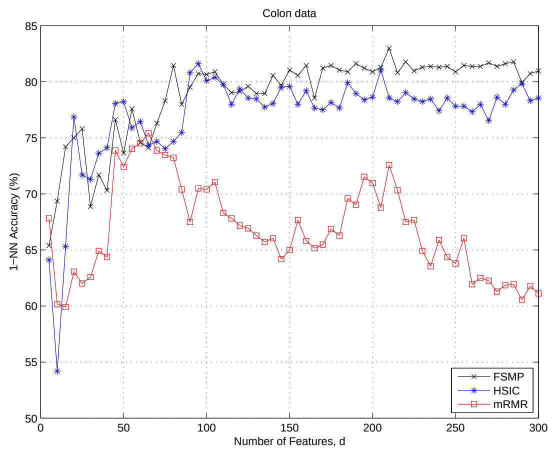


(b)

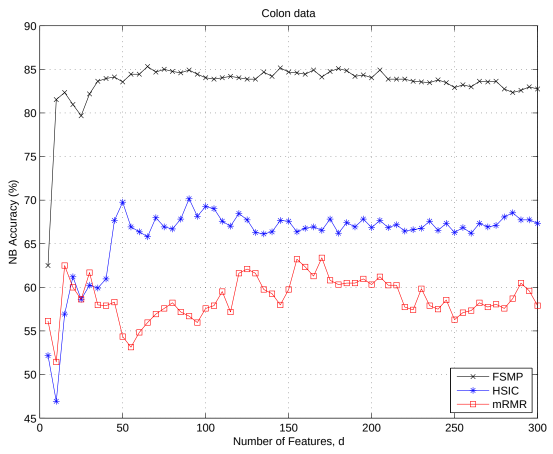


(c)

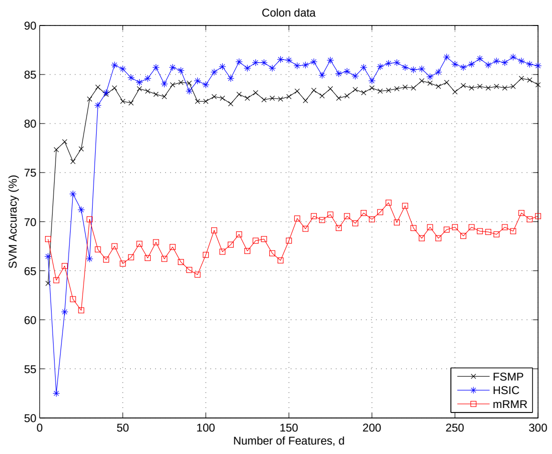
Figure 4.3: Performance comparison of FSMP (4.1, 4.2), mRMR and HSIC, over MLL dataset, using classifiers (a) k-NN, (b) NB, and (c) SVM



(a)



(b)



(c)

Figure 4.4: Performance comparison of FSMP (4.1, 4.2), mRMR and HSIC, over Colon dataset, using classifiers (a) k-NN, (b) NB, and (c) SVM

Table 4.1: Performance comparison over different data, using FSMP (4.1 and 4.2)

Data Characteristics	d	Algorithm	Accuracy (%) (standard deviation)		
			k-NN	NB	SVM
Colon	45	fsfs	70.8 (1.91)	64.2 (1.33)	78.1 (1.89)
$D = 2000$		HSIC	**78.0** (2.50)	66.9 (1.41)	**85.0** (1.56)
$N = 62$		mRMR	74.8 (1.36)	57.6 (3.95)	67.1 (3.24)
$C = 2$		FSMP (4.1, 4.2)	77.8 (1.41)	**84.2** (1.19)	84.1 (1.41)
	D	——	79.6 (1.13)	62.3 (1.45)	84.0 (1.74)
Leukemia	10	fsfs	53.9 (3.11)	60.3 (2.76)	63.3 (3.53)
$D = 7129$		HSIC	61.5 (1.80)	50.8 (3.14)	58.3 (2.43)
$N = 72$		mRMR	59.7 (1.83)	69.4 (1.65)	67.8 (2.53)
$C = 2$		FSMP (4.1, 4.2)	**88.5** (1.81)	**83.8** (0.82)	**80.0** (0.82)
	D	——	91.4 (1.73)	86.3 (1.98)	94.8 (0.67)
Prostate	20	fsfs	72.0 (2.99)	**70.0** (2.77)	73.8 (2.75)
$D = 12626$		HSIC	48.5 (3.28)	43.8 (5.34)	52.0 (2.51)
$N = 20$		mRMR	57.3 (4.00)	57.3 (5.60)	54.8 (6.17)
$C = 2$		FSMP (4.1, 4.2)	**85.8** (2.45)	55.2 (4.13)	**85.8** (1.83)
	D	——	61.0 (2.53)	72.3 (3.30)	72.0 (2.52)
DLBCL	40	fsfs	80.3 (1.35)	79.8 (1.43)	80.1 (1.28)
$D = 7070$		HSIC	88.3 (1.86)	85.3 (1.21)	92.5 (1.45)
$N = 77$		mRMR	73.0 (1.37)	79.1 (1.88)	79.7 (1.22)
$C = 2$		FSMP (4.1, 4.2)	**89.5** (1.55)	**90.5** (1.74)	**85.6** (1.22)
	D	——	90.3 (1.10)	75.1(1.90)	97.2 (0.74)
MLL	50	fsfs	62.6 (1.90)	66.4 (1.10)	65.7 (2.70)
$D = 12533$		HSIC	83.4 (1.22)	79.2 (1.13)	85.0 (1.82)
$N = 72$		mRMR	33.8 (2.72)	51.0 (3.14)	40.4 (5.53)
$C = 3$		FSMP (4.1, 4.2)	**91.0** (1.88)	**93.6** (1.34)	**93.2** (1.02)
	D	——	92.0 (1.18)	90.1 (1.95)	95.5 (0.50)

Table 4.2: Execution time comparison over smaller data, using FSMP (4.1 and 4.2)

Data	Algorithm	Execution time (Second)
Colon	fsfs	234
	HSIC	723
	mRMR	5320
	FSMP (4.1, 4.2)	308
Leukemia	fsfs	2377
	HSIC	185030
	mRMR	53444
	FSMP (4.1, 4.2)	6020
Prostate	fsfs	53074
	HSIC	18872
	mRMR	136983
	FSMP (4.1, 4.2)	5512
DLBCL	fsfs	2726
	HSIC	186840
	mRMR	97301
	FSMP (4.1, 4.2)	6520
MLL	fsfs	9240
	HSIC	248910
	mRMR	278700
	FSMP (4.1, 4.2)	22088

better results. As before, we obtain good accuracy at $d < D$. Since the processing over each div can be performed parallelly in Algorithm 4.3, it thereby serves to speed up the process of decision making with considerably lower number of features.

Table 4.3 demonstrates a comparative study using the three classifiers. The results for Algorithms (4.2, 4.3) [FSMP] correspond to a sample value of div, as indicated in the second-last row for each data. Since $N_t \sim N/div$, therefore the complexity reduces to $O(iter * D^2) + O(D^2 * N_t^2)$ when each subset $\vec{X}_t$ can be processed in parallel. Due to the high time and space complexity of HSIC, it was infeasible to be implemented here. The mRMR algorithm could not be used on *NSL-KDD* because of its high requirement of system memory. In most cases, FSMP (algorithms 4.2, 4.3) was found to provide best and/or comparable classification performance. For the largest data *NSL-KDD*, FSMP fared better in the much smaller reduced feature space ($d = 20, N_t \sim 297$), as compared to $D = 42, N = 148517$ of the original feature space. This underlines the usefulness of algorithm FSMP for effectively handling larger data. The supervised algorithm mRMR fared better than our algorithm in a few cases, *viz. Isolet* (all classifiers) and *COIL20* (SVM). However, our unsupervised algorithms generally resulted in comparable performance.

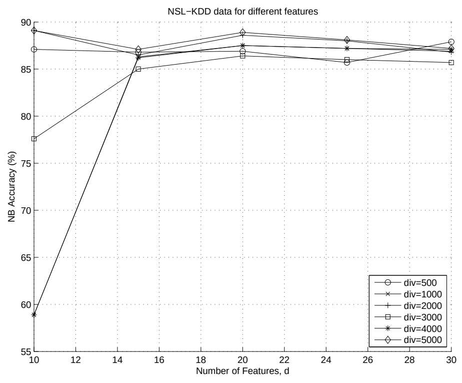
Table 4.4 demonstrates a comparative study of execution time over larger datasets using FSMP (4.2 and 4.3) with other methods. The execution time was computed on the platform mentioned in Section 4.4.1. The results depict that the computation time of FSMP is more over all datasets. The reason behind this is the computation of the pair-wise similarity computation procedure between the feature pairs, which is still the bottle-neck of the method.

Table 4.3: Performance comparison over larger data, using FSMP (4.2 and 4.3)

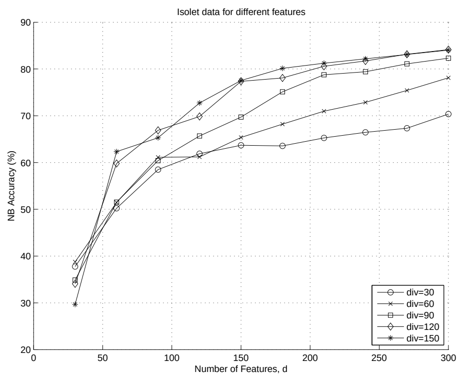
Data Characteristics	d	Algorithm	Accuracy (%) (standard deviation)		
			k-NN	NB	SVM
NSL-KDD	20	fsfs	98.5 (0.01)	60.4 (0.01)	**94.7** (0.30)
$D = 42$, $N = 148517$		$div = 500$ $N_t \sim 297$ FSMP (4.2, 4.3)	**99.0** (0.02)	**86.9** (0.00)	93.2 (0.13)
$C = 2$	D	——	99.3 (0.01)	86.8 (0.02)	49.4 (0.58)
Isolet	250	fsfs	65.5 (0.24)	56.5 (0.10)	70.7 (0.18)
$D = 617$		mRMR	**87.0** (0.16)	**84.0** (0.09)	**91.2** (0.05)
$N = 7797$, $C = 26$		$div = 150$ $N_t \sim 52$ FSMP (4.2, 4.3)	86.0 (0.19)	83.0 (0.09)	90.0 (0.02)
	D	——	96.9 (0.08)	85.1 (0.10)	98.9 (0.03)
COIL20	300	fsfs	95.8 (0.21)	73.0 (0.57)	84.6 (0.20)
$D = 1024$		mRMR	99.4 (0.17)	87.4 (0.34)	**94.0** (0.18)
$N = 1440$, $C = 20$		$div = 20$ $N_t \sim 72$ FSMP (4.2, 4.3)	**99.6** (0.09)	**89.0** (0.25)	93.0 (0.14)
	D	——	99.7 (0.07)	92.6 (0.16)	95.9 (0.11)
MF	150	fsfs	**96.0** (0.10)	90.7 (0.11)	91.0 (0.14)
$D = 649$		mRMR	84.6 (0.14)	94.0 (0.55)	83.0 (0.09)
$N = 2000$, $C = 10$		$div = 50$ $N_t \sim 40$ FSMP (4.2, 4.3)	95.0 (0.22)	**94.0** (0.54)	**92.4** (0.09)
	D	——	95.0 (0.28)	95.9 (0.22)	88.9 (0.12)

Table 4.4: Execution time comparison over larger data, using FSMP (4.2, 4.3)

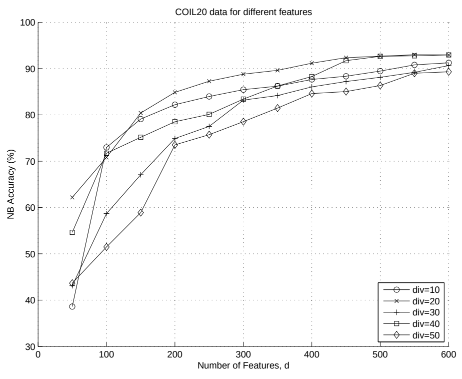
Data	Algorithm	Execution time (Second)
NSL-KDD	fsfs	12
	FSMP (4.2, 4.3)	1796
Isolet	fsfs	148
	mRMR	45
	FSMP (4.2, 4.3)	3543
COIL20	fsfs	127
	mRMR	182
	FSMP (4.2, 4.3)	1603
MF	fsfs	57
	mRMR	55
	FSMP (4.2, 4.3)	762



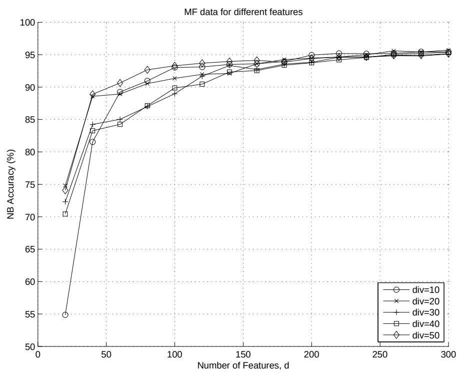
(a)



(b)



(c)



(d)

Figure 4.5: Classification performance of classifier NB, over different feature subsets selected by FSMP (4.2, 4.3), for different number of subsets (div), over datasets (a) NSL-KDD, (b) Isolet, (c) COIL20, and (d) MF

Table 4.5: Performance and execution time comparison between Algorithms PR, FSSNN, MFSSNN and FSMP

Data Set	Algorithm	d	Accuracy (%)			Execution time
			k-NN	NB	SVM	(Second)
Colon	PR	261	83.9	54.8	64.5	18000
	FSSNN	981	79.3	62.4	64.5	714
	MFSSNN	457	78.1	70.0	79.4	2603
		433	78.5	66.5	78.1	
	FSMP (4.1, 4.2)	261	**84.0**	**83.4**	83.7	308
	Original space	2000	79.6	62.3	**84.0**	——
Isolet	PR	274	77.5	84.8	94.8	86052
	FSSNN	348	88.5	88.2	93.4	15907
	MFSSNN	300	88.7	87.4	93.2	99145
		121	85.6	85.0	90.8	
	FSMP (4.2, 4.3)	274	86.2	85.2	94.0	3543
	Original space	617	96.9	85.1	98.9	——

Table 4.6: Performance comparison between Algorithms FSSNN, MFSSNN and FSMP

Data Set	Algorithm	d	Accuracy (%)	
			k-NN	NB
MF	FSSNN	305	96.1	95.8
	MFSSNN	206	94.1	93.7
		103	93.1	92.8
	FSMP (4.2, 4.3) div = 50	100	93.2	93.4
	FSMP (4.2, 4.3) div = 10	305	**97.3**	**96.0**
	Original space	649	95.1	95.9
COIL20	FSSNN	480	99.9	92.3
	MFSSNN	425	99.8	91.2
		253	99.7	90.7
	FSMP (4.2, 4.3) div = 20	250	99.3	87.3
	FSMP (4.2, 4.3) div = 20	480	**99.9**	**92.7**
	Original space	1024	99.7	92.6

4.4.3 Comparative analysis

Comparing with the proximity framework of model PR from Chapter 2, over datasets *Colon* and *Isolet* and model FSSNN and MFSSNN from Chapter 3, we observe from Table 4.5 that the message passing scheme (FSMP) always selects a better subset of features to provide higher classifier accuracy using k-NN, SVM and NB over Colon Data. Here, the algorithm FSMP leads to an improved classification, with just 261 features, with respect to the original feature space of cardinality 2000. This allows reduction in computational complexity along with enhanced performance. We also observe FSMP take least execution time to produce a feature subset. In case of Isolet data, the feature set (cardinality 348) generated by MFSSNN produces highest accuracy.

Next, we compare the shared nearest neighbor approach of Chapter 3 (Algorithms FSSNN and MFSSNN), with FSMP in Table 4.6. Parameters used for Algorithms FSSNN and MFSSNN are the same as indicated in Tables 3.1 and 3.7, respectively. Two subsets of features were selected from the Pareto front of MFSSNN, viz. those corresponding to the highest and lowest cardinality, respectively. Here Table 4.6 illustrates the overall best performance of FSMP, for both datasets *MF* and *COIL20*, using classifiers k-NN and NB. In fact, algorithm FSMP is found to be always better in terms of classifier accuracy over the reduced feature subset (involving lower computational complexity) as compared to that in original feature space. This serves to validates the effectiveness of the message passing scheme.

For $N >> D$, the time complexity of FSMP is $O(N_t^2 * div)$ while for Algorithm PR it is $O(gS_tN^2)$. Here g is the number of generations and S_t is the population size of the GA. The gain is obvious because, typically, $N_t < N$ and $div < gS_t$. In case of parallel computation of sim, the time complexity of FSMP can even approach $O(N_t^2)$. The time complexity of FSSNN becomes $O(sN^2 + S_tgs_1n^2)$, where s and s_1 correspond to the nearest neighbors and n is the cardinality of the selected pattern set. Here also, FSMP gains over FSSNN. Algorithm MFSSNN has higher time complexity than FSSNN, and is thereby worse than FSMP.

For $N << D$, the time complexity of FSMP is $O(iter * D^2)$ and for PR it is $O(gS_tD)$. But the value of the $(g * S_t)$ term is generally on the higher side, given that the total number of feature subsets is 2^D. Therefore, FSMP has an advantage over PR. The time complexity of FSSNN becomes $O(S_tgs_1n^2D)$ in this scenario. Thereby, again FSMP has an advantage over both FSSNN and MFSSNN.

4.5 Conclusion

In this chapter we have developed a new unsupervised feature selection framework, which identifies a subset of representative features by passing messages between them. The message passing scheme, adapted from affinity propagation for clustering [35], computed the pairwise similarity between features in terms of distance correlation R. It measured the degree of all possible relationships between feature pairs, without assuming any underlying distribution.

The algorithm was able to select a reduced set of features without exhaustively traversing the entire search space. One of the characteristic properties of R was utilized, to make the algorithm viable for handling data with large number of samples. The computational complexity was also reduced. Comparative study with related algorithms like fsfs, mRMR, and HSIC dependence criterion, demonstrated the suitability of the algorithms on publicly-available datasets.

Algorithm 4.2: **FSMP**

```
Input: $D \times D$ feature similarity matrix $sim$, damping factor $\lambda$, number of iterations $iter$, and
    cardinality of output feature set $d$.
Output: A feature subset $\{\vec{\mathcal{G}}_d\}$.
 1: Initialize messages with pairwise similarity between features.
 2: Initialize $avl(i,k) = res(i,k) = 0$, $\forall i, k$.
 3: for $n \leftarrow 1$ to $iter$ do
 4:     for $i \leftarrow 1$ to $D$ do
 5:         for $k \leftarrow 1$ to $D$ do
                Update responsibility $res(i,k)$ using eqn. (4.7).
                Damp $res(i,k)$ using eqn. (4.10).
 6:         end for
 7:     end for
 8:     for $i \leftarrow 1$ to $D$ do
 9:         for $k \leftarrow 1$ to $D$ do
10:             if $i \neq k$ then
                    Update availability $avl(i,k)$ using eqn. (4.8).
11:             else
                    Update self-availability $avl(k,k)$ using eqn. (4.9).
12:             end if
                Damp availability $avl(i,k)$ using eqn. (4.10).
13:         end for
14:     end for
15: end for
16: Compute energy (or cost) values of $D$ features by summing $res(k,k)$ and $avl(k,k)$, for $k \leftarrow 1$
    to $D$.
17: Sort features on the basis of energy (or cost) value. Select top $d$ features to constitute $\{\vec{\mathcal{G}}_d\}$.
```

Algorithm 4.3: **Similarity between features for large data**

```
Input: $N \times D$ data matrix $\vec{X}$, the number of subsets $div$.
Output: $D \times D$ similarity matrix $sim$.
1: Randomly divide $\vec{X}$ into $div$ disjoint subsets $\vec{X}^t$.
2: for $i \leftarrow 1$ to $D$ do
3:     for $j \leftarrow i$ to $D$ do
           Compute distance covariance $V(\vec{f}_i^t, \vec{f}_j^t)$, over all $div$ subsets $\vec{X}^t$, using eqns. (4.1)-(4.3).
           Compute $V(\vec{f}_i, \vec{f}_j)$ using eqn. (4.11).
           Evaluate distance correlation $R(\vec{f}_i, \vec{f}_j)$ using eqn. (4.4), and store in $sim(i, j)$ and $sim(j, i)$.
4:     end for
5: end for
6: $sim = -1 * sim$.
```

Chapter 5

Conclusions and Scope for Further Research

This chapter concludes the thesis and summarizes some open issues for future research.

5.1 Conclusions

In recent times data encompasses high dimensionality, or large number of features which may, again, include both relevant as well as irrelevant, redundant and noisy ones. In order to perform efficient machine learning or pattern recognition, data preprocessing becomes a necessity. Feature selection is one such commonly used technique. It determines a subset of the original set of attributes to enhance the comprehensibility of a model that describes a dataset. But data does not always come with labels. Due to the rapid generation of data, labeling may not always be possible by experts. Unsupervised technique is useful in this scenario. It uses the intrinsic properties of data to select an appropriate subset of features. Similarity is an important intrinsic property, which is employed in various manners in this thesis.

In every chapter we have presented conclusions drawn from respective methodologies developed, and the experimental results therein. Here we consolidate them to provide an overall picture of the contributions of the thesis.

The thesis dealt with certain tasks in unsupervised feature selection. It encompassed selecting feature subsets by (i) preserving structural similarity in terms of proximity, (ii) determining sample similarity in terms of shared nearest neighbors, and (iii) using distance correlation in a message passing framework. The effectiveness of the different methodologies and their statistically significant comparative study with related ones, were extensively demonstrated on several real life datasets from varied domains (like population census, computer intrusion and genomic analysis) involving dimensions ranging from 3 to 12626 and samples ranging from 20 to 148517.

The thesis consists of six chapters. Chapter 1 introduced the basics of pattern recognition and soft computing, followed by a detailed coverage of the task of feature selection. This was followed by the scope of the thesis.

Chapter 2 considered the structural similarity between patterns, in terms of fuzzy proximity relations, to select the important features. An objective function was generated to preserve structural similarity between the original and reduced feature spaces at a global level. The cardinality of the reduced feature space was minimized while maintaining high proximity. Since the two objective functions were mutually conflicting, a multi-objective framework was employed to resolve the issue. Feature subsets were selected from the Pareto optimal front. The use of soft computing helped producing acceptable solutions in the presence of uncertainty. However, there was a drawback in capturing local information during the computation of proximity, and the computational complexity was also high for large data.

The concept of preservation of pattern pair similarity between the original and reduced feature spaces, at a local level, was employed in Chapter 3 for unsupervised feature selection. A secondary SNN distance measure was used to compute the pairwise sample similarity in terms of their shared nearest neighbors. GA was used as an optimization tool. A divide-and-conquer strategy was incorporated to extend the algorithm to work for data involving large number of samples, in a scalable manner. The data was randomly partitioned into nearly equal subsets, followed by a merger of the sample pairs having an SNN distance measure below some user-defined threshold within each such subset. Finally a feature subset was selected from this merged set of patterns, while preserving the pairwise sample similarity based on SNN distance.

Comparing Table 2.2 with Table 3.1 for *Spambase* data, the efficacy of the SNN concept could be established in terms of improved classifier accuracy (with both k-NN and NB) over structural similarity (in PR from Chapter 2). This was due to the incorporation of local information from the neighborhood concept, implicit in the shared nearest neighbor distance.

The SNN similarity was used in Subsection 3.3.3, along with feature cardinality, in a multi-objective framework. The reduced set of samples, chosen to preserve sample similarity, helped in reducing the effect of outliers on the feature selection procedure while also decreasing computational complexity. Comparing Tables 3.1, 3.7, and Fig. 3.2, it was observed that the multi-objective framework of MFSSNN resulted in the selection of feature subsets having reduced cardinality, while generating comparable performance in terms of predictive accuracy and sample similarity.

Chapter 4 introduced the affinity propagation framework, in a novel manner, for feature selection. The message passing scheme worked on pairwise feature similarity, which was computed using distance correlation. The methodology was extended to large datasets utilizing one of the intrinsic properties of distance correlation. Considering Table 4.5 we infer that FSMP always selected a better subset of features to provide higher classification accuracy using k-NN, SVM and NB (with respect to Algorithm PR). In case of *Colon* data the algorithm FSMP resulted in an improved classification, involving 261 features, with respect to the original feature space of cardinality 2000. Table 4.6 validated the overall best performance of FSMP with respect to Algorithms FSSNN and MFSSNN (Chapter 3), for both datasets *MF* and *COIL20*, using classifiers k-NN and NB. In fact, algorithm FSMP was found to be always better, in terms of classifier accuracy, over the reduced feature subset as compared to that in original feature space. This serves to justify the effectiveness of the message passing scheme. Time complexity analysis also established the superiority of FSMP.

5.2 Scope for Further Research

Although we have restricted the application of structural similarity concept in this thesis to numeric attributes, the proximity approach could be extended to include mixed data by incorporating medoids and considering a symbolic framework for computing the cluster prototypes. However, the proximity measure has a bottle-neck in case of data involving larger number of patterns. This can be effectively handled by a divide-and-conquer modularization strategy involving some collaboration amongst independent smaller subsets of patterns.

The shared neighborhood concept could also be extended to include other kinds of attributes (like symbolic, categorical, hybrid), given that the SNN distance is based on the ranking of sample points induced by some primary distance measure. Other types of SNN distances, like linear inverse or logarithmic forms, could also be employed. Moreover, since genetic algorithm may not always be computationally efficient, some other specialized optimization technique may be designed by embedding the properties of SNN distance. These aspects are currently under investigation. Analogously, in Subsection 3.3.3, a divide and conquer strategy can be incorporated in the multi-objective MFSSNN algorithm.

Similarities between the objects play an important role in the affinity propagation framework. We plan to incorporate some other concepts like rank correlation, while modifying the message passing equations to make this robust. Fuzzy relations will also be applied to choose the exemplar.

Flourishing of the social media, explosion in the amount of data collected from sensors and machine-to-machine interactions, and reduction in the cost of storage media, has caused the creation of large volumes of data. Big data has five main characteristics like volume, velocity, variety, veracity and complexity. Data is streaming in at an unprecedented speed and must be dealt with in a timely manner. Reacting quick enough to deal with such high data velocity is a challenge for most organizations. Data today comes in all types of formats - structured, numeric data in traditional databases, as well as unstructured text documents, email, video, audio, stock ticker data and financial transactions. Managing, merging and governing different varieties of data is something many analysts continue to grapple with. Some investigations on feature selection is also planned in the context of big data.

Data flows can be highly inconsistent, with periodic peaks. Daily, seasonal and event-triggered peak data loads can be challenging to manage. Today's data comes from multiple sources. And it is still an undertaking to link, match, cleanse and transform data across systems. However it is necessary to connect and correlate relationships, hierarchies and multiple data linkages, as otherwise the data can quickly spiral out of control. The real issue is not to acquire large amounts of data but to manage such data.

The challenges are to obtain data from any source, harness relevant data and analyze it to find answers that enable 1) cost reduction, 2) time reduction, 3) new product development and optimized offerings, and 4) smarter business decision making. A number of recent technology advancements like cheap storage, faster processors, affordable open source and distributed platform e.g. Hadoop, parallel processing, large grid environments with high connectivity and high throughputs, cloud computing, and other flexible resource allocation arrangements, are enabling analysts and researchers to harness the big data for relevant decision making. We will explore such framework for extending the algorithms developed in this thesis.

Bibliography

[1] C. W. Ahn. *Advances in Evolutionary Algorithms: Theory, Design and Practice*. Springer, London, 2006.

[2] M. Alwadi and G. Chetty. A novel feature selection scheme for energy efficient wireless sensor networks. In *Proceedings of International Conference of Algorithms and Architectures for Parallel Processing*, pages 264–273, 2012.

[3] M. Ashraf, G. Chetty, D. Tran, and D. Sharma. Hybrid approach for diagnosing thyroid, hepatitis, and breast cancer based on correlation based feature selection and Naïve Bayes. In *Proceedings of International Conference of Neural Information Processing*, pages 272–280, 2012.

[4] A. Aspin. Tables for use in comparisons whose accuracy involves two variances. *Biometrika*, pages 245–271, 1949.

[5] M. Banerjee, S. Mitra, and H. Banka. Evolutionary-rough feature selection in gene expression data. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 37:622–632, 2007.

[6] A. Ben-Hur, A. Elisseeff, and I. Guyon. A stability based method for discovering structure in clustered data. In *Proc. of Pacific Symposium on Biocomputing*, pages 6–17, 2002.

[7] K. S. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When is "nearest neighbor" meaningful? In *Proceeding of International Conference on Database Theory (ICDT)*, pages 217–235, 1999.

[8] J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press, New York, 1981.

[9] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, Singapore, 2006.

[10] A.L. Blum and P. Langley. Training a 3-node neural networks is NP-complete. *Neural Networks*, 5:117–127, 1992.

[11] D. Cai, C. Zhang, and X. He. Unsupervised feature selection for multi-cluster data. In *Proceedings of Knowledge Discovery and Data Mining (KDD'10)*, pages 333–342, 2010.

[12] J. Casillas, O. Cordón, M. J. Del Jesus, and F. Herrera. Genetic feature selection in a fuzzy rule-based classification system learning process for high-dimensional problems. *Information Sciences*, 136:135–157, 2001.

[13] Y. Censor. Pareto optimality in multiobjective problems. *Applied Mathematics and Optimization*, 4:41–59, 1977.

[14] M. Charikar, S. Guha, É. Tardos, and D. B. Shmoys. A constant factor approximation algorithm for *k*-median problem. *Lecture Notes in Computer Science*, 5788:25–33, 2009.

[15] R. Chitta, R. Jin, T. C. Havens, and A. K. Jain. Approximate kernel k-means: Solution to large scale kernel clustering. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 895–903, 2011.

[16] C. A. Coello Coello. An updated survey of GA-based multiobjective optimization techniques. *ACM Computer Survey*, 32:109–143, 2000.

[17] C. A. Coello Coello. Evolutionary multi-objective optimization: Some current research trends and topics that remain to be explored. *Frontiers of Computer Science in China*, 3:18–30, 2009.

[18] P. Daniušis and P. Vaitkus. Supervised feature extraction using Hilbert-Schmidt norms. *Journal of Computer and System Sciences*, 65:129–149, 2009.

[19] M. Dash and H. Liu. Feature selection for clustering. In *Proceedings of 4th Pacific Asia Conference on Knowledge Discovery and Data Mining*, pages 110–121. Springer-Verlag, 2000.

[20] M. Dash, H. Liu, and J. Yao. Dimensionality reduction for unsupervised data. In *Proc. of the Nineteenth IEEE International Conference on Tools with AI*, pages 532–539, Newport Beach, CA , USA, 1997.

[21] R. Datta, D. Joshi, J. Li, and J. Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computer Survey*, 40(2):5:1–5:60, 2008.

[22] K. Deb, S. Agarwal, A. Pratap, and T. Meyarivan. A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6:182–197, 2002.

[23] M. Devaney and A. Ram. Efficient feature selection in conceptual clustering. In *Proc. of the Fourteenth International Conference on Machine learning*, pages 92–97, Nashville, TN, 1997.

[24] P. A. Devijver and J. Kittler. *Pattern Recognition: A Statistical Approach*. Prentice Hall, Englewood Cliffs, 1982.

[25] M. M. Deza and E. Deza. *Encyclopedia of Distances*. Springer Verlag, Berlin, 2013.

[26] C. Ding and H. C. Peng. Minimum redundancy feature selection from microarray gene expression data. In *Proc. of 2nd IEEE Computational Systems Bioinformatics Conf.*, pages 523–528, Newport Beach, CA , USA, 2003.

[27] F. Douglas. Knowledge acquisition via incremental conceptual clustering. *Machine learning*, 2:139–172, 1987.

[28] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley, New Jersey, 2001.

[29] J. G. Dy and C. E. Brodley. Feature subset selection and order identification for unsupervised learning. In *Proceedings of the 17th International Conference on Machine learning*, pages 247–254, Morgan Kaufmann, 2000.

[30] J. G. Dy and C. E. Brodley. Feature selection for unsupervised learning. *Journal of Machine Learning Research*, 5:845–889, 2004.

[31] L. Ertoz, M. Steinbach, and V. Kumar. Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data. In *Proceedings of SIAM International Conference on Data Mining (SDM'03)*, pages 333–352. 2003.

[32] B. S. Everritt. *Cluster Analysis*. John Wiley & Sons, New York, 1974.

[33] J. Fan, R. Samworth, and Y. Wu. Ultrahigh dimensional feature selection: Beyond the linear model. *Journal of Machine Learning Research*, 10:2013–2038, 2009.

[34] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy. *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press, Menlo Park, CA, 1996.

[35] B. J. Fery and D. Dueck. Clustering by passing messages between data points. *Science*, 315:972–976, 2007.

[36] G. Forman. An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research*, 3:1289–1305, 2003.

[37] E. B. Fowlkes and C. L. Mallowes. A method to compare two hierarchical clusterings. *Journal of the American Statistical Association*, 78:553–569, 1983.

[38] A. L. N. Fred and A. K. Jain. Data clustering using evidence accumulations. In *Proceedings of ICPR'02*, pages 276–280, 2002.

[39] K. Glocer, D. Eads, and J. Theiler. Online feature selection for pixel classification. In *Proceedings of the 22nd International Conference on Machine learning*, ICML '05, pages 249–256, 2005.

[40] D. E. Goldberg. *Genetic Algorithms: Search, Optimization and Machine Learning*. Addision-Wesley, Reading M. A., 1989.

[41] A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf. Measuring statistical dependence with Hilbert-Schmidt norms. *Lecture Notes on Artificial Intelligence*, 3734:63–77, 2005.

[42] M. Grineva, M. Grinev, and D. Lizorkin. Extracting key terms from noisy and multitheme documents. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, pages 661–670, 2009.

[43] S. Guha, R. Rastogi, and K. Shim. CURE: An efficient clustering algorithm for large databases. In *Proceedings of SIGMOD Conference'98*, pages 73–84, 1998.

[44] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[45] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machine. *Machine Learning*, 46:389–422, 2002.

[46] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA data mining software: An update. *SIGKDD Explorations*, 11(1), 2009.

[47] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, San Mateo, CA, 2000.

[48] S. Haykin. *Neural Networks- A Comprehensive Foundation*. Prentice-Hall, New Jersy, USA, 1999.

[49] X. He, D. Cai, and P. Niyogi. Laplacian score for feature selection. In *Proceedings of Neural Information Processing Systems (NIPS'05)*, 2005.

[50] Y. He, K. Fataliyev, and L. Wang. Feature selection for stock market analysis. In *Proceedings of International Conference on Neural Information Processing*, pages 737–744, 2013.

[51] M. E. Houle. Navigating massive datasets via local clustering. In *Proceeding of Knowledge Discovery and Data Mining (KDD)*, pages 333–352. 2003.

[52] M. E. Houle. The relevant-set correlation model for data clustering. *Stat. Anal. Data Min.*, 1(3):157–176, 2008.

[53] M. E. Houle, H. P. Kriegel, P. Kröger, E. Schubert, and A. Zimek. Can shared-neighbor distances defeat the curse of dimensionality? In *Proceedings of 22nd international Conference on Scientific and Statistical Database Management (SSDBM)*, 2010.

[54] L. Hubert and P. Arabie. Feature selection: Evaluation, application, and small sample performance. *Journal of Classification*, pages 193 – 218, 1985.

[55] H. Ishibuchi and A. Miyazaki. Determination of inspection order for classifying new samples by neural networks. In *Proceedings of IEEE - International Conference on Neural Networks*, pages 2907–2910, Orlando, USA, 1994.

[56] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, Englewood Cliffs, 1988.

[57] A. K. Jain, R. P. W. Duin, and J. Mao. Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:4–37, 2000.

[58] A. K. Jain and D. E. Zongker. Feature selection: Evaluation, application, and small sample performance. *IEEE Trans. Pattern Analysis Machine Intelligence*, 19:153–158, 1997.

[59] R. A. Jarvis and E. A. Patrick. Clustering using a similarity measure based on shared near neighbors. *IEEE Transactions on Computers*, C-22:1025–1034, 1973.

[60] Y. Jiang and J. Ren. Eigen sensitive feature selection. In *Proceedings of 28th International Conference on Machine Learning*, pages 89–96, 2011.

[61] L. Kanal. Patterns in pattern recognition. *IEEE Transactions on Information Theory*, 20:697–722, 1974.

[62] M. Kantardzic. *Data Mining: Concepts, Models, Methods, and Algorithms*. Wiley Interscience, IEEE Press, NJ, 2002.

[63] R. L. Kennedy, Y. Lee, B. van Roy, C. D. Reed, and R. P. Lippman. *Solving Data Mining Problems Through Pattern Recognition*. Prentice Hall, NJ, 1998.

[64] B. King. Step-wise clustering procedures. *Journal of American Statistical Association*, 69:86–101, 1967.

[65] K. Kira and L. Rendell. A practical approach to feature selection. In D. Sleeman and P. Edwards, editors, *Proceedings of International Conference on Machine Learning*, pages 368–377, Aberdeen, July 1992. Morgan Kaufmann.

[66] R. Kohavi and G. John. Wrappers for feature selection. *Artificial Intelligence*, 97:273–324, 1997.

[67] A. Konak, D. W. Coit, and A. E. Smith. Multi-objective optimization using genetic algorithns: A tutorial. *Reliability Engineering and System Safety*, 91:992–1007, 2006.

[68] H. P. Kriegel, P. Kröger, E. Schubert, and A. Zimek. Outlier detection in axis-parallel subspaces of high dimensional data. In *Proceedings of Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD'09)*, pages 831–838, 2009.

[69] P. P. Kundu and S. Mitra. Feature selection through message passing. *Information Sciences (Under Revision)*.

[70] P. P. Kundu and S. Mitra. Feature selection using multi-objective optimization and shared nearest neighbor distance. *Applied Soft Computing (Communicated)*.

[71] P. P. Kundu and S. Mitra. Feature selection using shared nearest neighbor distance. *IEEE Transactions on Systems, Man and Cybernetics, Part B (Communicated)*.

[72] P. P. Kundu and S. Mitra. Multi-objective evolutionary feature selection. In *Proceedings of Lecture Notes in Computer Science 5909*, pages 74–79. Springer, 2009.

[73] P. Langley. Selection of relevant features in machine learning. In *Proceedings of the AAAI Fall Symposium on Relevance*, pages 140–144. AAAI Press, 1994.

[74] W. Lee, S. J. Stolfo, and K. W. Mok. Adaptive intrusion detection: A data mining approach. *Artificial Intelligence Review*, 14(6):533–567, 2000.

[75] E. L. Lehmann. *Testing of Statistical Hypothesis*. John Wiley, New York, 1976.

[76] H. Liu, E. R. Dougherty, J. G. Dy, K. Torkkola, E. Tuv, H. Peng, C. H. Q. Ding, F. Long, M. E. Berens, L. Parsons, Z. Zhao, L. Yu, and G. Forman. Evolving feature selection. *IEEE Intelligent Systems*, 20:64–76, 2005.

[77] H. Liu, H. Motoda, R. Setiono, and Z. Zhao. Feature selection: An ever evolving frontier in data mining. In *JMLR : Workshop and Conference Proceedings : The Fourth Workshop on Feature Selection in Data Mining*, pages 4–13, 2010.

[78] H. Liu and L. Yu. Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on Knowledge Data Engineering*, 17:491–502, 2005.

[79] U. V. Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17:395–416, 2007.

[80] H. Mannila. Theoretical frameworks for data mining. *SIGKDD Explorations*, 1:30–32, 2000.

[81] M. Meila. Comparing clusterings – An information based distance. *Journal of Multivariate Analysis*, 98:873–895, 2007.

[82] T. M. Mitchell. *Machine Learning*. McGraw-Hill International Eds., New Delhi, 1997.

[83] P. Mitra, C. A. Murthy, and S. K. Pal. Unsupervised feature selection using feature similarity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:301–312, 2002.

[84] S. Mitra and T. Acharya. *Data Mining: Multimedia, Soft Computing, and Bioinformatics*. John Wiley, London, 2003.

[85] S. Mitra, P. P. Kundu, and W. Pedrycz. Feature selection using structural similarity. *Information Sciences*, 198:48–61, 2012.

[86] C. A. Murthy and S. Pradhan. Metric in feature space. In S. Chaudhury, S. Mitra, C. A. Murthy, P. S. Sastry, and S. K. Pal, editors, *Proceedings of Pattern Recognition and Machine Intelligence*, volume 5909 of *LNCS*, pages 50–55. Springer Verlag, Berlin, 2009.

[87] P. M. Narendra and K. Fukunaga. A branch and bound algorithm for feature subset selection. *IEEE Transactions on Computers*, 26(9):917–922, 1977.

[88] S. A. Nene, S. K. Nayar, and H. Murase. Columbia Object Image Library (COIL-20). Technical Report No. CUCS-006-96, Dept. of Computer Science, Columbia University, New York, 1996.

[89] K. Ng and H. Liu. Customer retention via data mining. *AI Review*, 14:590, 1999.

[90] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2-3):103–134, 2000.

[91] G. Obozinski and B. Taskar. Multi-task feature selection. Technical report, University of California, Berkeley, 2006.

[92] S. K. Pal and B. Chakraborty. Fuzzy set theoretic measure for automatic feature evaluation. *IEEE Transactions on Systems, Man, and Cybernetics*, 16:754–760, 1986.

[93] S. K. Pal and S. Mitra. *Neuro-fuzzy Pattern Recognition: Methods in Soft Computing*. John Wiley, New York, 1999.

[94] W. Pedrycz, V. Loia, and S. Senatore. P-FCM: A proximity-based fuzzy clustering. *Fuzzy Sets and Systems*, 148:21–41, 2004.

[95] H. Peng, F. Long, and C. Ding. Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:1226–1238, 2005.

[96] P. Pudil, J. Novovicov$\acute{a}$, and J. Kittler. Floating search methods in feature selection. *Pattern Recognition Letters*, 15:1119–1125, 1994.

[97] M. Radovanović, A. Nanopoulos, and M. Ivanović. Hubs in space: Popular nearest neighbors in high dimensional data. *Journal of Machine Learning Research*, 11:2487–2531, 2010.

[98] N. Ramakrishnan and A. Y. Grama. Data mining: From serendipity to science. *IEEE Computer*, 34:34–37, 1999.

[99] W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of American Statistics Association*, 66:846–850, 1971.

[100] M. L. Raymer, M. F. Punch, E. D. Goodman, L. A. Kuhn, and A. K. Jain. Dimensionality reduction using ganetic algorithm. *IEEE Transactions on Evolutionary Computation*, 4:164–171, 2000.

[101] C. J. V. Rijsbergen. *Information Retrieval*. Butterworths, London, 2nd edition, 1979.

[102] M. Robnik-Sikonja and I. Kononenko. Theoretical and empirical analysis of ReliefF and RReliefF. *Machine Learning*, 53:23–69, 2003.

[103] P. Rousseeuw. Silhouette: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987.

[104] S. Rovetta and F. Masulli. An experimental validation of some indexes of fuzzy clustering similarity. In *Fuzzy Logic and Applications*, volume 5571 of *Lecture Notes in Computer Science*, pages 132–139. Springer Berlin / Heidelberg, 2009.

[105] D. W. Ruck, S. K. Rogers, and M. Kabrisky. Feature selection using a multilayer perceptron. *Neural Network Computing*, 20:40–48, 1990.

[106] A. Saxena, N. R. Pal, and M. Vora. Evolutionary methods for unsupervised feature selection using Sammon's stress function. *Fuzzy Information Engineering*, 3:229–247, 2010.

[107] W. W. Siedlecki and J. Sklansky. A note on genetic algorithms for large-scale feature selection. *Pattern Recognition Letters*, 10:335–347, 1989.

[108] A. M. Silva, W. M. Caminhas, A. P. Lemos, and F. Gomide. Evolving neural fuzzy network with adaptive feature selection. In *Proceedings of International Conference on Machine Learning and Applications*, pages 440–445, 2012.

[109] L. Song, A. Smola, A. Gretton, J. Bedo, and K. Borgwardt. Feature selection via dependence maximization. *Journal of Machine Learning Research*, pages 1393–1434, 2012.

[110] G. J. Székely and M. L. Rizzo. Brownian distance covariance. *The Annals of Applied Statistics*, 3:1236–1265, 2009.

[111] L. Talavera. Dependency-based feature selection for clustering symbolic data. *Intelligent Data Analysis*, 4:19–28, 2000.

[112] J. Tang, X. Hu, H. Gao, and H. Liu. Unsupervised feature selection for multi-view data in social media. In *Symposium on Data Mining*, pages 270 – 278, 2013.

[113] J. Tang and H. Liu. Coselect: Feature selection with instance selection for social media data. In *Symposium on Data Mining*, pages 695 – 708, 2013.

[114] S. Theodoridis and K. Koutroumbas. *Pattern Recognition*. Elsevier, California, USA, 2009.

[115] L. Wang, N. Zhou, and F. Chu. A general wrapper approach to selection of class-dependent features. *IEEE Transactions on Neural Networks*, 19:1267–1278, 2008.

[116] H. L. Wei and S. A. Billings. Feature subset selection and ranking for dimensionality reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29:162–166, 2007.

[117] J. Weston, A. Elisseff, B. Schoelkopf, and M. Tipping. Use of the zero norm with linear models and kernel methods. *Journal of Machine Learning Research*, 3:1439–1461, 2003.

[118] E. P. Xing, M. I. Jordan, and R. M. Karp. Feature selection for high-dimensional genomic microarray data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 601–608, 2001.

[119] J. H. Yang and V. Honaver. Feature subset selection using a genetic algorithm. *IEEE Intelligent Systems*, 13:44–49, 1998.

[120] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pages 412–420, 1997.

[121] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory*, 51:2282–2312, 2005.

[122] N. Zhao and L. Wang. Class-dependent feature selection for face recognition. In *Proceedings of International Conference on Neural Information Processing*, pages 551–558, 2008.

[123] Z. Zhao and H. Liu. Spectral feature selection for supervised and unsupervised learning. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 1151–1157, 2007.

[124] Z. Zhao and H. Liu. Multi-source feature selection via geometry-dependent covariance analysis. *JMLR Workshop and Conference Proceedings: New Challenges for Feature Selection in Data Mining and Knowledge Discovery*, 4:36–47, 2008.

[125] Z. Zhao, J. Wang, H. Liu, J. Ye, and Y. Chang. Identifying biologically relevant genes via multiple heterogeneous data sources. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, pages 839–847, 2008.

[126] Z. Zhao, L. Wang, and H. Liu. Efficient spectral feature selection with minimum redundancy. In *Proceedings of Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI'10)*, 2010.

[127] Z. Zhao, L. Wang, H. Liu, and J. Ye. On similarity preserving feature selection. *IEEE Transactions on Knowledge and Data Engineering*, 25:619–632, 2013.

[128] J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani. 1-norm support vector machines. In *Proceeding of Advances in Neural Information Processing System 16*, 2003.

List of Publications of the Author Related to the Thesis

- S. Mitra, P. P. Kundu and W. Pedrycz, Feature selection using structural similarity, *Information Sciences*, Volume 198, 2012, pages 48-61.
- P. P. Kundu and S. Mitra, Feature selection using shared nearest neighbor distance, *Expert System with Applications* (communicated).
- P. P. Kundu and S. Mitra, Feature selection through message passing, *IEEE Transactions on Cybernetics* (communicated).
- P. P. Kundu and S. Mitra, Feature selection using multi-objective optimization and shared nearest neighbor distance, *Applied Soft Computing* (Under Revision).
- P. P. Kundu and S. Mitra, Multi-objective Evolutionary Feature Selection, *Lecture notes in Computer Science 5909* (S. Chaudhury, S. Mitra, C. A. Murthy, P. S. Sastry, and S. K. Pal Eds.), Springer, Pages 74-79, 2009.