Role of Feature Selection in Data Filtering: A Comparative Analysis.

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering
in
Computer Science & Engineering

By:
Ms. Kamlesh Dhayal
(80732029)

Under the supervision of:
Mrs. Shalini Batra
Sr. Lecturer, CSED

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004

JUNE 2009

Certificate

I hereby certify that the work which is being presented in the thesis entitled, **"Role of Feature selection in Data Filtering: A Comparative Analysis"**, in partial fulfillment of the requirements for the award of degree of Master of Engineering in Computer Science submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of Mrs. Shalini Batra and refers other researcher's works which are duly listed in the reference section.

The matter presented in this thesis has not been submitted for the award of any other degree of this or any other university.

kamlesh

(Ms. Kamlesh Dhayal)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

(Mrs. Shalini Batra)
Lecturer (SS),
Computer Science and Engineering Department
Thapar University,Patiala.

Countersigned by:

16/06/2009

(Dr. SEEMA BAWA)
Professor & Head,
Computer Science & Engineering Department,
Thapar University,
Patiala.

25/6/09

(Dr. R.K.SHARMA)
Dean (Academic Affairs),
Thapar University,
Patiala.

i

Acknowledgement

The real spirit of achieving a goal is through the way of excellence and austerous discipline. I would have never succeeded in completing my task without the cooperation, encouragement and help provided to me by various personalities.

First of all, I render my gratitude to the ALMIGHTY who bestowed self-confidence, ability and strength in me to complete this work. Without his grace this would never come to be today's reality.

With deep sense of gratitude I express my sincere thanks to my esteemed and worthy Supervisor **Mrs. Shalini Batra (Sr. Lecturer),** Department of Computer Science and Engineering for her valuable guidance in carrying out this work under her effective supervision, encouragement, enlightenment and cooperation. Most of the novel ideas and solutions found in this thesis are the result of our numerous stimulating discussions. Her feedback and editorial comments were also invaluable for writing of this thesis.

I shall be failing in my duties if I do not express my deep sense of gratitude towards **Dr. Seema Bawa,** Professor and Head of Computer Science and Engineering Department who has been a constant source of inspiration for me throughout this work.

I am also thankful to all the staff members of the Department for their full cooperation and help.

My greatest thanks are to all who wished me success especially my parents, my sisters whose support and care makes me stay on earth.

Place: TU, Patiala

Date:

Kamlesh

(Ms. Kamlesh Dhayal)

Abstract

The quality of the data is one of the most important factors influencing the performance of any classification or clustering algorithm. The attributes defining the feature space of a given data set can often be inadequate, which make it difficult to discover interesting knowledge or desired output. However, even when the original attributes are individually inadequate, it is often possible to combine such attributes in order to construct new ones with greater predictive power. Feature selection, as a preprocessing step to machine learning, has been very effective in reducing dimensionality, removing irrelevant data, and noise from data to improving result comprehensibility. This thesis addresses the task of feature selection for clustering and classification.

The goal of this thesis is to find out the best feature subset from the given features in order to improve the performance of classification and clustering techniques on complex, real world data. To partition a given document collection into clusters of similar documents a choice of good features along with good clustering algorithms is very important in clustering. The feature selection is an important part in automatic text categorization which can change the entire results of text clusters.

This thesis addresses the problem of feature selection for machine learning through various methods. The central hypothesis is that good feature sets contain features that are highly correlated with the class, yet uncorrelated with each other. A feature evaluation formula, based on ideas from test theory, provides an operational definition of this hypothesis. This thesis give a comparative study of variety of feature selection methods for data mining, including Information Gain (IG) and $\chi 2$ statistic (CHI) etc using Weka, an open source data mining tool.

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

Data mining is a term coined to describe the process of sifting through large databases for interesting patterns and relationships. With the declining cost of disk storage, the size of many corporate and industrial databases have grown to the point where analysis by anything but parallelized machine learning algorithms running on special parallel hardware is infeasible. Two approaches that enable standard machine learning algorithms to be applied to large databases are feature selection and sampling. Both reduce the size of the database. Recently lot of research work is going on feature selection, a process that can benefit learning algorithms regardless of the amount of data available to learn from.

In 1991 it was alleged that the amount of stored information doubles every twenty months. Unfortunately, as the amount of machine readable information increases, the ability to understand and make use of it does not keep pace with its growth. Machine learning provides tools by which large quantities of data can be automatically analyzed. Fundamental to machine learning is feature selection. Feature selection, by identifying the most salient features for learning, focuses a learning algorithm on those aspects of the data most useful for analysis and future prediction. Feature selection for clustering or classification tasks can be accomplished on the basis of correlation between features, and that such a feature selection process can be beneficial to a variety of common machine learning algorithms. A technique for correlation-based feature selection, based on ideas from test theory, is developed and evaluated using common machine learning algorithms on a variety of natural and artificial problems. The feature selector is simple and fast to execute. It eliminates irrelevant and redundant data and, in some cases, improves the performance of learning algorithms.

1.1 Introduction of Machine Learning

Machine learning is the study of algorithms that automatically improve their performance

with experience. At the heart of performance is prediction. Machine learning algorithms can be broadly characterized by the language used to represent learned knowledge. No single learning approach is clearly superior in all cases, and in fact, different learning algorithms often produce similar results [21]. One factor that can have an enormous impact on the success of a learning algorithm is the nature of the data used to characterize the task to be learned. If the data fails to exhibit the statistical regularity that machine learning algorithms exploit, then learning will fail. It is possible that new data may be constructed from the old in such a way as to exhibit statistical regularity and facilitate learning, but the complexity of this task is such that a fully automatic method is intractable.

If, however, the data is suitable for machine learning, then the task of discovering regularities can be made easier and less time consuming by removing features of the data that are irrelevant or redundant with respect to the task to be learned. This process is called feature selection. Unlike the process of constructing new input data, feature selection is well defined and has the potential to be a fully automatic, computationally tractable process. The benefits of feature selection for learning can include a reduction in the amount of data needed to achieve learning, improved predictive accuracy, learned knowledge that is more compact and easily understood, and reduced execution time. The last two factors are of particular importance in the area of commercial and industrial data mining [7].

Existing feature selection methods for machine learning typically fall into two broad categories those which evaluate the worth of features using the learning algorithm that is to ultimately be applied to the data, and those which evaluate the worth of features by using heuristics based on general characteristics of the data. The former are referred to as wrappers and the latter filters [24]. Within both categories, algorithms can be further differentiated by the exact nature of their evaluation function, and by how the space of feature subsets is explored.

Wrappers often give better results (in terms of the final predictive accuracy of a learning algorithm) than filters because feature selection is optimized for the particular learning algorithm used. However, since a learning algorithm is employed to evaluate each and

every set of features considered, wrappers are prohibitively expensive to run, and can be intractable for large databases containing many features. Furthermore, since the feature selection process is tightly coupled with a learning algorithm, wrappers are less general than filters and must be re-run when switching from one learning algorithm to another.

The advantages of filter approaches to feature selection outweigh their disadvantages. In general, filters execute many times faster than wrappers, and therefore stand a much better chance of scaling to databases with a large number of features than wrappers do. Filters do not require re-execution for different learning algorithms. Filters can provide the same benefits for learning as wrappers do. If improved accuracy for a particular learning algorithm is required, a filter can provide an intelligent starting feature subset for a wrapper, a process that is likely to result in a shorter, and hence faster, search for the wrapper. In a related scenario, a wrapper might be applied to search the filtered feature space that is, the reduced feature space provided by a filter. Both methods help scale the wrapper to larger datasets. For these reasons, a filter approach to feature selection for machine learning is explored.

Filter algorithms described in the machine learning literature have exhibited a number of drawbacks. Some algorithms do not handle noise in data, and others require that the level of noise be roughly specified by the user a-priori. In some cases, a subset of features is not selected explicitly; instead, features are ranked with the final choice left to the user. In other cases, the user must specify how many features are required, or must manually set a threshold by which feature selection terminates. Some algorithms require data to be transformed in a way that actually increases the initial number of features. This last case can result in a dramatic increase in the size of the search space.

1.2 Machine Learning: Concepts and Definitions

Machine learning can be defined in many different ways. Herbert Simon defines learning in general (human, machine, or otherwise) as "any change in a system that allows it to perform better the second time on repetition of the same task or on another task" [8]. While brief, this definition addresses two key issues involved in machine learning.

First, Simon's definition describes learning as allowing the system to "perform better the second time." This implies some kind of change to the system as it processes information.

In an environment of imperfect information, determining the right change to make is difficult. Consequently, the performance of the system will sometimes degrade; detecting and dealing with these "mistakes" is an important part of machine learning.

Second, performance should improve not only on the repetition of the same task, but also on similar tasks (or "on another task") in the domain. Memorizing a list of dates that paychecks are issued is not the same as being able to infer from the dates that paychecks are issued every other Friday. From a practical perspective, it would be impossible or unfeasible for a system to memorize all possible facts. The concept of even numbers cannot be "learned" by simply memorizing a list of numbers (infinitely large!). In addition, it is rare in real-world situations to have perfect information. This process of generalizing from limited experience is called induction.

Definition of Learning:

- From Mitchell (1997)

  "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."
- From Witten and Frank (2000)

  "Things learn when they change their behavior in a way that makes them perform better in the future."
- From Tom Dietterich

  "The goal of machine learning is to build computer systems that can adapt and learn from their experience."

1.2.1 Data Representation

In a typical machine learning task, data is represented as a table of examples or instances. Each instance is described by a fixed number of measurements, or features, along with a label that denotes its class. Features (sometimes called attributes) are typically one of two types: nominal (values are members of an unordered set), or numeric (values are real numbers). Table 1.1 shows fourteen instances of suitable and unsuitable days for which to play a game of golf. Each instance is a day described in terms of the attributes Outlook, Humidity, Temperature and Wind, along with the class label which indicates

whether the day is suitable for playing golf or not. A typical application of a machine learning algorithms requires two sets of examples: training examples and test examples. The set of training examples are used to produce the learned concept descriptions and a separate set of test examples are needed to evaluate the accuracy. When testing, the class labels are not presented to the algorithm. The algorithm takes, as input, a test example and produces, as output, a class label (the predicted class for that example).

Instance	Features				Class
	Outlook	**Temperature**	**Humidity**	**Wind**	
1	Sunny	85	85	False	Don't play
2	Sunny	80	90	True	Don't play
3	Overcast	83	78	False	Play
4	Rain	70	96	False	Play
5	Rain	68	80	False	Play
6	Rain	65	70	True	Don't play
7	Overcast	64	65	True	Play
8	Sunny	72	95	False	Don't play
9	Sunny	69	70	False	Play
10	Rain	75	80	False	Play
11	Sunny	75	70	True	Play
12	Overcast	72	90	True	Play
13	Overcast	81	75	False	Play
14	Rain	71	80	True	Don't play

Table 1.1: The "Golf" dataset

1.2.2 Advantages of Machine Learning

There are several reasons why machine learning is important. Some of these are:

- Some tasks cannot be defined well except by example, that is it might be easy to specify input output pairs but not a concise relationship between inputs and desired outputs. Machines should be able to adjust their internal structure to produce correct outputs for a large number of sample inputs and thus suitably constrain their input/output function to approximate the relationship implicit in the examples.

- Relationships and correlations can be hidden within large amounts of data. Machine Learning/Data Mining may be able to find these relationships.
- Human designers often produce machines that do not work as well as desired in the environments in which they are used. In fact, certain characteristics of the working environment might not be completely known at design time. Machine learning methods can be used for on-the-job improvement of existing machine designs.
- The amount of knowledge available about certain tasks might be too large for explicit encoding by humans (e.g., medical diagnostic).
- Environments change over time. Machines that can adapt to a changing environment would reduce the need for constant redesign.
- New knowledge about tasks is constantly being discovered by humans. It may be difficult to continuously re-design systems "by hand".

1.2.3 Types of Learning

Machine learning is not only about classification. The following main classes of problems exist:

- **Classification learning**: A method to Learn to put instances into predefined classes. Classification is a data mining (machine learning) technique used to predict group membership for data instances. For example, to use classification to predict whether the weather on a particular day will be "sunny", "rainy" or "cloudy". Popular classification techniques include decision trees, neural networks etc.
- **Association learning**: A method to learn relationships between the attributes. A type of learning principle based on the assumption that ideas and experiences reinforce one another and can be linked to enhance the learning process.
- **Clustering:** A method to discover classes of instances that are grouped together.

1.2.4 Machine Learning Applications

Application of machine learning methods to large databases is called data mining. The analogy is that a large volume of earth and raw material is extracted from a mine, which when processed leads to a small amount of very precious material; similarly in data

mining, a large volume of data is processed to construct a simple model with valuable use, for example, having high predictive accuracy. Its application areas are abundant, in addition to retail, in finance banks analyze their past data to build models to use in credit applications, fraud detection, and the stock market. In manufacturing, learning models are used for optimization, control, and troubleshooting. In medicine, learning programs are used for medical diagnosis. In telecommunications, call patterns are analyzed for network optimization and maximizing the quality of service. In science, large amounts of data in physics, astronomy, and biology can only be analyzed fast enough by computers. The World Wide Web is huge; it is constantly growing and searching for relevant information cannot be done manually.

But machine learning is not just a database problem; it is also a part of artificial intelligence. To be intelligent, a system that is in a changing environment should have the ability to learn. Machine learning also helps in find the solutions to many problems in vision, speech recognition, and robotics. Machine learning uses the theory of statistics in building mathematical models, because the core task is making inference from a sample. The role of computer science is twofold. First, in training, it needs efficient algorithms to solve the optimization problem, as well as to store and process the massive amount of data it generally has. Second, once a model is learned, its representation and algorithmic solution for inference needs to be efficient as well.

1.3 An Overview of Classification

In machine learning, the classification task is commonly referred to as supervised learning. In supervised learning there is a specified set of classes, and example objects are labeled with the appropriate class. The goal is to generalize (form class descriptions) from the training objects that will enable novel objects to be identified as belonging to one of the classes. The success of classification learning is heavily dependent on the quality of the data provided for training; a learner has only the input to learn from. If the data is inadequate or irrelevant then the concept descriptions will reflect this and misclassification will result when they are applied to new data.

Classification is composed of two steps:

- Training: Supervised learning of a training set of data to create a model.

- Testing: Classifying the data according to that model.

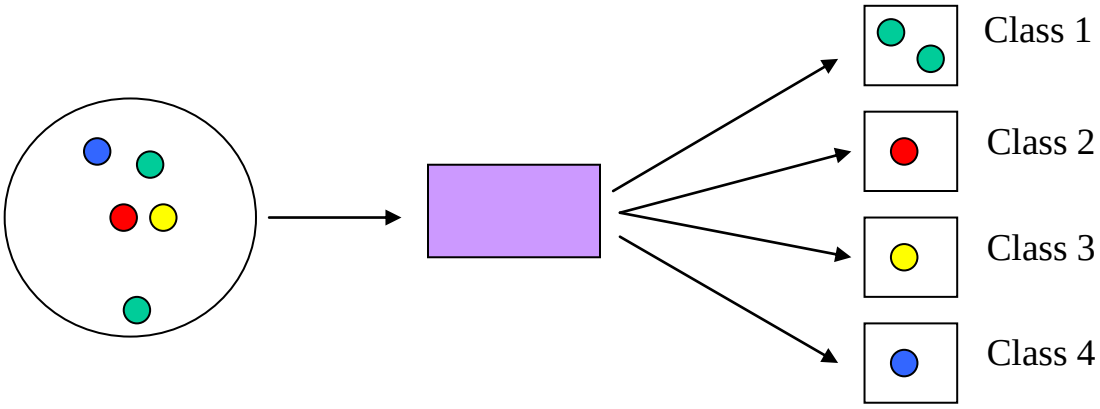


Figure 1.1: Classifier or a classification process.

Some well-known classification algorithms include Bayesian Classification (based on Bayes Theorem), decision trees, neural networks and back propagation (based on neural networks), k-nearest neighbor classifiers (based on learning by analogy), and genetic algorithms.

Decision Trees: An intuitive class of classification algorithms is decision trees. These algorithms solve the classification problem by repeatedly partitioning the input space, so as to build a tree whose nodes are as pure as possible (that is, they contain points of a single class). Classification of a new test point is achieved by moving from top to bottom along the branches of the tree, starting from the root node, until a terminal node is reached. Decision trees are simple yet effective classification schemes for small datasets. The computational complexity scales unfavorably with the number of dimensions of the data. Large datasets tend to result in complicated trees, which in turn require a large memory for storage. Decision trees are a popular top-down approach to classification that divides the data into leaf and node divisions until the entire set has been analyzed. A decision tree of golf dataset (table 1.1) is shown in figure 1.2.

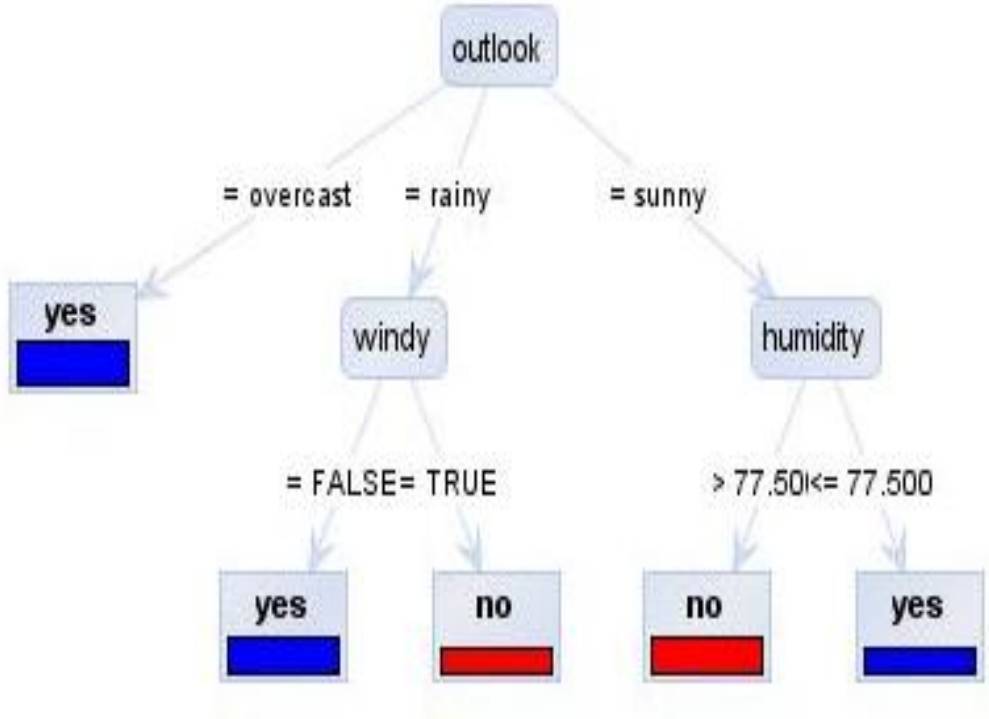


Figure 1.2: A decision tree for the "Golf" dataset. Branches correspond to the values of attributes; leaves indicate classifications.

1.4 An Overview of Clustering

Clustering is a process of partitioning data into similar groupings. A cluster is a collection of 'similar' items. In clustering, data is partitioned into classes. Class members of each class are in some way more "similar" to each other than with those pertaining to different classes. So

- Intra-class variance is low.
- Inter-class variance is high

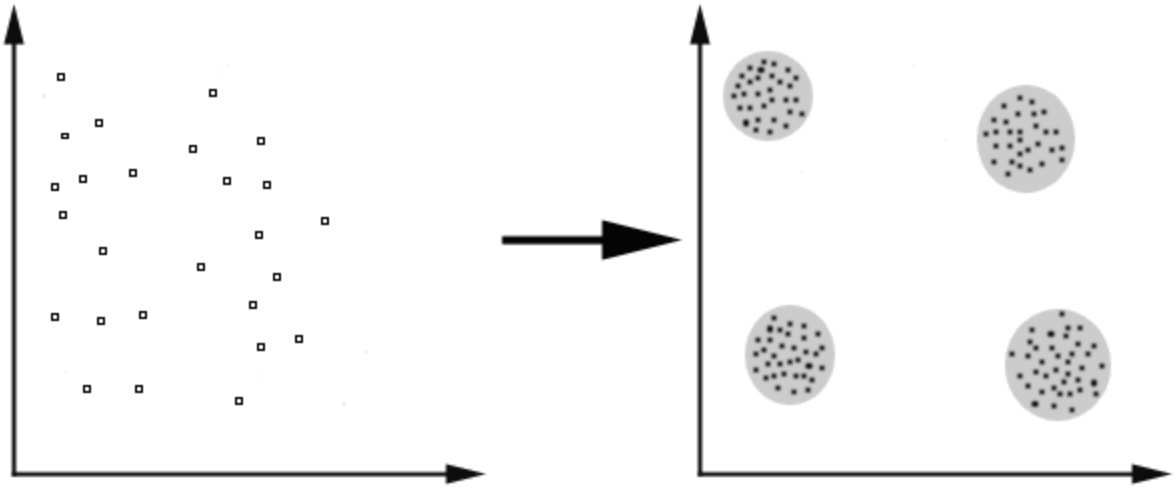


Figure 1.3: A clustering process.

Figure 1.3 shows how K clusters are formed such that objects of one cluster are similar to each other whereas objects of different clusters are dissimilar.

1.4.1 Classification of Clustering Techniques

To use most clustering algorithms two things are necessary:

• An object representation,

• A similarity (or distance) measure between objects.

Many different clustering algorithms have been proposed and tried for document clustering. In this section a few basic algorithms are presented. There are basically two types of clustering algorithms: hierarchal and partitioning. Hierarchical algorithms produce a hierarchy of clusters, while partitioning algorithms gives a flat partition of the set. In clustering each object belongs to only one cluster.

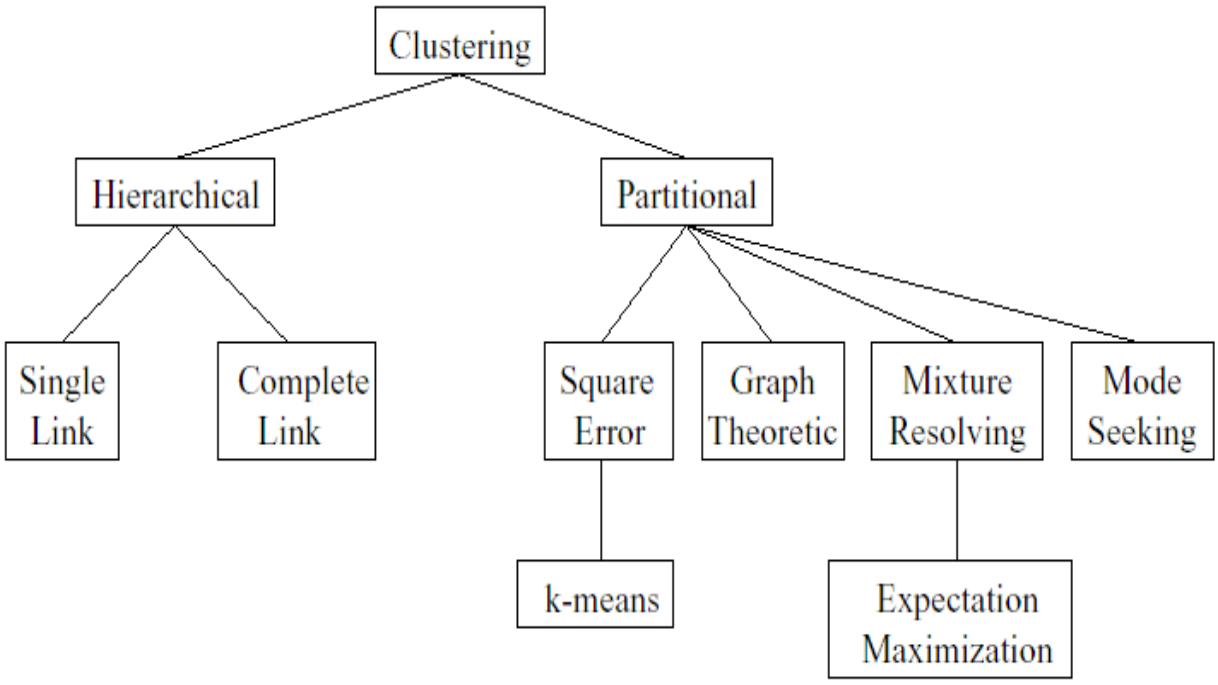


Figure 1.4: Taxonomy of Clustering Approaches [6].

1.4.1.1 Partitioning Algorithms

Partitioning algorithms divide a set of objects into a given number of smaller sets. The most common partitioning algorithms follow a 2-step process:

Step1: Choose a set of "representative" objects.

Step2: Assign each remaining object to its nearest representative.

The critical part of these algorithms lies in the first step.

One of the most commonly used partitioning clustering algorithm is K-Means, which is described in most texts on clustering. Following are the steps of k-mean partitioning algorithm may be elaborated with different outcomes.

K-Means Algorithm:-

1. Pick *k* objects at random and let them define *k* clusters.
2. Calculate cluster representatives.
3. Make new clusters, one per cluster representative. Let each text belong to the cluster with the most similar cluster representative.
4. Repeat from 2 until a stopping criterion is reached.

The first step defines a random initial partition. There are many other ways of constructing it and the result depends on which one is used. As cluster representative the

mean (the centroid) of the objects in the cluster is generally used. Other variants are to let the median or a few specific objects represent the cluster. The stopping criterion is normally when no objects change clusters, or when very few change clusters between iterations. It may also be to stop after a predefined number of iterations, since most quality improvement usually is gained during the first iterations. The stopping criterion may also be defined using some internal quality measure. In each iteration the cluster representatives and the similarities between all objects and all clusters must be computed. The K-Means algorithm requires a number of clusters as input. That is, one has to guess the appropriate number. Of course it is possible to run the algorithm with several different numbers of clusters and report only the clustering with the best result.

1.4.1.2 Hierarchical Algorithms

Hierarchical algorithms build a cluster hierarchy; small clusters are merged with each other to form new clusters that are composed of clusters and so on. There are two natural ways of constructing such a hierarchy: bottom-up and top-down.

Agglomerative Clustering

The agglomerative algorithms generate the same cluster hierarchy every time. In agglomerative clustering, it needs to compute the similarity between all objects to find the pair of objects that are most similar.

Agglomerative Clustering Algorithm:

1. Construct one cluster for each document.
2. Join the *t* most similar clusters.
3. Repeat 2 until a stopping criterion is reached.

Divisive Clustering

In the divisive algorithms any partitioning algorithm can be applied to split clusters (step 2). The Bisecting K-Means algorithm is a divisive algorithm for document clustering that uses the K-Means algorithm to split the worst cluster in two. The worst cluster is defined as the largest, which gives equally good results as choosing the cluster with lowest intra similarity.

Divisive Clustering Algorithm:

1. Put all documents into one cluster.
2. Split one cluster (the worst) in *t* new.

3. Repeat 2 until a stopping criterion is reached.

1.4.2 Clustering versus Classification

By automatic categorization we mean to let a machine decide to which of a set of predefined categories a text belongs. In clustering the machine decides how a given text set should be partitioned. Categorization is suitable when one wants to categorize new texts according to a known category, whereas clustering is suitable when one wants to discover new structures not previously known. Both methods may give interesting results on an unknown text set; categorization sorts them according to a well known structure, clustering displays the structure of the particular set. The objects clustered are sequences of words which are referred to as texts, documents, articles or papers depending on the context. A set of such objects is referred as a text set, a document collection or a corpus. A set may be grouped in several manners, either by a clustering algorithm, or by humans according to some agreement which is referred as a classification or a categorization that consists of classes or categories.

Supervised learning is when a classification system is given some inputs a long with their answers, which is called the training set. Unsupervised learning is when the algorithm is never given a training set and is basically left on its own to classify its inputs. One of the most known unsupervised methods is by clustering. Some methods for supervised learning can be altered a little bit and used for unsupervised classification.

This thesis is divided into six chapters. In this chapter, brief introduction of machine learning was described. It also gave some overview of clustering and classification.
Chapter 2 describes the literature survey.
Chapter 3 presents the problem statement.
Chapter 4 gives a detailed introduction about feature selection. It include a feature selection system architecture, characteristics of feature selection algorithms and categorization of feature selection methods. It also includes a brief introduction of Weka and RapidMiner (data mining tools).
Chapter 5 explains the experiment performed during the thesis and the results are evaluated. Accuracy of classification methods with and without feature selection is compared.

Chapter 6 presents the conclusion of this thesis and suggests future work. At the end of this thesis, references and paper published are presented.

Chapter 2

Literature Review

In the late ’80s the most popular approach to text categorization, at least in the “operational” (i.e., real-world applications) community, was a knowledge engineering (KE), but in ’90s this approach has increasingly lost popularity (especially in the research community) in favor of the machine learning (ML) paradigm, according to which a general inductive process automatically builds an automatic text classifier by learning, from a set of pre-classified documents, supervised and unsupervised learning have been the focus of critical research in the areas of machine learning and artificial intelligence. In the literature, these two streams flow independently of each other, despite their close conceptual and practical connections. The automated categorization (or classification) of data into predefined categories has seen a significant rising interest in the last ten years, due to the increased availability of documents in digital form and the ensuing need to organize them. Fabrizio Sebatiani [6], in his survey discusses the main approaches to text categorization that fall within the machine learning paradigm.

John, Kohavi and Pfleger [7] addressed the problem of irrelevant features and the subset selection problem. They presented definitions for irrelevance and for two degrees of relevance (weak and strong). They also state that features selected should depend not only on the features and the target concept, but also on the induction algorithm. Further, they claim that the filter model approach to subset selection should be replaced with the wrapper model. Pudil, and Kittler [20] presented ‘floating’ search methods in feature selection. These are sequential search methods characterized by a dynamically changing number of features included or eliminated at each step. It gives very good results and are computationally more effective than the branch and bound method.

Blum and Langley [1] focuses on two key issues: the problem of selecting relevant features and the problem of selecting relevant examples. Kohavi and John [24] introduced wrappers for feature subset selection. Their approach searches for an optimal feature subset tailored to a particular learning algorithm and a particular training set. In a

comparative study of feature selection methods in statistical learning of text categorization (with a focus on aggressive dimensionality reduction), Yang and Pedersen [27] evaluated document frequency (DF), information gain (IG), mutual information (MI), a 2-test (CHI) and term strength (TS); and found IG and CHI to be the most effective. Dash and Liu [4] gave a survey of feature selection methods for classification. They present a definition of feature selection after discussing many existing definitions. Four steps of a typical feature selection process are recognized: generation procedure, evaluation function, stopping criterion, and validation procedure. They also group the generation procedures into thee categories: complete, heuristic, and random, and the evaluation functions into five categories: distance, information, dependence, consistency, and classifier error rate measures. Thirty two existing feature selection methods are categorized based on the combinations of generation procedure and evaluation function used in them.

Liu and Motoda [12] wrote their book on feature selection which offers an overview of the methods developed since the 1970s and provides a general framework in order to examine these methods and categorize them. Jordan and Karp (2001) successfully applied feature selection methods (using a hybrid of filter and wrapper approaches) to a classification problem in molecular biology involving only 72 data points in a 7130 dimensional space. They also investigated regularization methods as an alternative to feature selection, and showed that feature selection methods were preferable in the problem they tackled.

From past few years, much research is going on in the area of data filtering with large focus on feature selection. As discussed above many methods of feature selection are proposed by various researchers and we are here giving a comparative study of various feature selection methods using data mining tools Weka and Rapid Miner.

Chapter 3

Problem Statement

3.1 Problem definition

It has been widely observed that the huge amount of data available for text categorization and clustering often give unexpected and irrelevant results due to noisy, irrelevant or misleading features found in stored information. To overcome this problem some of the important and most relevant features should be selected which serve as the basis for further analysis of the given data. Study of the filtering mechanism for removing trivial and unimportant features was the basic foundation of our research. After the literature review of the topic it was analyzed that there are some basic problems in feature selection which include:

- How do we find a set of features that is a good predictor of what class a sample belongs to?
- What are the strengths and weaknesses of existing feature selection methods applied to text categorization?
- To what extend can feature selection improve the accuracy of a classifier? How much of the document vocabulary can be reduced without losing useful information in category prediction?
- Problem of selecting some subset of a learning algorithm's input variables upon which it should focus attention, while ignoring the rest.

To deal with all the aforesaid issues we tried to apply various feature selection methods including chi square, information gain, etc. on data set with the help of open source data mining tools like Weka and Rapid miner.

Chapter 4

Feature Selection for Machine Learning

4.1 Introduction of Feature Selection

A "feature" or "attribute" or "variable" refers to an aspect of the data. Usually before collecting data, features are specified or chosen. Features can be discrete, continuous, or nominal.

Generally, features are characterized as:

1. Relevant: These are features which have an influence on the output and their role can not be assumed by the rest
2. Irrelevant: Irrelevant features are defined as those features not having any influence on the output, and whose values are generated at random for each example.
3. Redundant: A redundancy exists whenever a feature can take the role of another (perhaps the simplest way to model redundancy).

4.1.1 Feature selection Definition

Problem of selecting some subset of a learning algorithm's input variables upon which it should focus attention, while ignoring the rest. Feature selection is the process of selecting the best feature among all the features because all the features are not useful in constructing the clusters: some features may be redundant or irrelevant thus not contributing to the learning process.

The main aim of feature selection is to determine a minimal feature subset from a problem domain while retaining a suitably high accuracy in representing the original features. In many real world problems Feature selection is a must due to the abundance of noisy, irrelevant or misleading features. For instance, by removing these factors, learning from data techniques can benefit.

To be completely sure of the attribute selection, we would ideally have to test all the enumerations of attribute subsets, which is infeasible in most cases as it will result in 2n subsets of n attributes.

4.1.2 Advantages of feature selection

- It reduces the dimensionality of the feature space, to limit storage requirements and increase algorithm speed;
- It removes the redundant, irrelevant or noisy data.
- The immediate effects for data analysis tasks are speeding up the running time of the learning algorithms.
- Improving the data quality.
- Increasing the accuracy of the resulting model.
- feature set reduction, to save resources in the next round of data collection or during utilization;
- performance improvement, to gain in predictive accuracy;
- data understanding, to gain knowledge about the process that generated the data or simply visualize the data

4.2 Feature Selection System Architecture

A simple three step feature selection approach is explained below. The goal of this architecture is to reduce a large set of features (on the order of thousands) to a small subset of features (on the order of tens), without significantly reducing the system's ability. The basic three steps of this system are:

- In first step the irrelevant features are removed.
- After that the redundant features are removed.
- And finally a feature selection algorithm is applied to the remaining features.

In this approach each step is working as a filter that reduces the number of candidate features, until finally only a small subset remains.

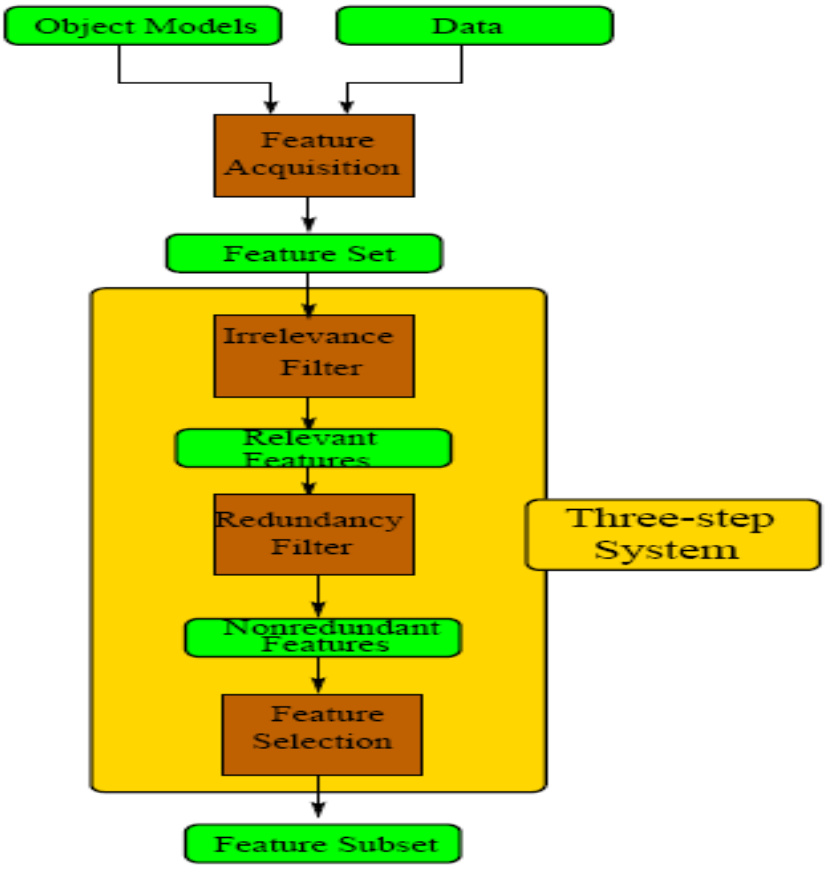


Figure 4.1 .A feature selection system architecture [2].

The first filter removes irrelevant features using a modified form of the Relief algorithm, which assigns relevance values to features by treating training samples as points in feature space. For each sample, it finds the nearest "hit" (another sample of the same class) and "miss" (a sample of a different class), and adjusts the relevance value of each feature according to the square of the feature difference between the sample and the hit and miss. There are several modifications to Relief to generalize it for continuous features and to make it more robust in the presence of noise. This system adopt Kononenko's modifications, and modify Relief again to remove a bias against non-monotonic features, as described in [13]. Within this feature selection system, Relief is used as a relevance filter. Therefore it threshold the relevance values, to divide the feature set into relevant and irrelevant features. This can be done either by thresholding the relevance value directly, or by selecting the highest n values and discarding the remaining features. In either case, relief does not detect redundancy, so the remaining feature set still contains redundant features. The second step is a redundancy filter that uses the K-

means algorithm [14] to cluster features according to how well they correlate to each other. When feature clusters are discovered, only the feature with the highest Relief score is kept; the other features in the cluster are removed from the feature set. This is an unusual application of K-means clustering, in that features are clustered (instead of samples), and correlation is used as the distance measure. A correlation threshold of 0.97 is used to detect when the features in a cluster are not sufficiently similar, in which case the cluster is split to make sure that potentially useful features are not removed from the feature set. The third and final filter is a combinatorial feature selection algorithm.

4.3 Characteristics of Feature Selection Algorithms

Feature selection algorithms (with a few notable exceptions) perform a search through the space of feature subsets, and, as a consequence, must address four basic issues affecting the nature of the search [21]:

1. Starting point. Selecting a point in the feature subset space from which to begin the search can affect the direction of the search. One option is to begin with no features and successively add attributes. In this case, the search is said to proceed forward through the search space. Conversely, the search can begin with all features and successively remove them. In this case, the search proceeds backward through the search space. Another alternative is to begin somewhere in the middle and move outwards from this point.
2. Search organization. An exhaustive search of the feature subspace is prohibitive for all but a small initial number of features. With N initial features there exist 2N possible subsets. Heuristic search strategies are more feasible than exhaustive ones and can give good results, although they do not guarantee finding the optimal subset.
3. Evaluation strategy. How feature subsets are evaluated is the single biggest differentiating factor among feature selection algorithms for machine learning. One paradigm, dubbed the filter [23, 24] operates independent of any learning algorithm, undesirable features are filtered out of the data before learning begins. These algorithms use heuristics based on general characteristics of the data to evaluate the merit of feature subsets. Another school of thought argues that the

bias of a particular induction algorithm should be taken into account when selecting features. This method, called the wrapper, uses an induction algorithm along with a statistical re-sampling technique such as cross-validation to estimate the final accuracy of feature subsets. Figure No. 4.2 and 4.3 illustrates the filter and wrapper approaches to feature selection.

4. Stopping criterion. A feature selector must decide when to stop searching through the space of feature subsets. Depending on the evaluation strategy, a feature selector might stop adding or removing features when none of the alternatives improves upon the merit of a current feature subset. Alternatively, the algorithm might continue to revise the feature subset as long as the merit does not degrade. A further option could be to continue generating feature subsets until reaching the opposite end of the search space and then select the best.

4.4 Feature Selection Methods

Feature selection methods can be subdivided in two categories: filter methods and wrapper methods. The main difference is that a wrapper method makes use of the classifier, while a filter method does not. From a conceptual viewpoint, wrapper approaches are clearly advantageous, since the features are selected by optimizing the discriminative power of the finally used classifier.

Filter feature selection methods evaluate attributes prior to the learning process, and without specific reference to the classification algorithm that will be used to generate the final classifier. The filtered dataset may then be used by any classification algorithms. Rank-based feature selection is a commonly-used filter method and is discussed in the following section. Wrapper methods use a controlled enumeration strategy and apply the classification algorithm that will be used to generate the final classifier to test the performance of each attribute subset. The ways in which the attribute sets are incrementally updated results in two types of wrapper models: forward selection and backward elimination.

4.4.1 Filter Methods

The earliest approaches to feature selection within machine learning were filter methods. All filter methods use heuristics based on general characteristics of the data rather than a

learning algorithm to evaluate the merit of feature subsets. As a consequence, filter methods are generally much faster than wrapper methods, and, as such, are more practical for use on data of high dimensionality.

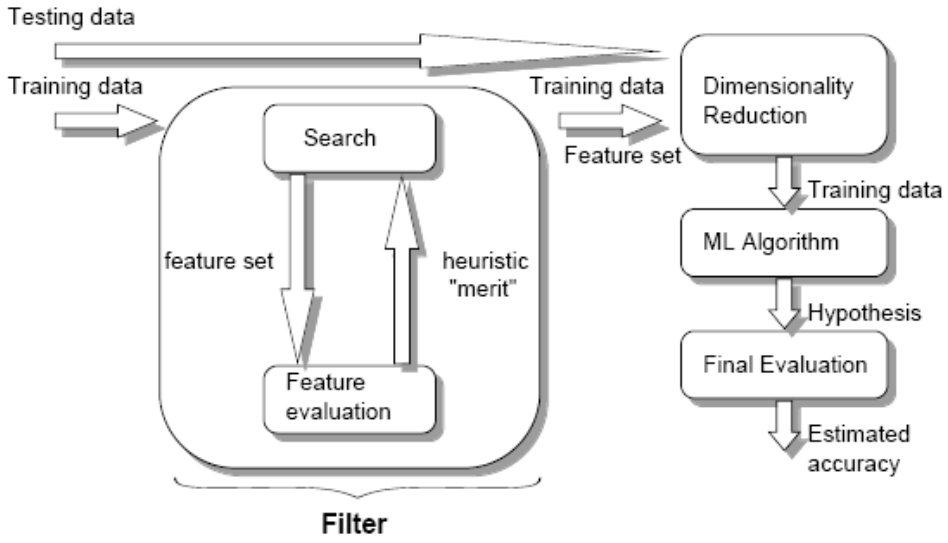


Figure 4.2: Filter Method [17].

The filter selection model is the earliest approach to feature selection. It utilizes an independent search criterion to find the appropriate feature subset before a machine learning algorithm is performed, thus it was termed as filter method[6]: it filters out irrelevant attributes before induction occurs, that is, the search is done independently of an induction algorithm. The procedure of the filter model is shown in Figure 4.2. The advantage of the filter model is that it does not need to re-run the algorithm for every induction algorithm when choosing to run on a reduced feature dataset, as a consequence, the filter approach is generally computational efficient, and it is practical for data sets with very high dimensionality. There are a number of different representative filter algorithms in the literature.

The RELIEF algorithm follows the general and simple filter scheme, that is, it first evaluates the individual feature according to the evaluation criterion, and thereafter, the best n features are selected. However it uses a more complex evaluation function. The training samples, characterized by the selected features, are then passed to ID3. Two

extensions were made to this algorithm [12], where more general data types can be treated. Although RELIEF use the decision tree induction algorithm after feature selection, it is naturally not confined to decision tree algorithms, i.e. other induction algorithms can be used instead. Since the filter approach does not take into account the learning bias introduced by the final induction algorithm, it may not be able to select the most suitable subset for the final induction algorithm. For this reason, the wrapper model was proposed.

Relief algorithm

The key idea of Relief is to estimate the quality of features according to how well their values distinguish between instances that are near to each other. For this purpose, given a randomly selected instance X from a data set S with k features, Relief searches the data set for its two nearest neighbors: one from the same class, called nearest hit H, and the other from a different class, called nearest miss M. The process is repeated m times, where m is a user-defined parameter. Efficiency is one of the major advantages of the Relief family over other algorithms.

Given m=desired number of sampled instances, and k=number of features,

1. set all weights W[Ai] :=0.0;
2. for j := 1 to m do begin
3. randomly select an instance X;
4. find nearest hit H and nearest miss M;
5. for i := 1 to k do begin
6. W[Ai] := W[Ai] −diff (Ai,X,H)/m+ diff(Ai,X,M)/m;
7. end;
8. end;

Relief algorithm

4.4.1.1 Problems with filter method

- Redundancy in selected features: features are considered independently and not measured on the basis of whether they contribute new information.
- Interactions among features generally can not be explicitly incorporated (some filter methods are smarter than others).

- Classifier has no say in what features should be used: some scores may be more appropriates in conjuction with some classifiers than others.

4.4.2 Wrapper Methods

Wrapper strategies for feature selection use an induction algorithm to estimate the merit of feature subsets. The rationale for wrapper approaches is that the induction method that will ultimately use the feature subset should provide a better estimate of accuracy than a separate measure that has an entirely different inductive bias [21]. Feature wrappers often achieve better results than filters due to the fact that they are tuned to the specific interaction between an induction algorithm and its training data. However, they tend to be much slower than feature filters because they must repeatedly call the induction algorithm and must be re-run when a different induction algorithm is used. Since the wrapper is a well defined process, most of the variation in its application is due to the method used to estimate the off-sample accuracy of a target induction algorithm, the target induction algorithm itself, and the organization of the search. This section reviews work that has focused on the wrapper approach and methods to reduce its computational expense.

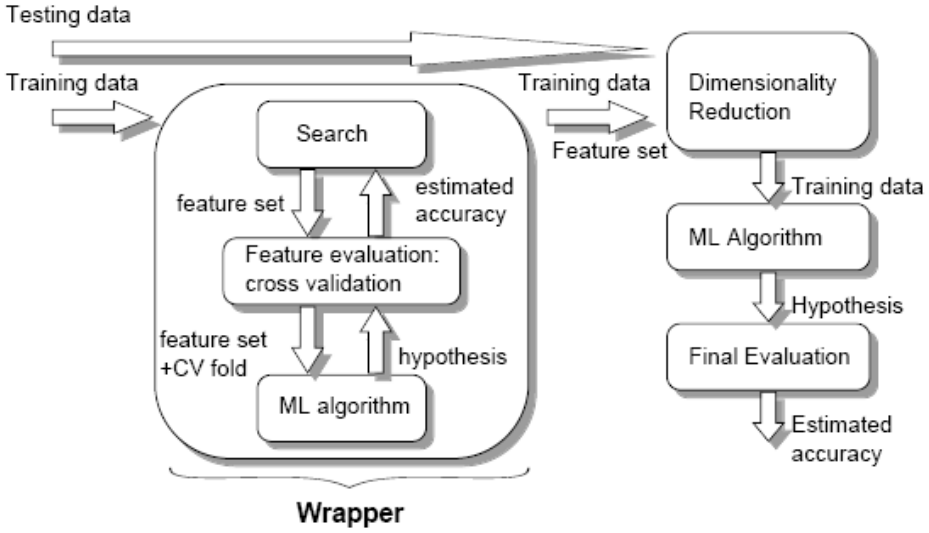


Figure 4.3: Wrapper Method [17].

The strategy of the wrapper model is to use an induction algorithm to estimate the merit of the searched feature subset on the training data and using the estimated accuracy of the resulting classifier as it's metric. The wrapper approaches often have better results than the filter approaches because they are tuned to the specific interaction between an induction algorithm and its training data. A typical wrapper selection approach called genetic feature selectors, which use genetic algorithms as the search engine (and one of them uses the ensemble nearest neighbor classifiers as the induction algorithm). In this way, feature selection takes into account the biases from the final learning algorithm. The wrapper selection procedure is illustrated in Figure 4.3. The disadvantage of the wrapper model is that it is less tractable because of the prohibitive cost of running the classification algorithm many times when the dimensionality is considerably high.

4.5 Supervised and Unsupervised feature Selection Methods

A brief introduction of several effective feature selection methods, including two supervised methods IG, and CHI and four unsupervised methods, DF, TS, En and TC.

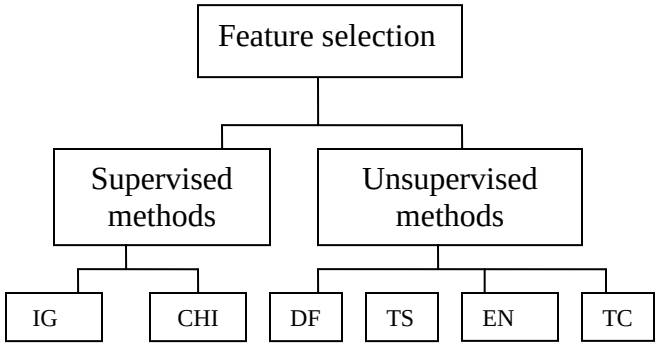


Figure 4.4: Classification of feature selection methods.

IG (Information Gain)

Information gain [27], of a term measures the number of bits of information obtained for category prediction by the presence or absence of the term in a document. Information Gain measures the decrease in entropy when the feature is given vs. absent. In text categorization problems usually have a m-array category space (where m may be up to tens of thousands), and need to measure the goodness of a term globally with respect to

all categories on average. Given a training corpus, for each unique term the information gain is computed and removed from the feature space those terms whose information gain is less than some predetermined threshold.

CHI (χ 2 statistic)

This method measure the lack of independence between a term and the category. Chi-Squared is the common statistical test that measures divergence from the distribution expected if one assumes the feature occurrence is actually independent of the class value. As a statistical test, it is known to behave erratically for very small expected counts, which are common in text classification both because of having rarely occurring word features, and sometimes because of having few positive training examples for a concept.

DF (Document frequency)

Document frequency is the number of documents in which a term occurs in a dataset. It is the simplest criterion for term selection and easily scales to a large dataset with linear computation complexity. It is a simple but effective feature selection method for text categorization [27]. Document Frequency simply measures in how many documents the word appears. Since it can be computed without class labels, it may be computed over the entire test set as well. Selecting frequent words will improve the chances that the features will be present in future test cases. It performed much better than Mutual Information in the study by Yang and Pedersen, but was consistently dominated by IG and Chi (which, they point out, each have a significant correlation with frequent terms).

TS (Term Strength)

Term strength is originally proposed and evaluated for vocabulary reduction in text retrieval [26], and later applied to text categorization.

It is computed based on the conditional probability that a term occurs in the second half of a pair of related documents given that it occurs in the first half. Since we need to calculate the similarity for each document pair, the time complexity of TS is quadratic to the number of documents. Because the class label information is not required, this method is also suitable for term reduction in text clustering.

EN (Entropy-based Ranking)

Entropy-based ranking is proposed by [4]. In this method, the term is measured by the entropy reduction when it is removed. The most serious problem of this method is its high computation complexity O (MN 2) . It is impractical when there is a large number of documents and terms, and therefore, sampling technique is used in real experiments [4].

TC (Term Contribution)

We introduce a new feature selection method called “Term Contribution” that takes the term weight into account. Because the simple method like DF assumes that each term is of same importance in different documents, it is easily biased by those common terms which have high document frequency but uniform distribution over different classes. TC is proposed to deal with this problem. Term strength is originally proposed and evaluated for vocabulary reduction in text retrieval, and later applies to text categorization. This method estimates term importance based on how commonly a term is likely to appear in "closely-related" documents. It uses a training set of documents to derive document pairs whose similarity is above a threshold.

4.6Introduction of Data Mining Tools: Weka and Rapid Miner

Weka is machine learning/data mining software written in Java language (distributed under the GNU Public License). Weka is a collection of machine learning algorithms for data mining tasks. It is used for research, education, and applications. Main features of Weak are:

- Comprehensive set of data pre-processing tools, learning algorithms and evaluation methods.
- Graphical user interfaces (incl. data visualization).
- Environment for comparing learning algorithms.

Weka contains tools for developing new machine learning schemes. It can be used for

- Preprocessing
- Classification
- Clustering

- Association
- Visualization

Input to Weka is given as a data set. Weka permits the input data set to be in numerous file formats like CSV (comma separted values: *.csv), Binary Serialized Instances (*.bsi) etc. However, the most preferred and the most convenient input file format is the attribute relation file format (arff). Typically, here is how an ARFF data set looks like:

The header section of an ARFF file is very simple and merely defines the name of the dataset along with the set of attributes and their associated types.

@relation

The @relation <name> tag is declared at the beginning of the file, where <name> is the name of the relation you wish to use.

@attribute

Attributes are defined in the following format: @attribute <attribute-name> <type>.

An attribute can be one of four types:

1. Numeric- It can be a real or integer value.
2. Nominal-specification where the value must a from a pre-defined set of possible values.
3. String- textual values.
4. Date- [<date-format>] for storing dates. The the optional date format argument instructs Weka on how to parse and print the dates. However, this type will not be very useful for NLP tasks.

An example of ARFF file format:

@relation employees

@attribute empName string

@attribute empSalary numeric

@attribute empGender {male, female}

@atttribute empDob date "yyyy-MM-dd"

@data

’Andrew Roberts’, 50000, male, 1980-11-09

'Phil Space', 20000, male, 1976-04-12

The attributes are case-sensitive, so the label Name is different to name. It's also worth being aware that Weka can act weirdly when there is a value within the data that is the same as an attribute name.

The second half to an ARFF file is the data itself. The @data tag signifies to Weka that the instance data is about to commence. Each line after the tag is a set of values (separated by commas) that represent a single instance. It should be obvious that Weka will expect the order of the values to be in the same order in which the attributes were declared. In Weka string values and nominal values are case sensitive.

Parameters of outputs:-

Accuracy:

It gives a measure for the overall accuracy of the classifier:

$$\text{Accuracy (\%)} = \frac{\text{Number of correctly classified instances}}{\text{Number of instances.}} * 100$$

Precision and recall:

With respect to classifiers:

$$\text{Precision(X)} = \frac{\text{Number of correctly classified instances of class X}}{\text{Number of instances classified as belonging to class X.}}$$

$$\text{Recall(X)} = \frac{\text{Number of correctly classified instances of class X}}{\text{Number of instances in class X.}}$$

Confusion matrix:

Confusion matrices are very useful for evaluating classifiers, as they provide an efficient snapshot of its performance, by displaying the distribution of correct and incorrect instances. Typical Weka output contains the following:

=== Confusion Matrix ===

a b <-- classified as

7 2 | a = yes

3 2 | b = no

Weka classifies instances into two possible classes: yes or no. For the sake of simplicity, Weka substitutes ‘yes’ for a, and ‘no’ for b. The columns represent the instances that were classified as that class. For example classifier that uses a set of attributes to decide whether a patient has cancer or not.

=== Confusion Matrix ===

a b <-- classified as

10 3 | a = cancer

1 6 | b = no cancer

This classifier successfully classifies 16 out of the 20 cases presented. However, an alarming 3 patients will be given the all clear when they did in fact have cancer. The one patient who was told they had cancer despite it being the opposite will also not be happy in the short term, but the outcome is much more favorable. Now, the classifier was updated, and when fed the same data, it resulted in the following matrix:

RapidMiner is an open source learning environment for data mining and machine learning. This environment can be used to extract meaning from a dataset. There are hundreds of machine learning operators to choose from, helpful pre and post processing operators, descriptive graphic visualizations, and many other features. RapidMiner introduces new concepts of transparent data handling and process modeling which eases process configuration for end users. Additionally clear interfaces and a sort of scripting language based on XML turns Rapid- Miner into an integrated developer environment for data mining and machine learning. Today, RapidMiner is the world-wide leading open-source data mining solution and is widely used by researchers and companies.

Features of Rapid Miner:

- Operator trees or subtrees can be saved as building blocks for later re-use
- Internal XML representation ensures standardized interchange format of data mining experiments
- Simple scripting language allowing for automatic large-scale experiments multi-layered data view concept ensures efficient and transparent data handling
- Flexibility in using RapidMiner:
- Graphical user interface (GUI) for interactive prototyping
- Command line mode (batch mode) for automated large-scale applications
- Java API (application programming interface) to ease usage of RapidMiner from your own programs
- Simple plugin and extension mechanisms, a broad variety of plugins already exists and you can easily add your own
- Powerful plotting facility offering a large set of sophisticated high-dimensional visualization techniques for data and models
- Machine learning library WEKA fully integrated (WEKA web page)
- RapidMiner was successfully applied on a wide range of applications where its rapid prototyping abilities demonstrated their usefulness, including text mining, multimedia mining, feature engineering, data stream mining and tracking drifting concepts, development of ensemble methods, and distributed data mining.

Chapter 5

Testing & Results

Implementation

In implementation a comparative study of variety of feature selection methods for data mining, including Information Gain (IG) and $\chi 2$ statistic (CHI) etc using Weka (data mining tool) is done. We also compare the accuracy of different classifier methods with and without using feature selection in Weka [30]. We can find out the weight of each n every attribute in a given file using RapidMiner [31] by applying attribute weight process over a zoo.arff file which includes 18 different attributes. Output of this process shows that which attribute among all the attributes have the highest weightage.

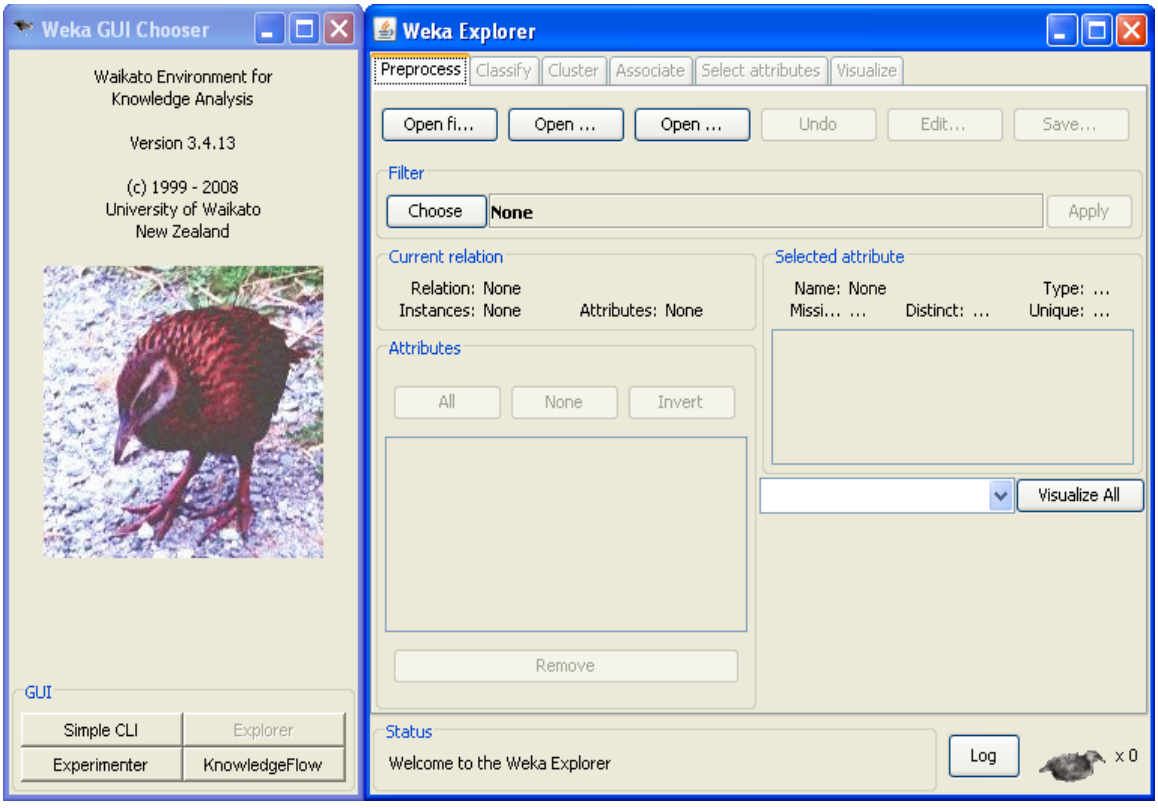


Figure 5.1: Weka Interface.

The input file “Zoo.arff” contains:
Relation: zoo
Instances: 101
Attributes: 18

Animal,
hair,
feathers,
eggs
milk,
airborne,
aquatic,
predator
Toothed,
backbone,
reathes,
venomous
Fins,
legs,
tail,
domestic
Catsize,
type

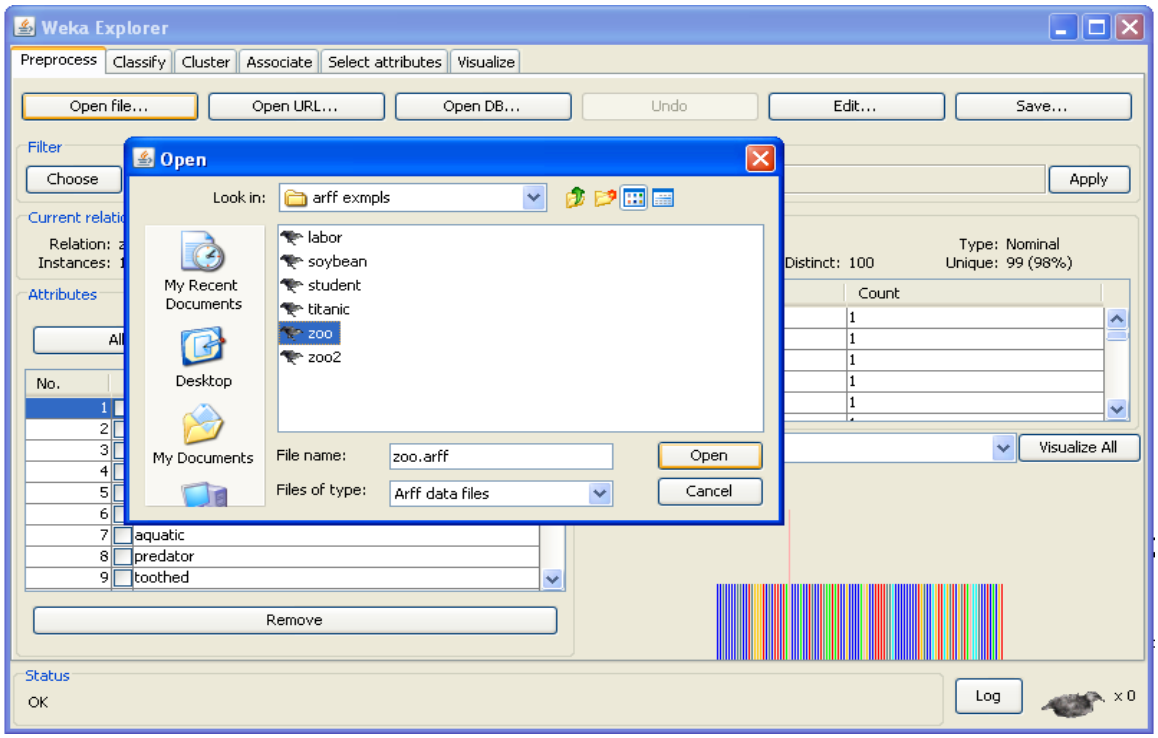
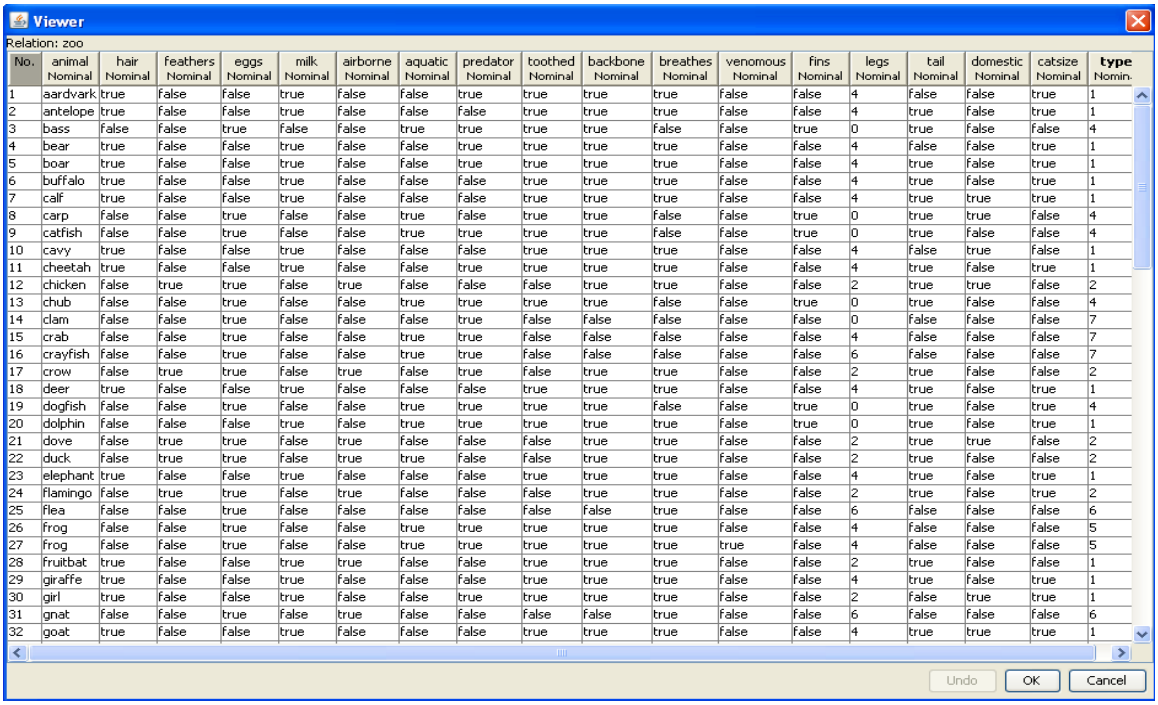


Figure 5.2: Opening a file in Weka.



No.	animal Nominal	hair Nominal	feathers Nominal	eggs Nominal	milk Nominal	airborne Nominal	aquatic Nominal	predator Nominal	toothed Nominal	backbone Nominal	breathes Nominal	venomous Nominal	fins Nominal	legs Nominal	tail Nominal	domestic Nominal	catsize Nominal	type Nomin.
1	aardvark	true	false	false	true	false	false	true	true	true	true	false	false	4	false	false	true	1
2	antelope	true	false	false	true	false	false	false	true	true	true	false	false	4	true	false	true	1
3	bass	false	false	true	false	false	true	true	true	true	false	false	true	0	true	false	false	4
4	bear	true	false	false	true	false	false	true	true	true	true	false	false	4	false	false	true	1
5	boar	true	false	false	true	false	false	true	true	true	true	false	false	4	true	false	true	1
6	buffalo	true	false	false	true	false	false	false	true	true	true	false	false	4	true	false	true	1
7	calf	true	false	false	true	false	false	false	true	true	true	false	false	4	true	true	true	1
8	carp	false	false	true	false	false	true	false	true	true	false	false	true	0	true	true	false	4
9	catfish	false	false	true	false	false	true	true	true	true	false	false	true	0	true	false	false	4
10	cavy	true	false	false	true	false	false	false	true	true	true	false	false	4	false	true	false	1
11	cheetah	true	false	false	true	false	false	true	true	true	true	false	false	4	true	false	true	1
12	chicken	false	true	true	false	true	false	false	false	true	true	false	false	2	true	true	false	2
13	chub	false	false	true	false	false	true	true	true	true	false	false	true	0	true	false	false	4
14	clam	false	false	true	false	false	false	true	false	false	false	false	false	0	false	false	false	7
15	crab	false	false	true	false	false	true	true	false	false	false	false	false	4	false	false	false	7
16	crayfish	false	false	true	false	false	true	true	false	false	false	false	false	6	false	false	false	7
17	crow	false	true	true	false	true	false	true	false	true	true	false	false	2	true	false	false	2
18	deer	true	false	false	true	false	false	false	true	true	true	false	false	4	true	false	true	1
19	dogfish	false	false	true	false	false	true	true	true	true	false	false	true	0	true	false	true	4
20	dolphin	false	false	false	true	false	true	true	true	true	true	false	true	0	true	false	true	1
21	dove	false	true	true	false	true	false	false	false	true	true	false	false	2	true	true	false	2
22	duck	false	true	true	false	true	true	false	false	true	true	false	false	2	true	false	false	2
23	elephant	true	false	false	true	false	false	false	true	true	true	false	false	4	true	false	true	1
24	flamingo	false	true	true	false	true	false	false	false	true	true	false	false	2	true	false	true	2
25	flea	false	false	true	false	false	false	false	false	false	true	false	false	6	false	false	false	6
26	frog	false	false	true	false	false	true	true	true	true	true	false	false	4	false	false	false	5
27	frog	false	false	true	false	false	true	true	true	true	true	true	false	4	false	false	false	5
28	fruitbat	true	false	false	true	true	false	false	true	true	true	false	false	2	true	false	false	1
29	giraffe	true	false	false	true	false	false	false	true	true	true	false	false	4	true	false	true	1
30	girl	true	false	false	true	false	false	true	true	true	true	false	false	2	false	true	true	1
31	gnat	false	false	true	false	true	false	false	false	false	true	false	false	6	false	false	false	6
32	goat	true	false	false	true	false	false	false	true	true	true	false	false	4	true	true	true	1

Figure 5.3: View of "Zoo.arff" file.

The output shows the features ranked by information gain, which is one possible metric for measuring the "goodness" of a partition. The features at the top of the list are the most "informative" as attributes for classification as shown in the figure 5.4. Similar process is repeated for other feature selection methods.

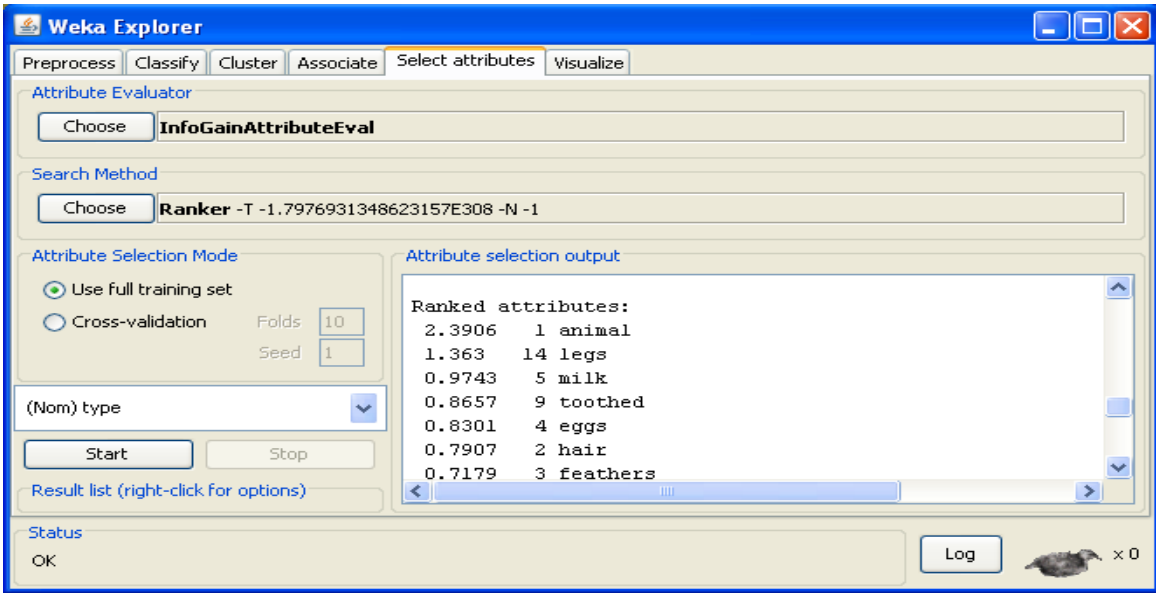


Figure 5.4: Attribute Selection Snapshot

A process of without attribute selection and with attribute selection classification is shown in figure 5.5 and 5.6 respectively. The main window will show a variety of result summary statistics, such as accuracy, true positives, false positives, and a confusion matrix as shown in figure 5.5. Repeat this process for various classifier methods and measure the accuracy of each method.

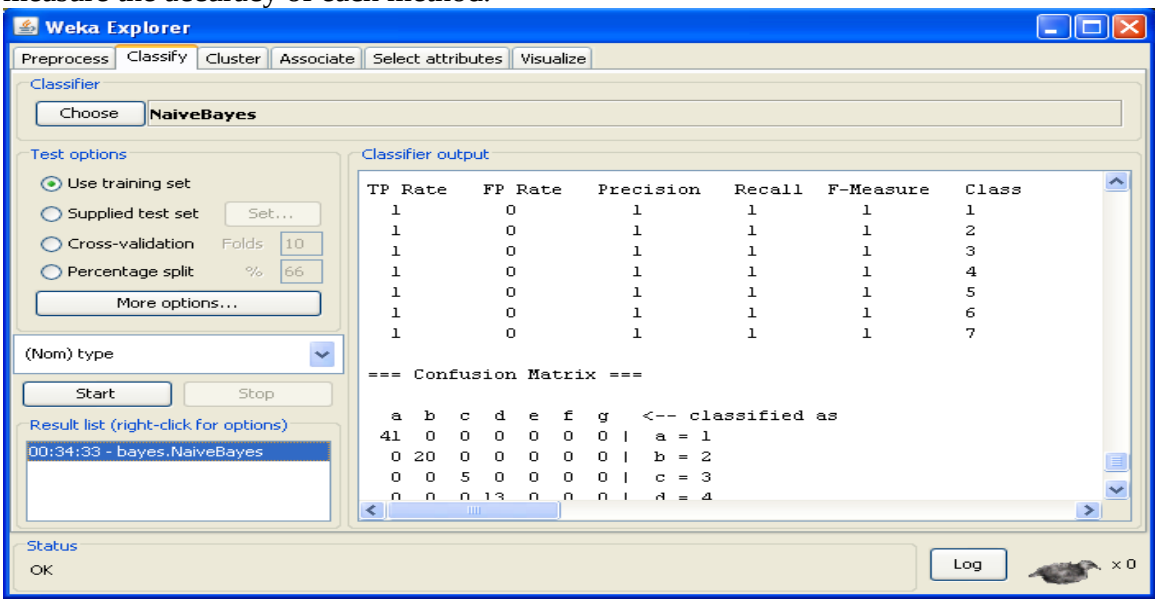


Figure 5.5: Classification without attribute selection.

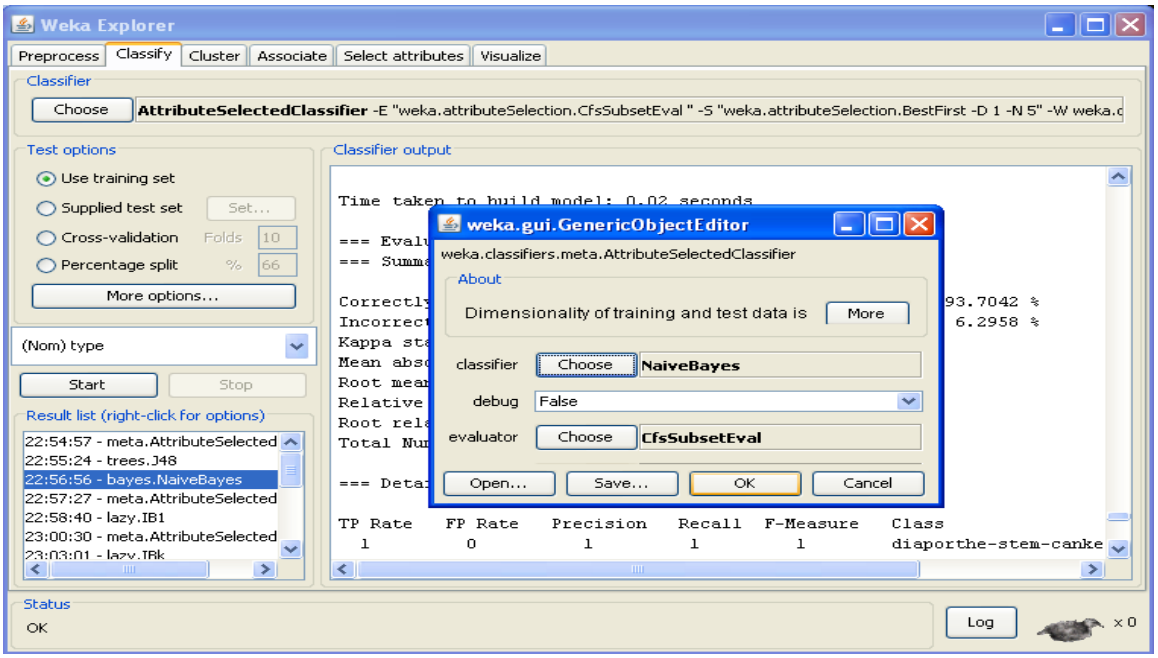


Figure 5.6: Classification with attribute selection.

Feature Selection Outputs using Weka:-

Given below are some outputs of feature selection using different methods of feature selection in Weka (data mining tool).

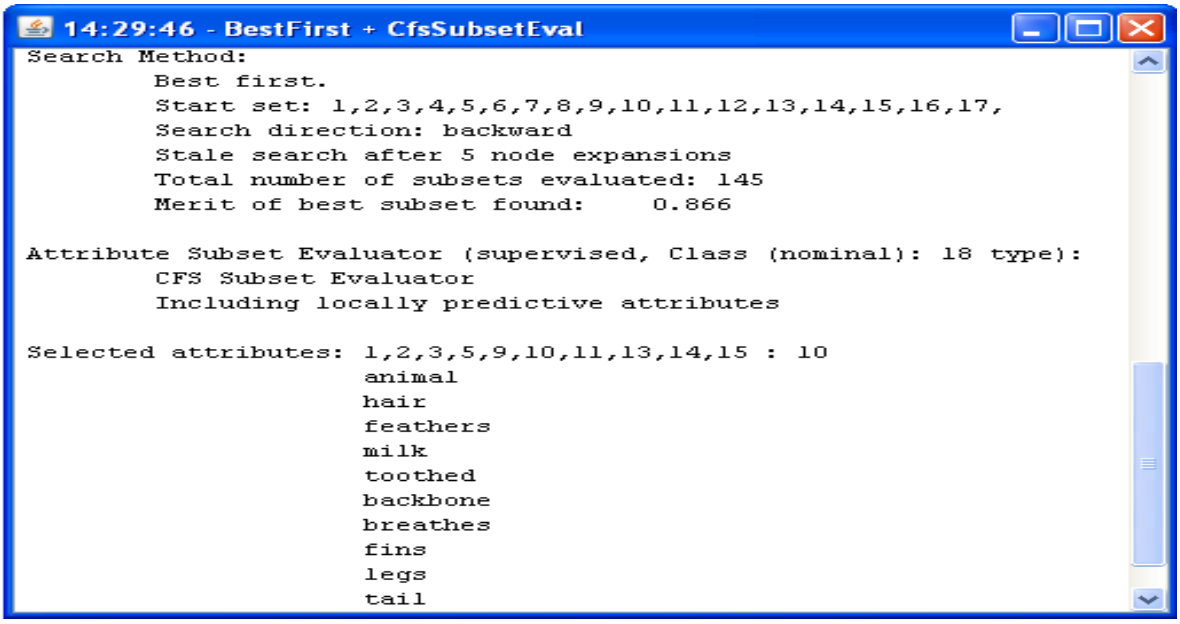


Figure 5.7: Output of attribute selection in Weka using CfsSubsetEval and BestFirst.

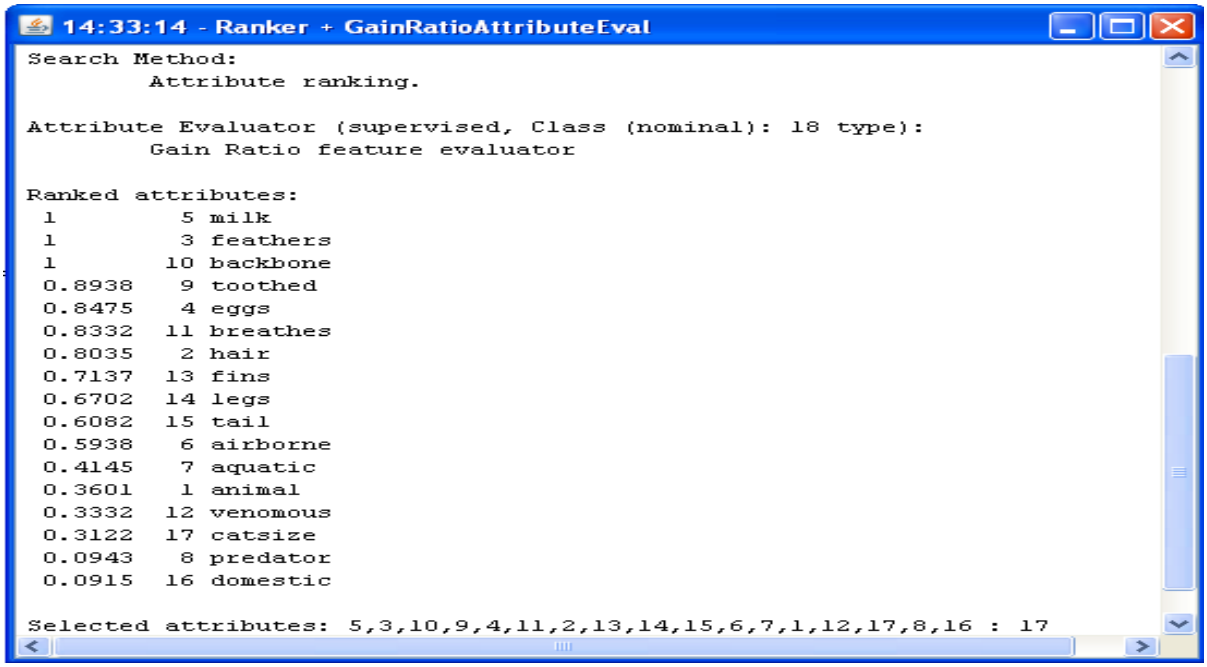


Figure 5.8: Output of attribute selection in Weka using GainRatioAttributeEval and Ranker.

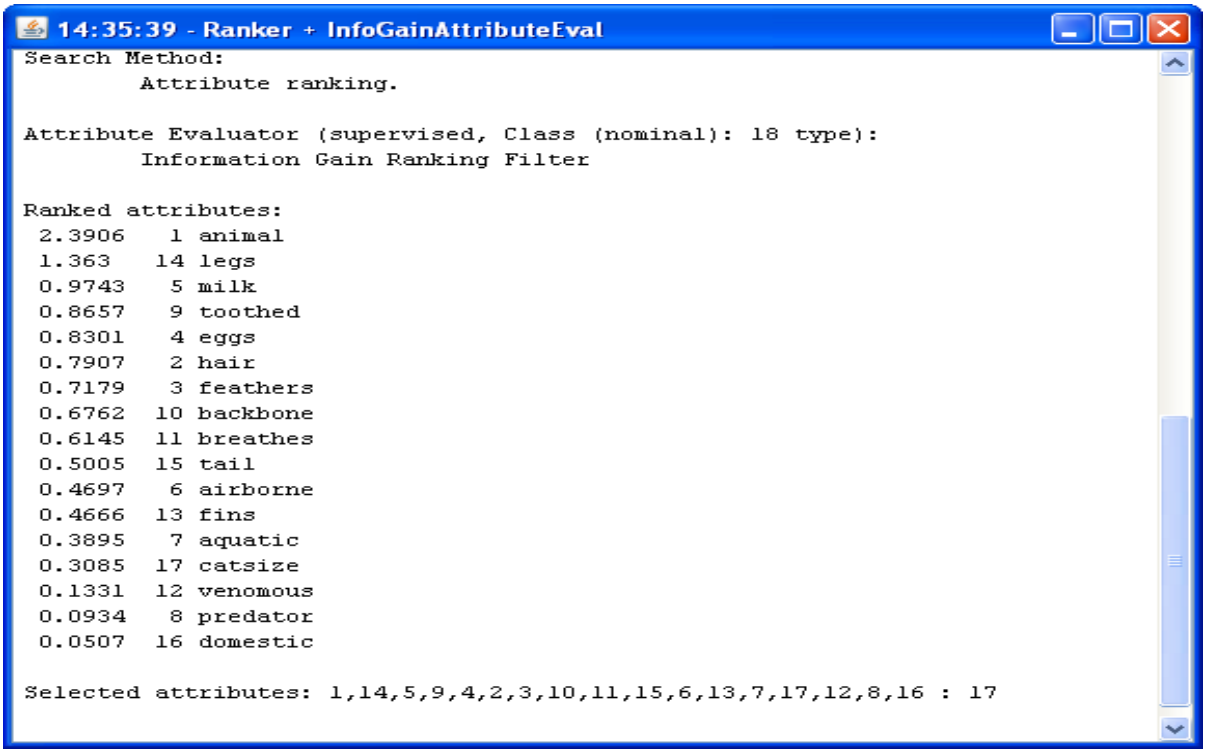


Figure 5.9: Output of attribute selection in Weka using InfoGainAttributeEval and Ranker.

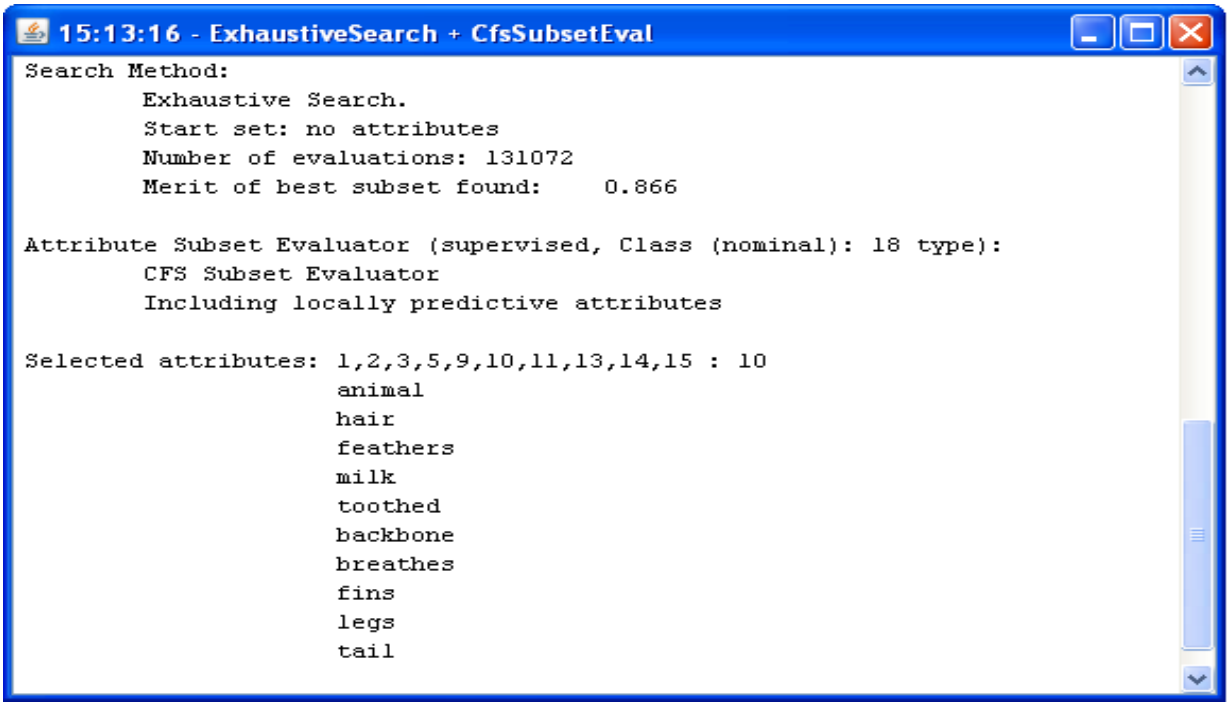


Figure 5.10: Output of attribute selection in Weka using CfsSubsetEval and ExhaustiveSearch.

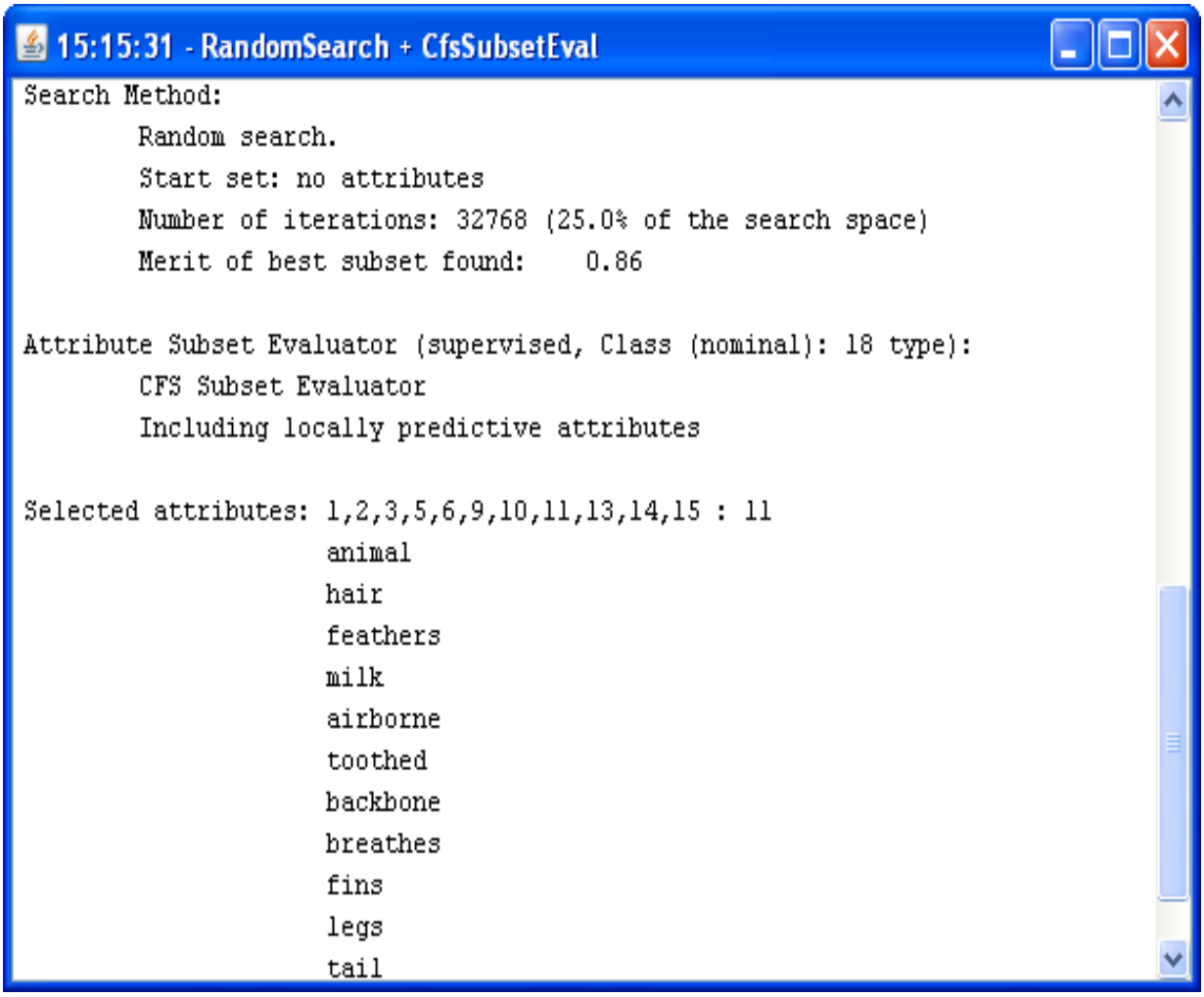


Figure 5.11: Output of attribute selection in Weka using CfsSubsetEval and RandomSearch.

Effect of features over clustering outputs:

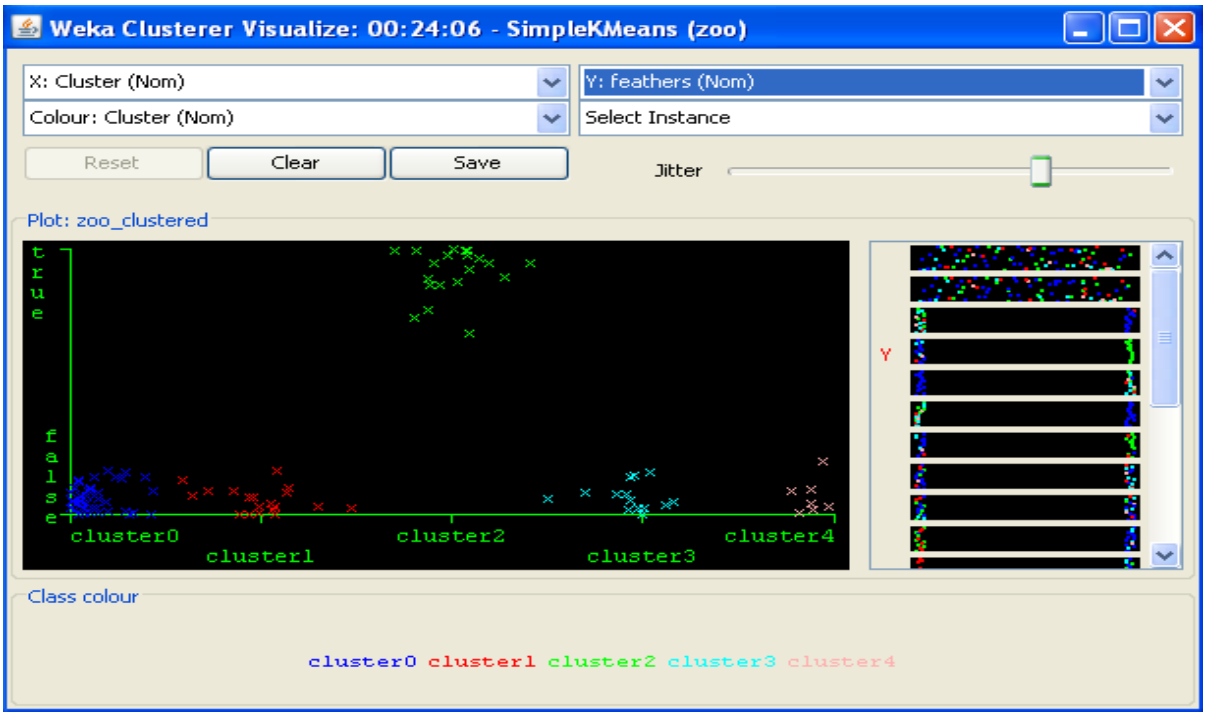


Figure 5.12: Clustering output when selected feature is "feathers".

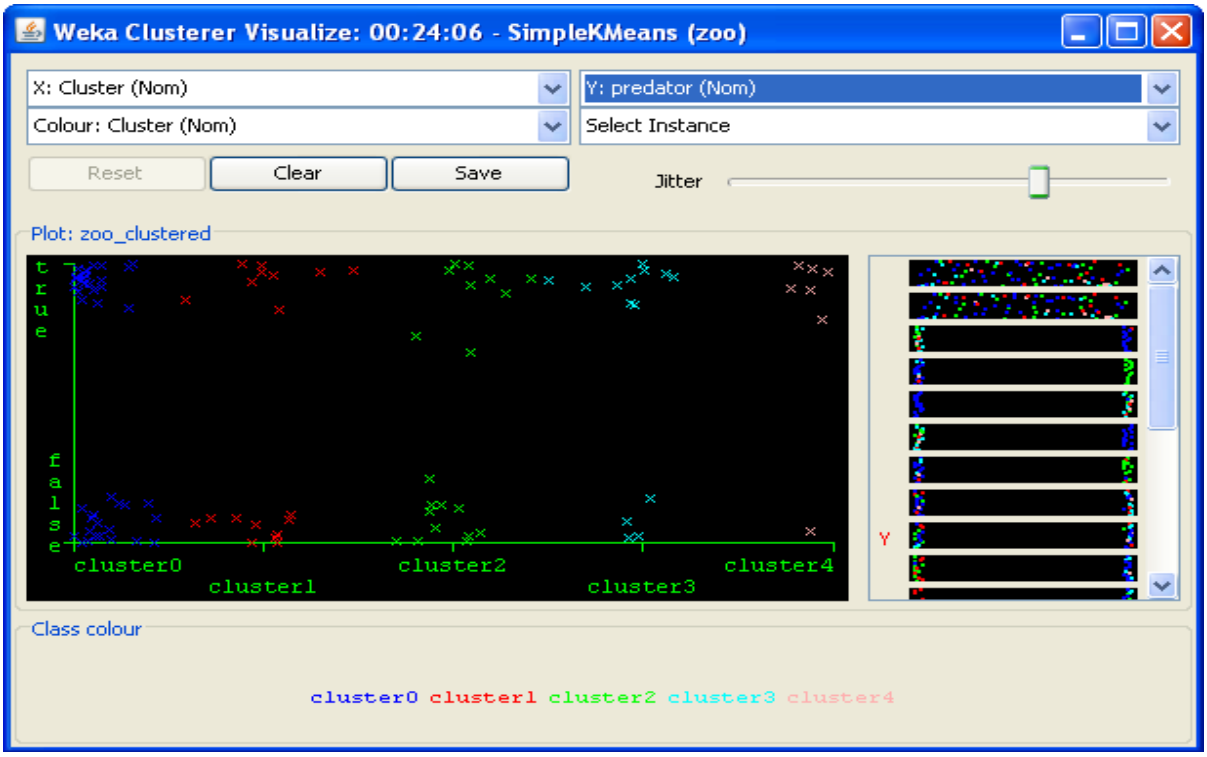


Figure 5.13: Clustering output selected feature is "predator".

Outputs using various Classification techniques for feature selection.

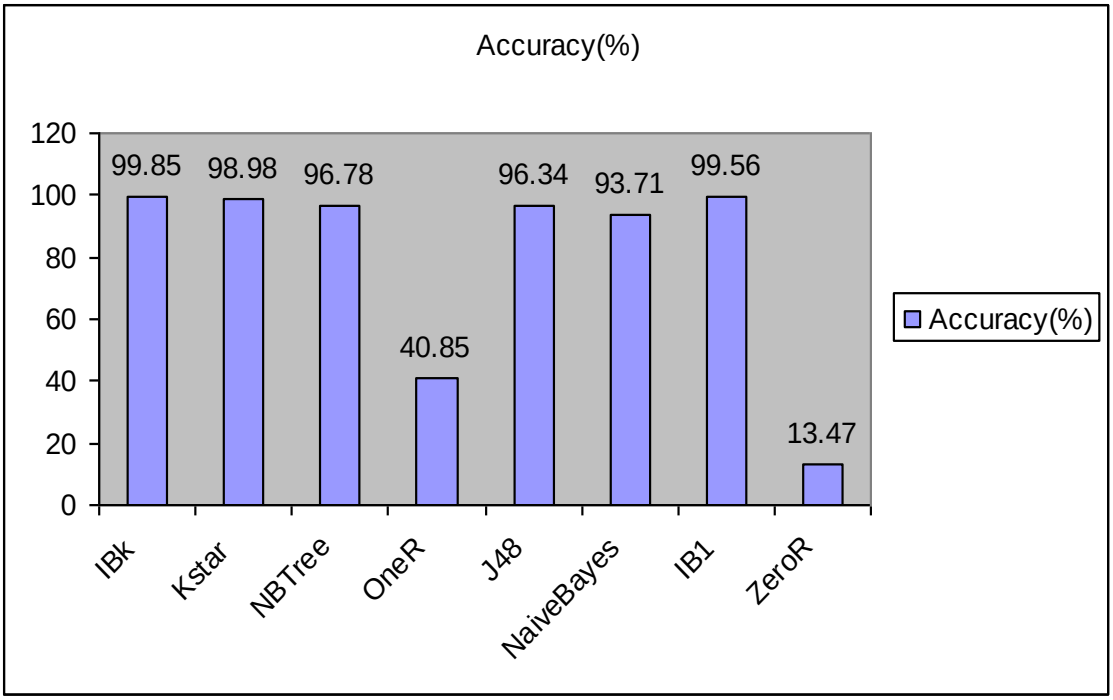


Figure 5.14: Accuracy results without using feature selection.

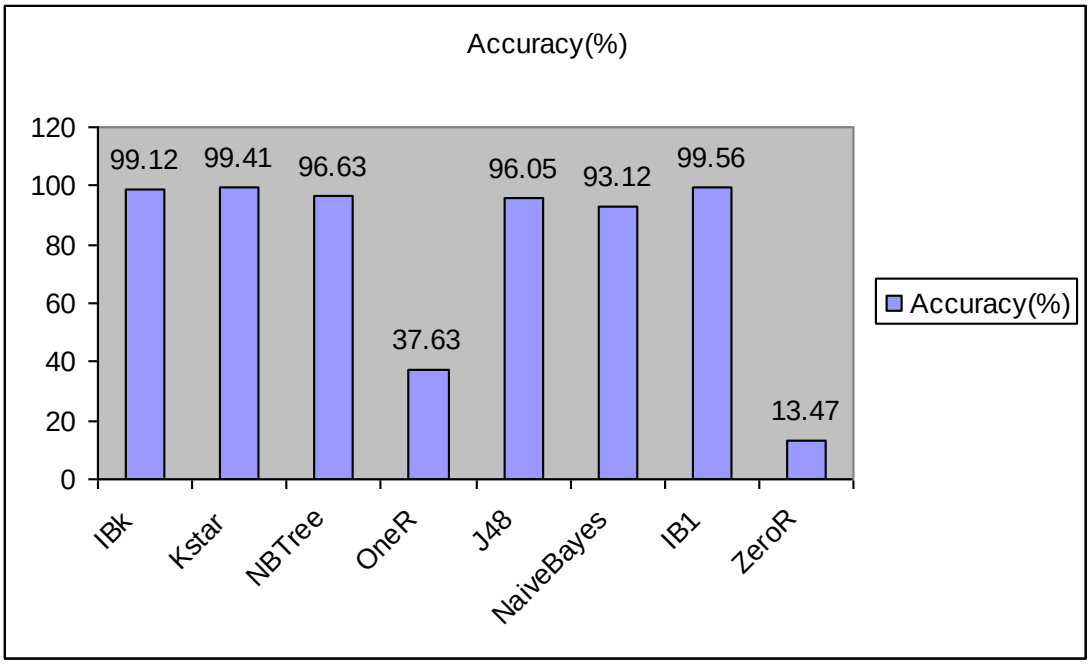


Figure 5.15: Accuracy results with feature selection.

RapidMiner

Initial screenshot of the toll RapidMiner:

Figure 5.16: RapidMiner Interface.

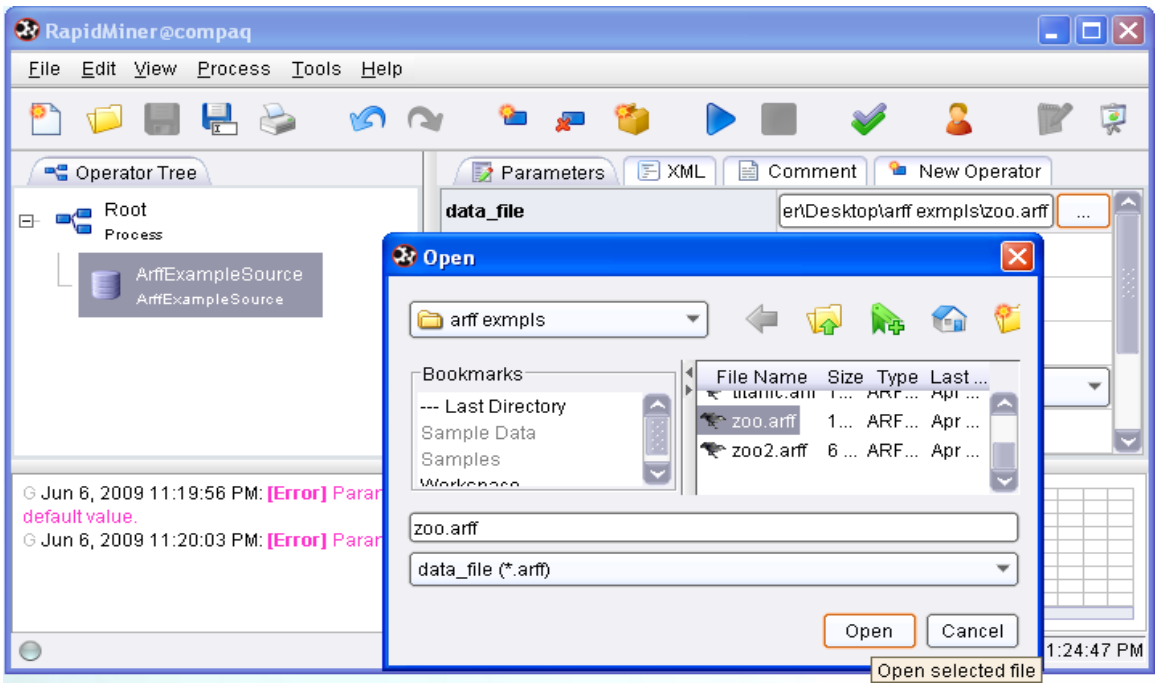


Figure 5.17: How to open a file in RapidMiner.

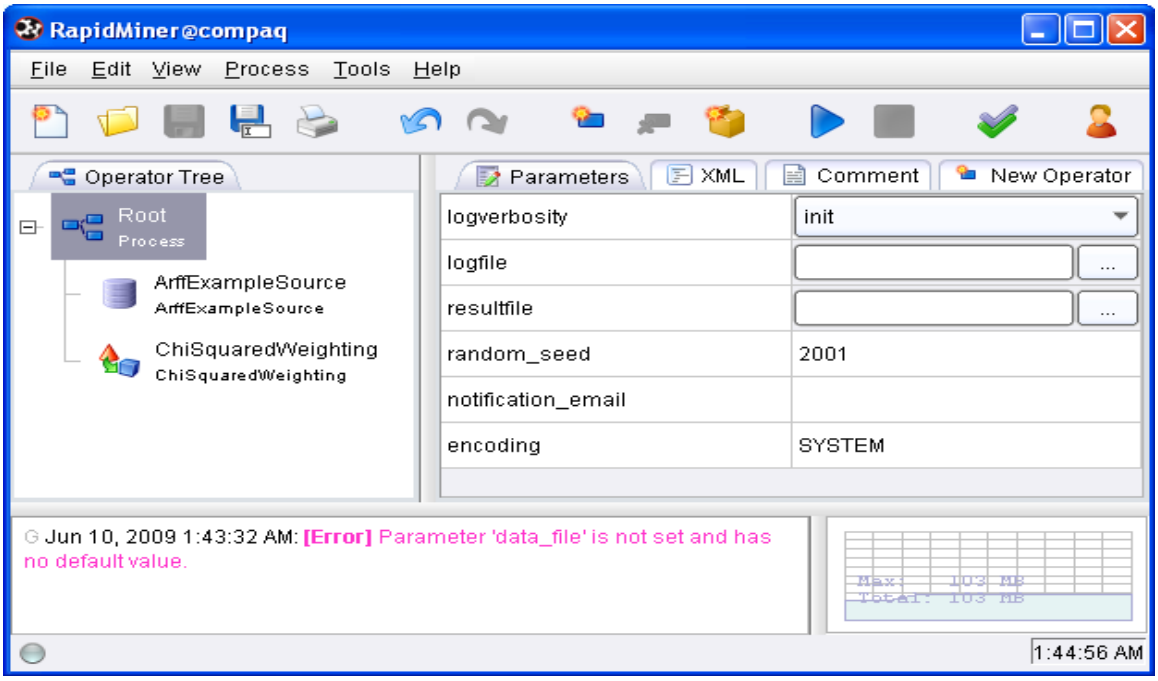


Figure 5.18: Attribute weightage process.

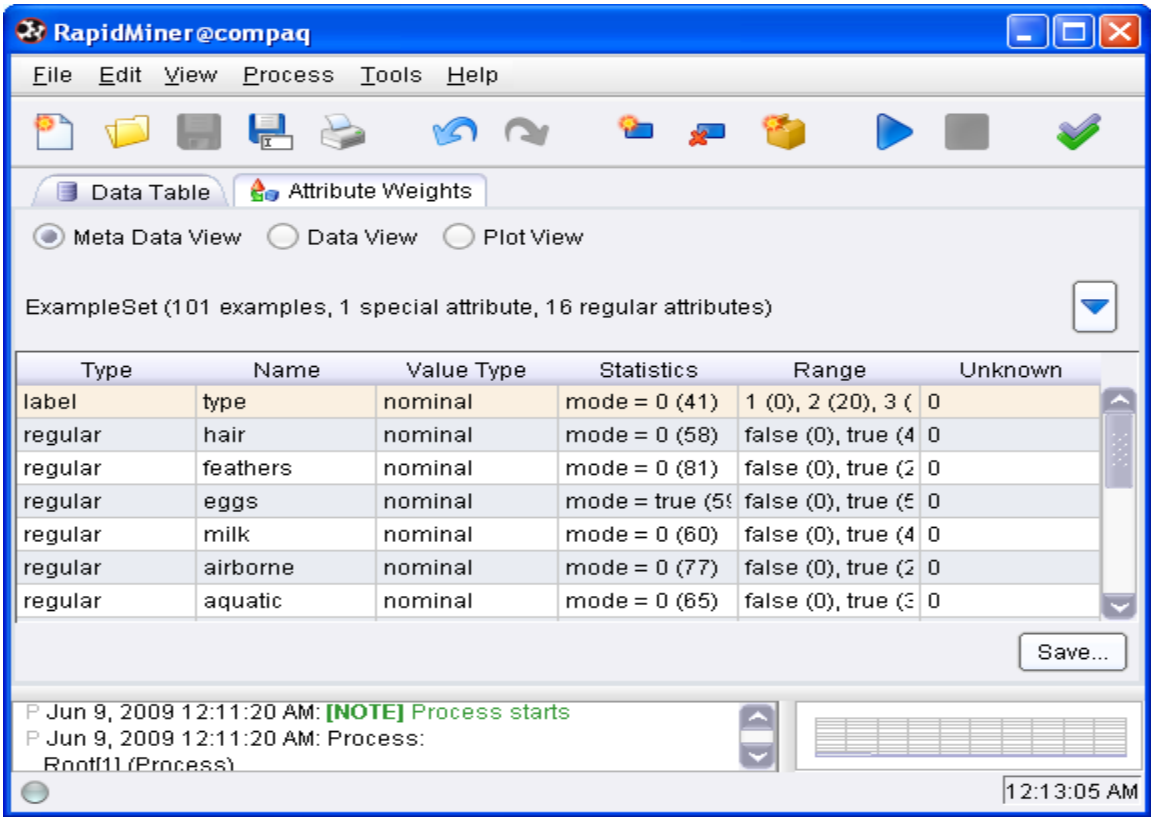
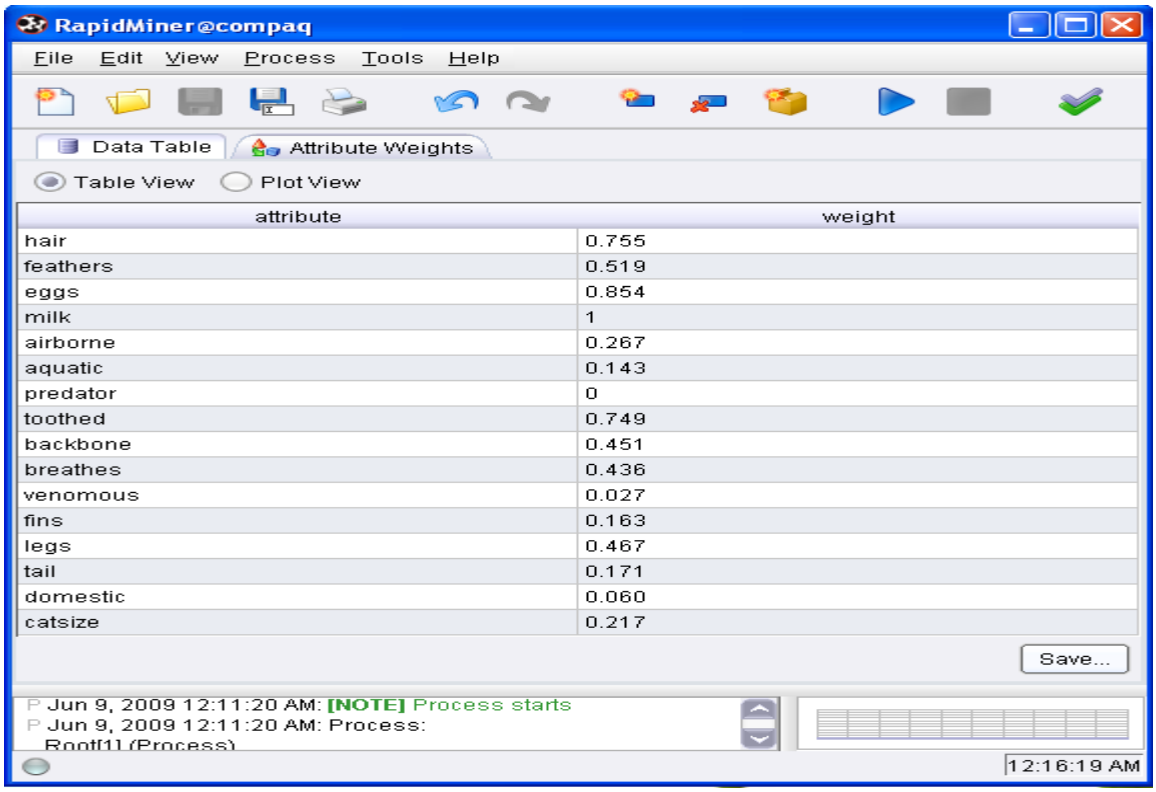


Figure 5.19: Meta Data View of "Zoo.arff" file.



attribute	weight
hair	0.755
feathers	0.519
eggs	0.854
milk	1
airborne	0.267
aquatic	0.143
predator	0
toothed	0.749
backbone	0.451
breathes	0.436
venomous	0.027
fins	0.163
legs	0.467
tail	0.171
domestic	0.060
catsize	0.217

Figure 5.20: Attribute Weights Output.

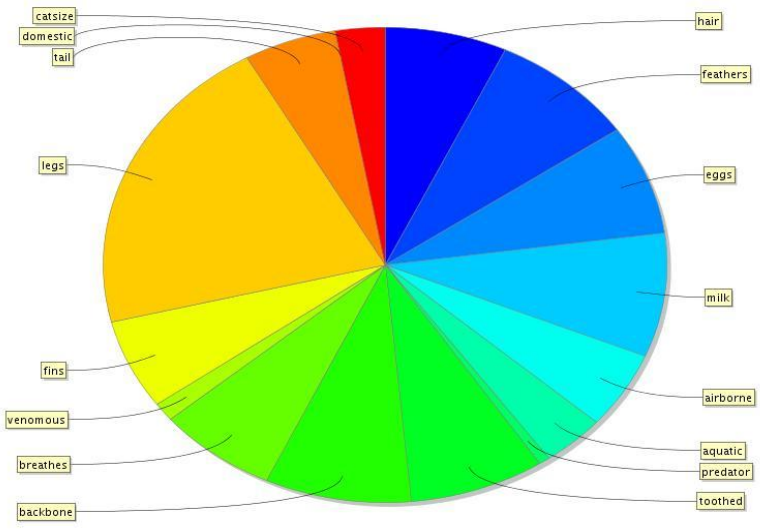


Figure 5.21: Plot view of Chi-square weighting

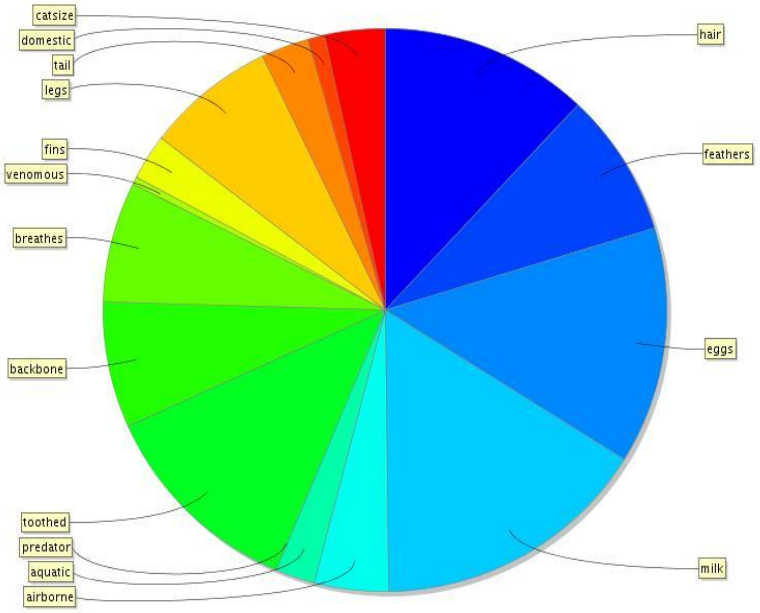


Figure 5.22: Plot view of output of Relief weighting.

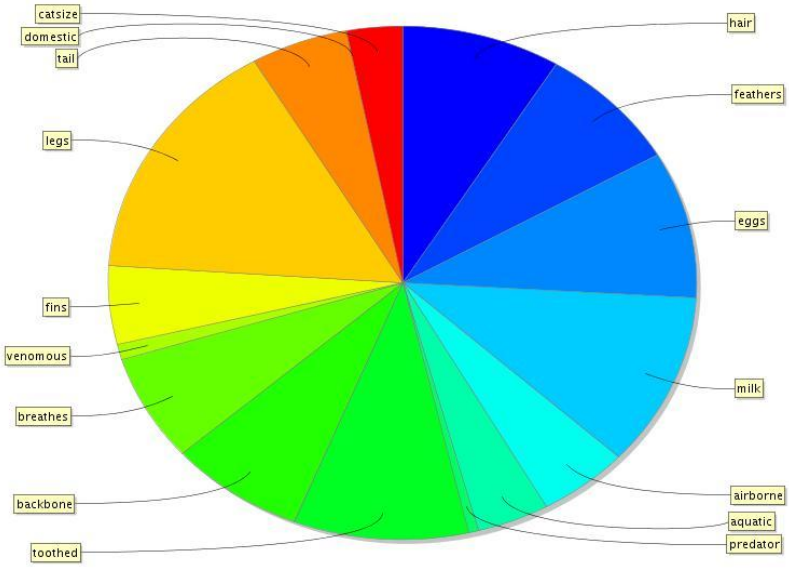


Figure 5.23: Plot view of output of InfoGain Weighting

Chapter 6
Conclusion

6.1 Conclusion

The recent developments in various methods used for feature selection have addressed the problem from the pragmatic point of view of improving the performance of textual data. There is a challenge in operating an input spaces of several thousand variables.

In this thesis an analysis on the feature selection methods is carried, and implementation of an arrf file in Rapidminer and Weka is done. The major conclusions after going through the literature review, analysis and experimentation is that choice of a good feature can contribute a lot to the classification and clustering the text documents. It has also been observed that if a special weight age to the attributes or parameters is given, which are not considered efficient enough by a particular method employed, the overall result obtained by the cluster are not good.

A comparative study of various classification methods is also done, through calculating the accuracy of all methods using Weka. Classification of data is done in two ways, without using feature selection and with using feature selection and comparative results have been studied. Through these results it can be conclude that the accuracy of classification is degraded if the appropriate features are removed by the feature selection methods.

6.2 Future scope

There is an open area of research in the combination of feature selection and discretization. Some modifications are also required in the present feature selection methods to improve the efficiency and accuracy of the classification results. To find common features from different feature selection methods is another interesting problem. The union of features selected by different methods can also be considered.

References

[1] Blum, Avrim L., and Pat Langley, “Selection of relevant features and examples in machine learning” Artificial Intelligence, pp. 245–271, 1997.

[2] Bruce A. Draper, “Feature Selection from Huge Feature Sets”, Computer Science Department, Colorado State University, Fort Collins, CO 80523, USA.

[3] C. A. Ratanamahatana and D. Gunopulos, “Feature selection for the naive Bayesian classifier using decision trees”. Applied Artificial Intelligence, Vol. 17(5-6), pp. 475–487, 2003.

[4] Dash, M., & Liu, H. (2000), “Feature Selection for Clustering”. Proc. of PAKDD-00, pp. 110-121.

[5] D. Mlademnic, M. Gtobelnik, “Feature selection for unbalanced class distribution and Naïve Bayees”, Sixteenth International Conference on Machine learning, pp.258-267, 1999.

[6] Fabrizio Sebastiani, "Machine Learning in Automated Text Categorization", ACM Computing Surveys, Vol. 34, No. 1, pp. 1–47, 2002.

[7] G. H. John, R. Kohavi, and K. Pfleger, “Irrelevant features and the subset selection problem”, International Conference on Machine Learning, pp.121–129, 1994.

[8] H. Almuallim and T.G. Dietterich, “Learning Boolean concepts in the presence of many irrelevant features”, Artificial Intelligence, pp.279-306, 1994.

[9] H. Almuallim and T. G. Dietterich, “Efficient algorithms for identifying relevant features”, Ninth Canadian Conference on Artificial Intelligence, Morgan Kaufmann, pp 38–45, 1992.

[10] Huan Liu and Hiroshi Motoda, "Feature Selection for Knowledge Discovery and Data Mining", Kluwer Academic Publishers Norwell, MA, USA,1998.

[11] I. Guyon and A. Elisseeff, “An introduction to variable and feature selection”, Journal of Machine Learning Research, Vol. 3, pp. 1157–1182, 2003.

[12] I. Kononenko, “Estimating attributes: analysis and extensions of Relief”, European Conference on Machine Learning , 1994.

[13] J. Bins, “Feature Selection of Huge Feature Sets in the Context of Computer Vision”, 2000.

[14] J. MacQueen, “Some Methods for Classification and Analysys of Multivariate Observations”, Fifth Berkley Symposium on Mathematics, Statistics and Probability, University of California Press, Berkeley, pp. 281-297, 1967.

[15] K. Kira and L.A. Rendell, “The Feature Selection Problem: Traditional Methods and a New Algorithm”, 10th National Conference on Machine Intelligence, pp. 129-134, 1992.

[16] L. Yu and H. Liu, “Feature Selection for High-Dimensional Data: A Fast Correlation-Based Filter Solution”, In ICML, pp. 856–863, 2003.

[17] Mark A. Hall, "Correlation-Based Feature Selection for Machine Learning", Department of Computer Science, NewZealand, 1999.

[18] M. Dash and H. Liu, “Feature Selection for Classification”, Intelligent Data Analysis, vol.1, no. 3, pp. 131-156, 1997.

[19] M. Modrzejewski, “Feature selection using rough sets theory”, Proceedings of the European Conference on Machine Learning, pp. 213–226, 1993.

[20] Pudil, P., Kittler, J., "Floating search methods in feature selection", Pattern Recognition Letters 15 (11),pp. 1119-1125, 1994.

[21] P. Langley and H. A. Simon, “Applications of machine learning and rule induction Comunications of the ACM, pp. 55–64, 1995.

[22] P. Langley, “Selection of relevant features in machine learning”, AAAI Fall Symposium on Relevance. AAAI Press, 1994.

[23] P. Soucy and P. Mineau, “A simple feature selection method for text classification”, Seventeenth International Joint Conference on Artificial Intelligence, pp. 897–902, 2001.

[24] R. Kohavi, “Wrappers for Performance Enhancement and Oblivious Decision Graphs”, PhD thesis, Stanford University, 1995.

[25]. R. Kohavi and G. John, “Wrappers for feature subset selection”, Artificial Intelligence, Vol. (1-2), pp. 273-324, 1997.

[26] S. Das, “Filters, wrappers and a boosting-based hybrid for feature selection”, International Conference on Machine Learning, 2001.

[27] Wilbur J.W., & Sirotkin, K. (1992), “The automatic identification of stop words”, Journal of Information Science, pp 45-55, 1992.

[28] Y. Yang, J. O. Pedersen, “A comparative study on feature selection in text categorization”, 14th International Conference on Machine Learning, pp.534-547, 1997.

[29] Y Zhao and G Karypis, "Hierarchical clustering algorithms for document datasets", Data Mining and Knowledge Discovery, pp. 141-168, 2005.

[30] http://www.cs.waikato.ac.nz/ml/weka/

[31] http://rapid-i.com/content/view/26/82/

Papers Published

1. Kamlesh Dhayal and Shalini Batra, "High Performing Feature Selection for Text Clustering," National Conference on Advances in Computer Networks & Information Technology, Guru Jambheshwar University of Science & Technology, Hisar, Haryana, pp. 486-489, March 24 – 25, 2009.
2. Kamlesh Dhayal and Shalini Batra, "Effects of Feature Selection on Clustering" in National Conference on "Emerging Trends in Software and Networking Technologies", Amity University, U.P., pp. 316-319, April 17-18,2009.