Developing New Feature Selection Methods for Discrete and Continuous Class Prediction

by
Firas Ali Al-Balas

Supervisor
Dr Kalil El Hindi

Co – Supervisor
Dr. Ahmad Al- Jaber

Submitted in Partial Fulfillment of the Requirements For the degree
of master of Science in
Computer Science.

Faculty of Graduate Studies
University of Jordan

Jun 2000

This thesis was successfully defended and approved on : 24-5-2000

Examination Committee	**Signature**
Dr. Kalil El Hindi **Asst. Prof. of Artificial Intelligence**	
Dr. Ahmad Al- Jaber **Assoc. Prof. of Algorithm Analysis**	
Dr. Ahmad Sharieh **Asst. Prof. of Parallel Processing**	
Dr. Munib Qutashat **Asst. Prof. of Data Base Management System**	
Dr. Mohmmad Al- Haj Hassan **Assoc. Prof. of Algorithm Analysis**	

Dedication

I dedicate this thesis to my parents, my wife and to my friend Issa R. Abu-Eid for his support.

List of contents

List Of tables

List of figures.

Abstract

Developing New Feature Selection Methods for Discrete and Continuous Class Prediction

by
Firas Ali Al-Balas

Supervisor
Asst. Prof. Dr Kalil El Hindi

Co – Supervisor
Assoc. Prof. Dr. Ahmad Al- Jaber

To achieve the best possible performance with a particular learning algorithm on a particular training set, a feature subset selection method should consider how the algorithm and the training set interact.

In this research, we developed a method to do feature selection by combining two approaches for feature selection (filter approach and Wrapper approach) into one approach. Experiments on 10-data sets confirm the utility of the new method improves the accuracy on most of the data sets. The new method also selects the most relevant features besides the improvement in accuracy.

The second part of this thesis is concerned with generating new attributes from a primitive set of attributes that describes the data sets. The generation is done by developing a new method for constructive induction on data sets labeled by continuous classes. Experiments on real and artificial data sets show that this new method improves the accuracy and generates a few numbers of attributes.

Chapter 1 : Introduction

1.1 Machine learning

Artificial intelligence (AI) is a sub-field of computer science which mainly aims “to make computers more useful and to understand the principles which make intelligence possible “(Winston, 1977). (Barr, and Feigenbum, 1982) stated that (AI) is “ concerned with designing intelligent computer systems, such as: learning, solving problems and understanding languages”.

Machine learning is a sub-field of (AI) that aims to develop computational methods that would implement various forms of learning in particular mechanisms capable of inducing knowledge from examples or data. The main purpose of machine learning is to employ a learning system that will acquire higher-level concepts and problem solving strategies through examples in a way analogical to human learning.

Several tasks have received attention in machine learning literature. One of these tasks is the method of determining the most relevant features for use in representing the data sets. In our thesis, we will discuss several approaches of this method and develop a new one.

In the next section, we will define a set of definitions that should be understood before going into the rest of this thesis.

1.2 Definitions

An important set of terms must be defined to make the rest of this thesis easily understood.

First, we can define an instance, sometimes called an example, as a fixed set of features values. An instance describes the basic entity to deal with, such as: a board position in a chess game or a DNA sequence.

A feature, sometimes called an attribute, describes some characteristics of an instance. There are two main kinds of features (attributes): nominal and continuous. A nominal attribute, such as color and shape, has a fixed set of values (e.g. red, green, blue). Continuous attributes that have a maximum and minimum values are an ordered set of values such as temperature. Every instance may contain a combination of the two kinds of attributes in addition to a categorical attribute (or class attribute) which describes the phenomenon to learn and make prediction about their values. The class may also be nominal or continuous. An instance may contain unknown (or missing) values for some features, which influence the evaluation of accuracy.

A training set is a group of instances used by machine learning algorithms to build a classifier. Whereas the classifier is a function that maps test instances to predicted classes. It may be a decision tree or a set of rules.

1.3 An overview of learning algorithms

In this thesis some learning algorithms are used to test our work; according to our use, these learning algorithms can be divided, into two types: -

1) learning algorithms to evaluate the accuracy of our work in feature selection
2) learning algorithms to evaluate the accuracy of our work in constructive induction for continuous classes

Next we will discuss each type in details.

The learning algorithms we have used to evaluate our work in feature selection are Naïve Bayesian, Instance – Based and Decision tree learning algorithm.

1.3.1 The Naïve Bayesian algorithm

The Naïve Bayesian Classifier (Langley, et al.,1992) uses Bays rule to compute the probability of each class depending on the given instances and assuming that the attributes are independent. The predicted class is the one with the higher probability.

The probabilities for nominal attributes are estimated from training set where the probabilities for continuous attributes are assumed to be normally distributed and taken from Gaussian Distribution, by estimating the mean and standard Deviation from data.

1.3.2 C4.5 algorithm

The C4.5 Algorithm (Quinlan, 1993) is a top – down decision tree algorithm like ID3 (Quinlan, 1986) with a pruning method. The tree is constructed by finding the best single – feature test to construct at the root node of the tree. After choosing the root node of the tree, the data is split according to the attribute values (test values). Then, sub-problems are solved recursively yielding sub-trees. The process continues until we reach homogeneous data partition in which all instances are of the same class.

C4.5 uses gain ratio as a feature selection measure to select the attribute to be used in the test; other measures have been proposed such as Gain Index (Breimen, et al., 1984), Distance – based measure (Mantaras, 1989) and **RELIEFF** (Kononenko, 1994).

C4.5 prunes the estimated classifier by using the upper bound of a confidence interval on the substation error as the error estimate; since nodes with fewer instances have a wider confidence interval, they are removed if the difference of error between them and their parents are not significant.

1.3.3 Instance – Based learning algorithm

Instance - Based learning algorithm (IBL) (Aha,et al.,1991) is an extension to the nearest neighbor algorithm. The main idea of (IBL) algorithm is to use the nearest (K) instances to predict a class for a new instance. The user chooses K (number of nearest instances to be used) that usually has the values of 1,2 or 3. This learning algorithm is easy to implement and understand. At the same time, it has a disadvantage that the choice of distance function highly influences the algorithm accuracy.

The next type of learning algorithms used in this thesis is concerned with continuous classes. These algorithms are Regression Rules (Issa, 1998) and Regression Tree learning algorithms (Quinlan,1992).

1.3.4 Regression rule learning algorithm

This algorithm construct rules from examples (instances) after building and pruning a tree.

Finally, a set of rules is generated from the training data. When an instance from test data is entered, the algorithm applies the rule the conditions of which covers the new instance and compare the value of the test instance target with the value of output rule.

1.3.5 Regression tree learning algorithm

This algorithm developed by (Quinlan,1992) it constructs a linear model in a form of a model tree. The linear model is at the leaf of the tree, which represents the relation between the class values of the training cases to their attribute values rather than a class name as in decision trees.

The two types of learning algorithms discussed above will be discussed in more details in the next two chapters.

1.4 Thesis goals

At this section we will present the motivation and goals of our work in this thesis. These goals are:

1) Selecting the most relevant attributes from the initial set of attributes to describe a data set.
2) Constructing new attributes from the primitive attributes to describe the data set with continuous classes. The aim is to achieve better generalization accuracy

1.4.1 Using RELIEFF as a wrapper approach

Irrelevant attributes may degrade the performance of the classification in terms of its prediction accuracy. These irrelevant attributes are difficult to find in real – world data sets. So removing these irrelevant features will improve the accuracy. Feature selection methods are used to select the most relevant features from a set of features that describes the data set as all attributes do.

There are three main category approaches developed to deal with the method of selecting the most relevant features, these approaches are:

1) The Filtering Approaches (Kohavi, et al., 1994). These approaches filter out irrelevant features before the process begins such as RELIEFF (Kononenko, 1994).
2) The Wrapper Approaches (Kohavi, and George, 1998). These approaches select the relevant attributes by wrapping around the induction algorithm.
3) Embed Approach. Algorithms on this category are integrated as a part of the learning algorithm such as ID3 (Quinlan, 1986).

RELIEFF (Kononenko, 1994) is a method that has been used to give a numeric weights to attributes (features). Attributes with low weights (below a certain threshold) are considered irrelevant and are filtered out .In other words, RELIEFF has been used as a filtering approach.

In this thesis, we use RELIEFF as a wrapper approach and evaluate its utility.

1.4.2 Performing constructive induction for continuous classes

The ability of inductive learning systems to find a solution to a given problem is dependent upon the representation of the features of the problems. These features (attributes) may be inadequate for the learning task when they are weakly relevant. Constructive induction methods are general methods for coping with this kind of problems by improving the representation space by generating additional attributes to the original (primitive) attributes. So the main goal of constructive induction is to automatically generate new features (Michalski, and Bloedron,1998), yielding improvement in classification accuracy.

There are several constructive induction methods such as FRING (Pagallo, and Haussler,1989) and GALA (Hu, and Kibler,1996) which differ in the strategy of constructing new attributes, but all of these methods work on problems (Data Sets) with nominal classes.

We investigate the utility of some of these techniques for problems with continuous classes.

Experiments on five artificial data sets under two learning algorithms –Regression Tree (Quinlan,1992) and Regression Rule(Issa,1998) – achieve higher accuracy on 4 of the data sets.

1.5 Structure of this thesis

Chapter 2 presents an overview of feature selection methods.

Chapter 3 describes the combined approach for feature selection, together with empirical evaluation of the new method in 10-data sets from the UCI repository.

Chapter4 describes a new method for constructive induction on continuous class problems, with an empirical evaluation on artificial data sets.

The last chapter summarizes our conclusion and directions for future work in constructive induction and feature selection.

At a conceptual level, one can divide the task of concept learning into two subtasks: - deciding which features to be selected to describe the concept and which attributes to be combined to produce a new feature.

At a practical level we would like the induction algorithm to scale well to domains with many irrelevant features.

2.2.1 Definition of relevancy

Many definitions in machine learning clarify when features are relevant. In the following we will mention some of these definitions that are reviewed by (Langley,and Blum , 1992).

Definition 1 (*Relevant to target*)

A feature (Xi) is relevant if there exists some examples in the instance space for which twiddling the value of (Xi) affects the classification given by the target concept.

A drawback of this definition is that the learning algorithm only accesses the sample (S) that makes it, in some cases, unable to determine the feature to be relevant or not.

Definition 2 (*Strongly Relevant to the sample |distribution*).

A feature (Xi) is strongly relevant to a sample (S) if there exists examples (A) and (B) in (S) that differ only in their assignment to Xi and have different classes $C(A) \neq C(B)$. So (Xi) is strongly relevant to target (C) and distribution (D) if there exists examples (A) and (B) having non-zero probability over (D) that differ only in their assignment to (Xi) and satisfy $C(A) \neq C(B)$.

The difference between this definition and the previous one is that the two examples (A) and (B) are required to be in (S).

Definition 3 (*Weakly relevant to sample |distribution*)

A feature (Xi) is weakly relevant to a sample (S) (or target (c) and distribution (D)) if it is possible to remove a subset of features so that (Xi) can become strongly relevant.

Definition 4 (*Relevance as a complex measure*)

Let r(S, C) be the number of relevant features using definition 1 - for a sample of data (S) and a set of Classes (C) – that have the least of errors over (S) and have the fewest relevant features.

In other words, the definition is looking for the smallest set of features that has the least error over (S) via a concept (C). This definition takes class (C) into consideration because there are many features that are highly relevant from point of view of the information contained, but they are useless with respect to the classes if taken into consideration.

One can observe that the above definition is independent of the used learning algorithm. There is no guarantee that when the feature is relevant then it will be useful for the induction algorithm. The next definition takes this idea into consideration.

Definition 5 (*Incremental usefulness*)

Given a sample of data (S) and a learning algorithm (L) and a feature set (A). Then a feature (Xi) is incrementally useful to (L) with respect to (A) if the hypothesis that (L) produce using the feature set {A} Ù Xi is better than the accuracy achieved using just the feature set (A).

2.2.2 Direction of search

The selection of a subset of features to be used to describe the data set and give good accuracy under learning algorithms can be seen as an optimization problem. This search space of optimization contains all possible subsets of features, so its size is $\mathbf{2}^{n}$ where n is the number of features used to describe the data set and it is an NP complete problem(Caruana,and Fring,1994). Different methods have been developed and used for feature selection in machine learning using different strategies and evaluation functions .The following search strategies are commonly used in feature subset selection.

1) Forward Selection (FS). Start with an empty set of features and add features one at a time until all attributes are added or stop criterion is found. At each step, FS adds the attributes that when added, they improve the accuracy of the algorithm. Once an attribute is added, forward selection cannot remove it later.

2) *Backward Elimination (BE).* Start with a feature set containing all features and remove features one at a time until a stop criterion is found. As in Forward selection, backward elimination removes at each step the attribute whose removal will improve the accuracy of the algorithm on a particular learning algorithm. Once an attribute is removed, backward elimination cannot later add it.

3) *Forward Stepwise (Sequential) Selection.* Start with an empty set and add or remove, at each step, an attribute that will improve the accuracy. This strategy differs from (FS) that allows removing an attribute added before.

4) *Backward Stepwise (Sequential) Selection.* This strategy starts with all attributes and removes, or adds an attribute at each time that improves the accuracy. The main difference between this strategy and (BE) is that this strategy allows to add an attribute that is removed before.

2.2.3 Feature selection as a heuristic search

The most convenient paradigm for dealing with data sets that contain large number of irrelevant attributes, is the heuristic search where a subset of possible features is specified in each search space .

The followings are the strategies to evaluate alternative subsets of attributes that determine the nature of heuristic search:

1- determining the starting points in the space, which influence the direction of search.

2- organizing direction search. It is not practical to have an exhaustive search in the space, because there exist 2^n possible subsets of attributes. A good method for organizing the search is by making changes on the set of attributes at each point by using the search strategies discussed above.

3- evaluating the alternative subset of attributes, this means how the learning algorithm works.

4- halting the search. There are several ways to halt the search; for example, one can stop adding or removing attributes when none of the alternatives improves the estimate of classification accuracy; one might continue to revise the feature set as long as accuracy does not degrade.

We must note that the above design decision must be made for any induction algorithm that carries out feature selection.

2.3 Filtering approach algorithms for feature selection

Kohavi, et al. (1994) introduced a method that filters out irrelevant features before the induction step occurs and called it a filter approach. This approach is a preprocessing step that uses general characteristics of the training set to select some features and exclude others. So this approach is independent of the induction algorithm.

Figure 2.1 shows a diagram for this approach for feature selection.

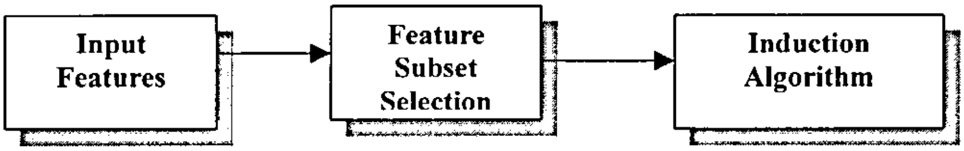

Fig.2.1 *The filter approach*

In the next three sub-sections, we will discuss three most common filter approach algorithms.

2.3.1 A probabilistic approach for feature selection

The proposed probabilistic approach is Las Vegas Feature Selection Algorithm (Liu, and Setiono,1996). (LVF) algorithm makes a probabilistic choice to guide them to select a subset of features from all features. It uses randomness to guide their search in the features and use an inconsistency check for the selected subset on training data using learning algorithm. Then it stores the random subset that has the smallest inconsistency check.

This algorithm gives good results in finding the relevant attributes by running this algorithm on artificial data sets where the relevant attributes are known during the generation of data sets.

2.3.2 FOCUS algorithm

Dietterich, and Almuallim, (1991), developed this algorithm which looks for the minimal subset of features starting from one feature and increases the size of the subset by adding features until a sufficient subset is encountered. The feature subset is said to be sufficient if there is no conflict in it, where conflict means that there is no pair of examples that have different class values and have the same values for all the features in the selected subset.

2.3.3 The RELIEF algorithm

Kira, and Rendell, developed an algorithm called RELIEF. This algorithm is used to estimate the quality of attributes .The key idea of this algorithm is to estimate the attributes (features) according to how well their values distinguish between instances that are near to each other. So RELIEF for a given instance searchs for two nearest instances for that instance: one from the same class called nearest hit and the other from the different class called nearest miss.

RELIEF evaluates a weight W [A] for each feature as an approximation of the features difference of probabilities:

W [A]=P (different value of A | nearest instance from different class)

- P (different value of A | nearest instance from same class).

The above rational means that good attribute should differentiate between cases from different classes and should have the same values for instances from the same class. In other words, the good attribute is the one that has the large value of W [A] between all attributes.

The original RELIEF (Kira,and Rendell ,1992) can deal with discrete and continuous features but limited to only two-class problems. Figure 2.2 shows the original RELIEF algorithm.

```
H: Nearest hit instance.
M: Nearest miss instance.
m: Total number of instances.
Begin
    Set all features weight to zero W [A]=0;
    For I=1 to m do
    Randomly select an instance R.
    Find Nearest hit H and nearest miss M.
    For A=1 to all_attributes do
W[A]=W[A]- diff (A,R,H)/m + diff (A,R,M)/m
    End;
```

Fig 2.2:*Simple RELIEF for feature selection.*

Where diff (Attribute, Instance 1, Instance 2) calculates the difference between the values of attributes for two instances .The function is also used to calculate the distance between instances to find the nearest neighbors .The total difference is the sum of differences over all attributes. The distance measure used in this algorithm will be discussed later in this chapter.

As we mentioned original RELIEF is limited to two-class problems, but what about problems with more than two classes? (Kononenko, 1994) developed extensions to original RELIEF to deal with multi – class problems, the extensions are:

RELIEF-E the nearest miss of the instance (I) is defined as the nearest neighbor from different class. This is a straightforward extension of original RELIEF.

RELIEF-F Instead of finding one nearest miss M from different class, the algorithm finds one nearest miss M(c) for each class and average their contribution for updating the weight of an attribute W[A]. The average is weighted with the prior probability of each class.

$$W[A] = W[A] - \mathrm{diff}(A,R,H)/m + \Sigma\ [p(c) * \mathrm{diff}(A,R,M(c))]/m$$

RELIEF-F algorithm can be shown on Figure 2.3

```
•   Input   A: set on non-categorical attributes.
            C:  the categorical attribute.
            T:  set of training cases.
•   Output W [A]: Weight of attributes A.
Function Estimating Attributes Weights
•   Begin
•      For Each attribute A, let  $W[A] := 0$
•        Begin
•          For Each case I in the training set (of size N)
•            Begin
•               Randomly select a case R.
•               Find nearest hit H.
•               For Each class $c \neq class(R)$ find the nearest miss $M[c]$
•               For A := 1 to all attributes  do
•            $W[A] := W[A] - diff(A, R, H)/N + \sum_{c \neq class(R)} (P(c) * diff(A, R, M))/N$
•          End.
•        End.
•   End.
```

• **Fig. 2.3:** *Estimating the weight of attributes from the training set.*

RELIEF-F is an algorithm that estimates the quality of attributes with and without dependencies between attributes and deals with discrete and continuous attributes for data sets with multi – class problems.

2.4 Embedded approach algorithms for feature selection

Techniques for this approach are integrated as a part of the induction algorithm.

ID3 (Quinlan, 1986) and C4.5 (Quinlan, 1993) are examples of induction algorithms that integrate a feature selection techniques. These algorithms carry out a greedy search through the space of decision trees. At each stage, an evaluation function (e.g. Gain Ratio) is used to select a feature that has the ability to discriminate among classes. It partitions the training data based on this attribute (feature) and repeats the process until all features are selected or a stop criterion is fired.

Divide and Conquer methods for learning decision trees (Quinlan, 1986) use an evaluation function to select a feature that helps to distinguish a class C from others, then add the resulting test to a single conjunction rule for C. This process is repeated until the rule excludes all members of other classes, then it removes the members of C that the rule covers and repeats the process on the remaining training cases.

This approach explicitly selects features for induction in branch or rule, in preference to other features that appear less relevant or irrelevant. (Kononenko,1994) expected it to scale well with domains that involves many irrelevant features.

2.5 Wrapper approach algorithms for feature selection.

Techniques in this approach are wrapped around the induction algorithm. (Kohavi, and George,1998) called these a wrapper approaches. These approaches search in the data set for a subset of features and evaluate an alternative sets by running some induction algorithm on the training data and use the estimated accuracy of the resulting classifier as a measure for this new subset. Figure 2.4 shows a diagram for wrapper approach. This diagram shows that searching for a good subset using the induction algorithm itself as a part of evaluation function.

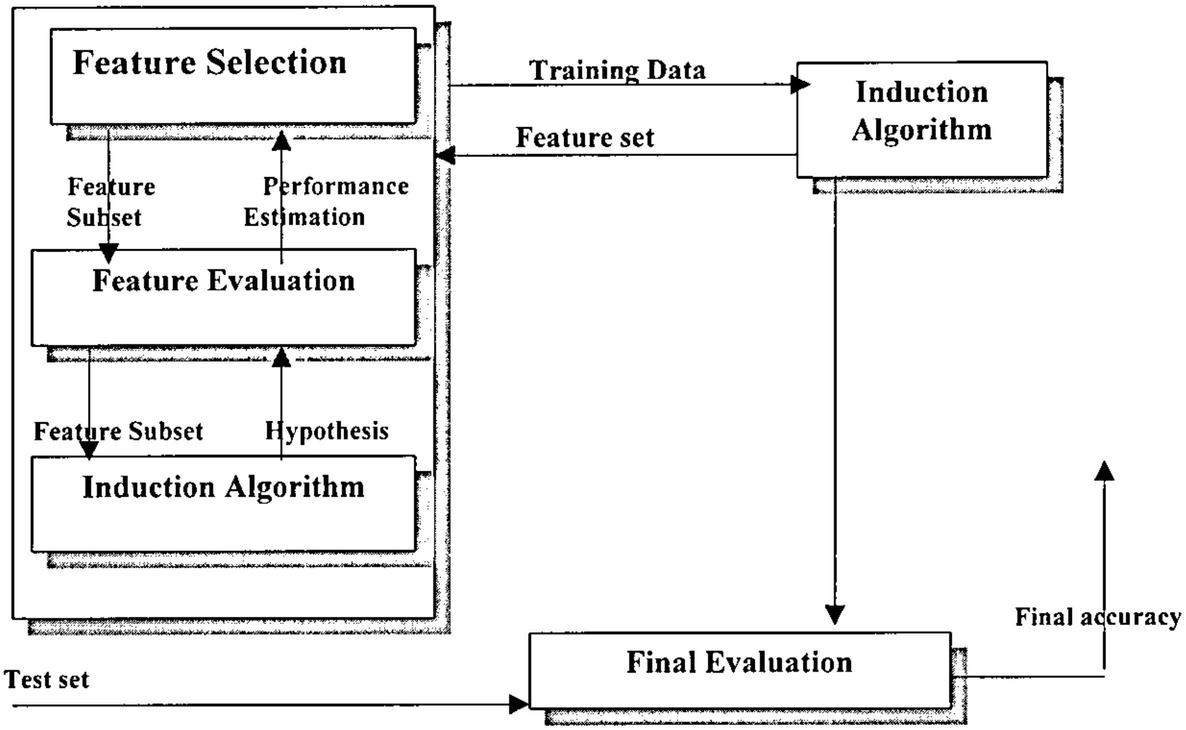

Fig 2.4: *The wrapper approach.*

Wrapper approach has some disadvantages. First, the computational cost becomes from calling the induction algorithm for each new feature added or removed from the subset of features. This disadvantage appears very well on data sets that contain a lot of features. The second disadvantage is that the big influence on selecting the induction algorithm that will give us a decision about the relevancy of features.

On the other hand, wrapper approach also has advantages. First the feature subset selected by this approach will give a small size of trees when used on ID3 or C4.5 algorithm which make the understanding of the domain better. Second advantage is that this approach allows us to observe the relevancy of each feature because the accuracy of the subset is evaluated at each time a feature is added or removed.

2.6 Heterogeneous distance functions

During the evaluation of the quality of attributes using RELIEF, this algorithm needs to find two near instances to the given instance: - one is from the same class (nearest hit) and the other instances from different class (nearest miss). To find these two instances, a distance function should be used.

There are many distance functions that have been proposed to decide which instance is closest to a given input vector.

Many of these metric functions work well for nominal attributes.

The value difference metric (VDM) was introduced to define an appropriate distance function for nominal attributes. The modified Value Difference Metric (MVDM) uses a different weighing scheme than VDM and is used in many systems. These two distance functions work well in domains with nominal attributes but they do not handle continuous attributes directly. Instead, they rely upon discretization that may degrade the accuracy.

Many real-world applications have both nominal and continuous attributes. The choice of a distance function influences the behavior of an algorithm. (Wilson, 1997) proposed that no distance function can be strictly better than any other in terms of generalization when considering all possible problems with equal probability. However, when there is a higher probability of one class of problems occurring than another, some learning algorithms can generalize more accurately than others can.

In the next section, we will discuss the most common distance functions used in machine learning algorithms and we will discuss in detail one of these distance functions which will be used in our work.

2.6.1 Common distance functions

As we mentioned above, there are many learning algorithms that depend on a good distance function to be used. The most commonly functions used are: -

1. Euclidean distance function

This distance function can be defined as: -

$$\mathrm{E}(x,y)=\sqrt{\sum_{a=1}^{m}(x_a-y_a)} \quad ....(2.1)$$

Where (x) and (y) are two instances (one is the stored instance and the other is the input instance) and (m) is the number of attributes. The square root is often not computed because the closest instance(s) will still closest regardless of whether the square root is taken.

The main weakness of the basic Euclidean Distance Function is that if one of the input attributes has relatively large range, it can overpower the other attributes. For example, if an application is implemented by two attributes (A) and (B) where (A) can have values between 1 and 1000, and B can have values between 1 and 10, then (B)'s influence on the distance function will be overpowered by A's influence. Therefore, dividing the distance for each attribute over the range of that attribute often normalizes distance functions, so that the distance for each attribute is the approximate range [0.1].

This distance function is good for continuous attributes but it makes a little sense for discrete attributes. For example, if we have an attribute that represents the color, that has values such as red, green, blue, brown, black, and white and these attributes are represented by integers 1 through 6 respectively, this representation will give a little sense.

2. Heterogeneous overlap – euclidean metric (HOEM)

One way to handle applications with both continuous and discrete attributes is to use heterogeneous distance functions on different kinds of attributes. One approach that has been used is to use overlap metric for discrete attributes and normalized Euclidean for continuous attributes.

Equation 2.2 shows the definition of this function for an attribute (a) with value (x ,y): -

$$d_a(x,y)=\begin{cases}1 & \text{if x or y is unknown, else}\\ Overlap(x,y) & \text{if a is a discreate, else}\\ rn\_diff_a(x,y)\end{cases} \quad \text{..........(2.2)}$$

If either of the attribute values is unknown, the distance function returns 1 (the maximum distance). Equations 2.3,2.4-show *overlap* and *rn_diff* functions respectively: -

$$\text{Overlap}(x\,,y)=\begin{cases}0 & \text{if } x=y\\ 1 & \text{Othewise}\end{cases} \quad \text{............(2.3)}$$

$$\text{rn\_diff}_a(x,y)=\frac{|x-y|}{\text{range}_a} \quad \text{............... (2.4)}$$

The value range$_a$ at equation 2.4 is used to normalize the attributes, and is defined as:

$$range_a=\max_a-\min_a \quad \text{.......... (2.5)}$$

Where max$_a$ and min$_a$ are the maximum and minimum values, respectively, observed in the training set for attribute (a).

Equation 2.2 returns a value for (d$_a$), which is in the range [0…1], whether the attribute is discrete or continuous. The overall distance between two input vectors x and y is given by the Heterogeneous Overlap-Euclidean Metric function HOEM (x, y)

$$HOEM(x,y)=\sqrt{\sum_{a=1}^{m} d_a(x_a,y_a)} \qquad ....(2.6)$$

m : the number of attributes.

3. Value difference metric (VDM)

This distance provides an appropriate distance function for nominal attributes. This function can be defined as:

$$vdm_a(x,y)=\sum_{c=1}^{C}\left|\frac{N_{a,x,c}}{N_{a,x}}-\frac{N_{a,y,c}}{N_{a,y}}\right|^q=\sum_{c=1}^{C}\left|P_{a,x,c}-p_{a,y,c}\right|^q \qquad ....(2.7)$$

Where

* $N_{a,x}$ is the number of instances in the training set T that has value $_x$ for attribute a;

* $N_{a,x,c}$ is the number of instances in T that has value x for attribute a and output class c;

* C is the number of output classes in the problem domain;

* q is a constant;

- $P_{a,x,c}$ is the conditional probability that the output class is (c) given that attribute (a) has the value (x) .
- $P_{a,x,c}$ is defined as :

$$P_{a,x,c}=\frac{N_{a,x,c}}{N_{a,x}}$$

Where $N_{a,x}$ is the sum of $N_{a,x,c}$ over all classes.

The sum of $P_{a,x,c}$ over all classes is 1 for fixed value of (a) .

$$N_{a,x} = \sum_{c=1}^{C} N_{a,x,c}$$

Using this distance measure, two values are considered to be closer if they have more similar classifications (more similar correlation with the output classes) regardless of what the values may be given.

One problem with the formulas presented above is that they do not define what should be done when a new value for an attribute appears in the test data where this value did not appear in the training data before. This problem makes $P_{a.x,c}$ undefined.

This distance function cannot be used directly for continuous attributes. Because most of the values of this type of attributes will be unique and the value for the attribute in the input vector will be also unique $P_{a,x,c}$ will be undefined. One way to resolve this problem is to discretize the continuous values of attributes into an arbitrary number of discrete ranges, and treat these ranges as nominal values. This way of treating continuous attribute has a disadvantage by treating the values in the same discretized range equally even if the values are on opposite ends of the range.

3. Heterogeneous value difference metric (HVDM)

As discussed before, the Euclidean function is inappropriate for nominal attributes, and VDM is inappropriate for continuous attributes, so neither of them is sufficient to be used on heterogeneous applications (application contains both continuous and nominal attributes).

HVDM is a heterogeneous distance function that returns the distance between two instances x and y. it is defined as follows:

$$HVDM(x,y)=\sqrt{\sum_{a=1}^{m} d_a(x_a,y_a)^2} \qquad ....(2.8)$$

m : the number of attributes. The function d_a (x, y) gives the distance between two input vectors and is defined as:

$$d_a(x,y)=\begin{cases}1\text{, if x or y is unknown; otherwise}\\ normalized\_vdm_a(x,y)\text{ if a is nominal}\\ normalized\_diff_a(x,y)\text{, if a is continuous}\end{cases} \qquad ....(2.9)$$

The *normalized_diff* is defined as follows:

$$normalized\_diff_a(x,y)=\frac{|x-y|}{4\sigma_a} \qquad ....(2.10)$$

Where σ_a is the standard deviation of the values of continuous attribute a. Dividing the function by 4 standard deviation comes from a statistical idea that 95% of the values in a normal distribution fall within two standard deviations of the mean.

The *normalized_vdm* can be defined as follows :

$$normalized\_vdm_a(x,y)=\sqrt{\sum_{c=1}^{C}\left|\frac{N_{a,x,c}}{N_{a,x}}-\frac{N_{a,y,c}}{N_{a,y}}\right|^2} \qquad ....(2.11)$$

Wilson proved by running (Euclidean, HOEM and HVDM) on more than 25 data sets that HVDM is the best in average between these distance function to find the distance between instances.

In this thesis, we do use HVDM as a distance function to find the distance between instances that will be discussed in the next chapter.

Table 2.1 summarizes the definition of each distance function.

All sunction use the same overall Distance function : $HOEM(x,y) = \sqrt{\sum d_a(x_a,y_a)^2}$

Distance Function	Definition for d_a (x,y) for each attribute type Continuous	 Discrete
Euclidean	$\frac{x_a - y_a}{\sigma_a}$	$\frac{x_a - y_a}{\sigma_a}$
HOEM	$\frac{\lvert x - y \rvert}{\text{range}_a}$	0 if $x_a = y_b$ and 1 x_a # y_b
HVDM	$\frac{x_a - y_a}{4\sigma_a}$	$\sqrt{vdm_a(x_x, y_a)}$
IVDM	$ivdem_a(x_a,y_a)$ interpolate probabilities from range midpoints	$\sqrt{vdm_a(x_x, y_a)}$

Where $range_a = Max_a - Min_a$, and

$Vdm_a(x,y) = \sum^{C} \lvert P_{a,x,c} - P_{a,y,c} \rvert^2$

Table 2.1: *Summary of distance function methods.*

2.7 Learning algorithms used In this thesis

To evaluate the performance of any feature selection method, it must be examined under one or more learning algorithms.

2.7.1 Decision tree learning algorithm

A decision tree can be defined as either a leaf (terminal node) that names a class or a decision node (non-terminal node, internal node) that specifies an attribute with a branch to another decision tree called subtree for each value of the attribute. The node that is linked by a branch from an internal node is called a child of the internal node. A path is a sequence of nodes from root (the first node in the tree) to a leaf. For binary –class problems, a leaf with a positive class is called a positive leaf while a leaf with negative class is called a negative leaf . The size of the decision tree is the number of nodes in it, including decision nodes as well as leaves.

During classification, the decision tree is built using the training data. To classify a new instance from test data the values of the instance are moved down from the root of the tree to a leaf. At each decision node, the test is evaluated on the example, and the example goes to the branch corresponding to the outcome of the test .When the new example reaches a leaf ,it is asserted to belong to the class labeled by the leaf.

Decision tree learning algorithms typically use the divide – and – conquer strategy. They build a decision tree by recursively dividing training set into subsets using splitting tests. A greedy divide – and – conquer approach can be briefly described as follows:

1-The leaf of the tree is labeled with the class of the majority of the examples at the current node when a stop criterion is satisfied.

2-Otherwise, an attribute is chosen with respect to some criterion and divides the training set into subsets, each corresponding to one possible outcome of the test. For this attribute it builds a subtree and repeats the same procedure on the remaining attributes until the final decision tree is built.

During the building of the decision tree, training sets, especially local training, might not be large enough at some nodes far from the root. In this case, the accuracy of test data may be very low. This is called overfitting (Kohavi, and Sommerfield,1995), there are two basic approaches to build a tree without this kind of damage to the accuracy:

First, by stopping the growth of the tree when just predefined stop criterion (or thresholds) together with the evaluation function used. This approach has a problem of selecting the good threshold, which will give the best tree.

The Second approach for solving the problem of overfitting is to leave the growing of the tree working using the simple stopping criteria without having a threshold until we get a large tree (raw tree) then applying a pruning technique to get the good size of the tree. Several pruning methods have been developed and used for decision tree learning, such as cost – complexity pruning (Breiman, et al.,1984), reduced error pruning (Quinlan, et al.,1987), and pessimistic pruning(Quinlan,1993).

ID3(Quinlan,1986) and C4.5(Quinlan,1993) are examples of TDIDT (Top Down Induction of Decision Tree) .They use information based on heuristic functions for test selection .C4.5 is the most powerful system which can handle problems with multiple classes and all types of attributes.

C4.5 Learning Algorithm

C4.5(Quinlan,1993) is an algorithm that builds a decision tree by employing a two-stage processes for growing and pruning . It produces a raw tree and a pruned tree. Figure 2.5 shows this algorithm.

```
  C4.5 Tree(Att, $D_{training}$ )

  INPUT :Att : a set of attributes,
             $D_{training}$ :a set of training examples represented using Att
             and classes.
  OUTPUT :two decision trees : a raw tree and a pruned tree.
  C:= The majority class in $D_{training}$
  Tree.raw := Grow-Tree(Att, $D_{training}$, C)
  Tree.Pruned :=Prune-Tree(Tree.raw, Att, $D_{training}$)
RETURN Tree.
```

Fig.2.5 *C4.5 Algorithm.*

The main idea is to grow a large tree and leave the overfitting problem to be solved by the pruning stage.

C4.5 grows a tree greedily. At each decision node, by examining all candidate attributes and all possible tests, it selects one test with high evaluation function value. Then it uses the same procedure recursively to generate a decision tree for the remaining subset. C4.5 may use one of the following evaluation functions: information gain function or gain ratio function. Information gain is a function that determines which attribute will give us the most information in the data set (Quinlan,1993). Suppose that there is a training set in which there are (P) examples of class (c1) and (n) examples of class (c2), then the information needed to return a class for (a) particular example can be defied as :

$$I(p,n) = -\frac{p}{p+n}\log 2\frac{p}{p+n} - \frac{n}{p+n}\log 2\frac{n}{p+n} \quad ....(2.12)$$

If we suppose that the attribute (A) can have any value from the set {a1,a2,a3,…,av} and there are (p_I) examples with attribute value (a_I) and (n_I) examples with attribute value (a_I) belonging to class (c2) . The expected information necessary for testing attribute (A) as the root of the tree is defined as :

$$H(A) = \sum_{i=1}^{v} \frac{p_{i+} + n_i}{p+n} I(p_i, n_i) \quad ......(2.13)$$

Then the information gained by using attribute (A) as the root of the decision tree is:

Gain(A) =I(p,n) – H(A) .

A problem of this function is that it selects an irrelevant attribute as a root of a tree if this attribute has a lot of values in the data set. One way to solve this problem is by grouping the attribute values into two sets and considers the attribute as a binary one.

Gain ratio (Quilan,1993) is a function to deal with a multi-value problems , the main idea over information gain is that the attribute itself has some information (irrespective of the class) and defined as :

$$IV(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} \log 2 \frac{p_+ + n_i}{p + n} \qquad \text{.............(2.13)}$$

The new measure is defined as:

$$\frac{gain(A)}{IV(A)} \text{.......(2.15)}$$

Which is called the gain ratio.

The second process in C4.5 algorithm is to prune the raw tree to get the final pruned tree to solve the problem of overfitting. (Quinlan,1993) developed a technique that used in C4.5 algorithm; in this technique, the error rate is obtained at a leaf by dividing the number of mis-classifications by the number of examples covered by the leaf. Then the number of predicted errors at a node is simply the error rate times the number of examples at that node. The error of a sub tree is the sum of the errors of their branches.

A sub-tree is pruned if the predicted error of its node is smaller than the predicted error of the sub tree.

2.7.2 Instance-Based learning algorithm

The nearest neighbor algorithm is a learning algorithm that stores the training instances during learning. During execution, the new input vector (instance) is compared with each instance in the training set .The class of the instance that is most close to the input instance - using distance function discussed before - is the predicted output class.

The nearest neighbor algorithm has some advantages when compared with other learning algorithm :

1- It learns very quickly O(n) for n instances during learning.

2- It has no problem if there is no training instance similar to the input instance, because it gets the most closest instance for the input instance.

3- Easy to implement and understand.

Like any learning algorithm it has some drawbacks:

1- The selected distance function has a high influence on the learning algorithm that appears in the output accuracy.

2- It has a large storage requirement because it stores all the training instances.

3- It is slow during execution because it needs to search for each input instance in all the training data to get the closest instance.

4- Its accuracy degrades with existence of noise data in the training data.

Kia, (1995), developed an extensions to the basic nearest neighbor algorithm and called it Instance - based learning algorithms.

The main idea of the development is to select some of the training data during the learning phase and make the new algorithm have less affect because of noise data in the data set.

IBL2 is the first extension by reducing the size of the training data during the learning by forcing a rule to select the instances, such as Condensed Nearest Neighbor Rule (CNN). This extension still suffers from the noise data in the data set. IBL3 (Aha et al. , 1991) addresses IBL2's problem of keeping noisy instances by using statistical test to retain only the acceptable miss-classified instances. (Aha et al.,1991) proposed that IBL3 was able to achieve grater reduction in the number of stored instances and also achieved higher accuracy than IBL2 on applications which were tested.

The last and best extension for IBL algorithms is by selecting a number of nearest instances for the input instance and take the average weight for the values of these instances and then find the max class weight from the classes of the selected instances to be the predicted class.

The last extension of IBL algorithm is the one we used in our work in this thesis.

2.7.3 Naïve Bayesian classifier

The Bayesian Classifier (Duba, and Hart ,1973) is a probabilistic approach for classification. It can give the probability that an example (J) belongs to a class (Ci) given the values of attributes at an example. This classifier uses Bayes rule to compute the probability of each class given the instance, assuming that the features are conditionally independent given the label. To classify a new instance ,naïve bayesian classifier applies Bayes rule at each description .

$$P(C_i \mid I) = \frac{P(C_i)P(I \mid C_i)}{P(I)} \text{......(2.16)}$$

However, since (I) is a conjunction of (j) values, one can expand the equation 2.16 to

$$\mathrm{P}(C_i \wedge V_i) = \frac{P(C_i)P(\wedge V_j \mid C_i)}{\sum_{\mathrm{j}} P(V_j \mid V_k)p(C_k)} \qquad \text{.........(2.17)}$$

Where the denominator sums over all classes and P(∧Vi|Ci) is the probability of the instance (I) given tha class Ci .After calculating these quantities for each description, the algorithm assigns the instance to the class with the highest probability .

To compute P($\wedge V_i|C_i$) ,Naïve bayesian classifier assumes the independence of attributes within each class , which lets it use the equation

$$P(\wedge V_i \mid C_k) = \prod_i P(V_i \mid C_k) \qquad ....(2.18)$$

Where P(V_i| C_k) is the pre evaluated provability of each class from train data .

The probability for nominal attributes are estimated from data . The probabilities for continuous attributes are estimated from the Gaussian Distribution for these attributes, where the mean and the standard deviation are estimated from the data.

Chapter 3 : Combined Approach for Feature Selection

3.1 Introduction

Feature selection can be defined as a method for finding a minimum set of M relevant features that describe the data set as well as the original set of features(Langley, and Blum ,1992).

Feature selection methods can be divided into three categories (approaches):

1) Filtering Approach: Techniques of this category filter out the relevant attributes before using the induction algorithm (A preprocessing step).

2)Wrapper Approach: Techniques of this category are wrapped around the induction algorithm.

3)Embedded Approach: Techniques of this category are integrated as a part of the induction algorithm

The above three approaches were discussed in detail in chapter 2

3.2 Combined approach to do feature selection

Selecting the relevant features from a large number of features that describe a data set was previously done by one of the discussed approaches in separate. We developed a new technique to select relevant attributes by taking an initial set of relevant attributes using a filter approach and then apply the wrapper approach in two different ways: Forward selection on the remaining attributes and Backward Elimination on the initial set of attributes around three well known learning algorithms; Bayesian algorithm , Instance based algorithm and Decision tree learning algorithm .The main diagram of this improved approach is shown on Fig 3.1.

1- Choosing The Initial Set of Attributes

To get the initial set of attributes, we estimate a weight for each attribute that indicates how good the attribute is by using RELIEFF algorithm developed by (kononenko,1994) which was discussed in chapter 2 of this thesis .The key idea of RELIEFF algorithm is to estimate attributes according to how well their values distinguish among cases that are near each other.

The second step of choosing the initial step is to use the weight of attributes to get the best initial set. But how ? To do this we tried three alternative mechanisms; They are :-

Mechanism 1: Choose attributes with positive weights, i.e., weight above zero.

Mechanism 2: Choose attributes with weight above average weight , where the average weight is the sum of weights of all attribute divided by the number of attributes.

Mechanism 3: Choose attributes with weight above median, where the median is the weight of attribute which is in the middle after sorting the attributes according to their weights.

```
begin

• Input   A          : All attributes in the data set.
          D          :Train Data .
          Forward    : Forward Selection .
          Backward   :Backward Elimination.
          S1         :Initial set of attributes.
          S2         :The remained attributes from the first phase.
• Output
          S3         :The final set of relevant attributes.
          Err2       :The final error on test data .

Combined algorithm.
1.  Set all weights of attributes to zero.
2.  Evaluate the weight of attributes using RELIEFF algorithm.
                                  [See Fig 2.2]

3.  Get the initial set of attributes.
                          ┌─  Att. Above Zero.
                S1 =      ├─  Att. Above average.
                          └─  Att. Above median.
4.  C1 = num of attributes in S1.

5.  S2 = All_atributes – S1.

6.  If Forward Do Forward Selection.
                                  [See fig(3.2)]

7.  If Backward Do backward Elimination.
                                  [See fig (3.3)]
8.  Err2 =Learn Alg. (S3,Dtest) .
```

Fig 3.1:- *The new combined algorithm.*

2- Selecting More Features: - (Forward Selection)

In forward selection fig 3.2 we start with the set of attributes that are above median (3 rd mechanism) and start adding new feature at each step and evaluate the accuracy until all the remaining attributes are included, or when adding a new feature does not improve the accuracy of the attribute set under an evaluation algorithm . The evaluation process is done on the training data of the data set where the final accuracy is evaluated on the test data.

- **Input** A : All attributes in the data set.

  S1 :Initial set of attributes.
  S2 :The remained attributes from the first phase.
- **Output**
  S3 :The final set of relevant attributes.
  Err :The final error on train data .

Forward Selection

```
Err1=0;
Err=0;
Repeat

   K = K+ 1

   S3 = S1 + K

    Err1 = Learn Alg. (S3 ,Dtrain).
     If   err1<err then
                    Begin
                          S3 is the final set of attributes.
                          Err=Err1.
                          K=K+1
                    End.
  9.  Until  Err is not  decrease .
```

Fig 3.2:- *Forward selection .*

3- Eliminating Features: - (Backward Elimination)

In backward elimination, we start with attributes with weight above zero and start remove one the attribute with the lowest weight at as long as this improves the accuracy under the evaluation algorithm. As in forward selection, evaluation of accuracy is done on the training data of the data set where the final accuracy is evaluated on (unseen) test data Fig 3.3.

The results and their discussion will be discussed later in this chapter.

```
• Input   A      : All attributes in the data set.

          S1     :Initial set of attributes.
          S2      :The remained attributes from the first phase.
• Output
          S3      :The final set of relevant attributes.
          Err    :The final error on train data .
  Backward Elimination.

Err1=0;
Err=0;
Repeat
   K =  K+ 1
   S3 = S1 -  K
    Err1 = Learn Alg. (S3 ,Dtrain).
     If    err1<err then
                    Begin
                          S3 is the final set of attributes.
                          Err=Err1.
                          K=K+1
                    End.
10. Until  Err is not   decrease .
```

Fig 3.3:- *Backward elimination.*

Both Forward selection and Backward elimination give a final set of relevant attributes which were evaluated on unseen (test) data under three common learning algorithms, Bayesian , Instance based learning and Decision tree algorithms .

3.3 . Empirical evaluation

To test if this method will improve the accuracy and reduce the error rate several experiments were performed . Within machine learning, the standard experimental method (Kibler, and Langley, 1988) involves running a learning algorithm on a set of training data and then using the new approach to make prediction about separate test cases and then measure the accuracy .To evaluate the accuracy of our algorithm in classifying unseen cases, we carried out an experimental test using three learning algorithms, Bayesian classifier, Instance based classifier and Decision tree learning algorithm.

Cross – Validation: -

For all the data sets a 10-folds cross validation process is used to obtain the estimated accuracy on data sets. This method divides the data set into ten sets of training data as defined by (Kohavi,1995). Ten experiments were performed, each used 9 different sets as a training set and the tenth set as a test set.

To support our results of the approach, we apply a statistical method on the results; it is the two - tailed t-test which give us if these results logically increase or decrease or they come by chance .

3.4.1 Description of data sets

In our experiments, we have used nine-data sets from the UCI repository(Murphy, and Aha,1994) and an artificial data set (INF2). Table 3.1 summarizes the number of cases, the number of classes and the number of nominal and numeric attributes in each data set.

INF2 :- The artificial data set contains nine binary attributes and three classes. We set five of these attributes to be relevant attributes that affect the class value. The main idea of this data set is that the attributes are strongly dependent by making an XOR relation between pairs of attributes.

3.4.2 Experimental methods

In the following experiments the new approach is evaluated under three learning algorithms; Bayeisian algorithm, instance-based algorithm and Decision tree learning algorithm.

The mark ++ indicates that the new approach is statistically better accuracy, * means that the new approach is statistically worse.

Tables 3.2, 3.3, 3.4 show the results of applying the new approach under the Instance Based algorithm. Table 3.2 shows the results of applying the new approach using forward selection. It gives good results in five of the data set and degrades the accuracy in three data sets, and the accuracy remains the same in two data sets. Table 3.3 shows the results of applying the new approach using backward elimination. It gives good results in six of the data sets and degrades the accuracy in two data sets, and the accuracy remains the same in two data sets.

We can see that the main improvement in the accuracy for the data set is in the artificial (INF2) data set. Because our new approach deals well with data sets that have strong dependencies between attributes. Another thing should be noticed in (INF2) data set is the number of attributes chosen in forward selection and backward elimination is five. These attributes are the relevant attributes known to us before we apply this approach, which means that the new approach is good for selecting the relevant attributes in the data set.

Tables 3.5, 3.6, 3.7 show the results from applying the new approach under the Bayesian algorithm. Table 3.6 shows the results of applying the new approach using forward selection. It gives good results in six of the data sets and degrades the accuracy in two data sets, and the accuracy remains the same in two data sets. Table 3.7 shows the results of applying the new approach using backward elimination. It gives good results in six of the data sets and degrades the accuracy in two data sets, and the accuracy remains the same in two data sets.

Tables 3.8, 3.9, 3.10 are the results of applying RELIEFF as a selection criterion in a decision tree algorithm as done by Kononenko and Simec (1994) instead of the gain ratio and then apply the forward selection and backward elimination . The results of the new approach are compared with results of applying the data sets used in our thesis using Gain Ratio as a selection criteria. The tables show that applying RELIEFF is better than using Gain Ratio as a selection criteria for data sets with strong dependencies between attributes as in (INF2) data set, Applying the Wrapper approach using Forward selection or Backward elimination did not improve the accuracy of the data set because the decision tree learning algorithm contains an embedded feature selection .

Table 3.1 : *Description of data sets.*

No.	Name	No. of Cases	No. of Classes	Attribute	
				Con.	Discr.
1	Breast Cancer	699	2	9	---
2	Standardized Audiology	200	23	----	69
3	Wine	178	3	13	---
4	Solar	216	6	3	9
5	Vote	436	2	17	______
6	Zoo	101	7	17	--
7	Inf2	200	3	----	9
8	Dermatology	366	6	1	33
9	Crx	620	2	6	9
10	Iris Plants	150	3	4	----

Table 3.2: *Final accuracy for initial set of attributes under instance based algorithm*

Name	Above Zero			Above Median			Above Average.			
	Train. Data %	Test data %	Avg no of att	Train. Data %	Test data %	Avg no of att	Train. Data %	Test data %	Avg no of att	Total No. Of Att
Breast Cancer	96.93	95.84	7	95.63	94.12	5	94.82	92.84	4	9
Standardized Audiology	78.06	64.00	41	78.61	62.00	35	78.72	62.00	18	69
Wine	98.68	97.5	11	98.68	96.67	6	98.68	95.83	6	13
Solar	79.99	69.11	8	78.28	75.63	5	76.11	71.41	2	12
Vote	96.70	95.62	7	96.50	95.85	8	96.14	95.85	4	16
Zoo	99.21	91.00	16	98.09	5.00	8	98.43	83.00	7	17
Inf2	87.00	68.57	5	72.67	59.52	4	74.06	57.62	4	9
Dermatology	99.21	98.63	32	95.48	93.45	17	90.47	89.04	14	34
Crx	85.83	85.51	2	88.08	86.39	7	89.63	85.10	11	15
Iris Plants	96.22	93.33	4	96.96	96.00	2	96.96	96.00	2	4

Table 3.3:- *Final Accuracy for our combined algorithm under instance baesd algorithm (IBL) (Forward selection).*

	Flittering approach Above Median		Forward Wrapper Approach.		Base case	
Name	Accuracy %	Avg No of Att	Accuracy %	#att	Accuracy %	#att
Breast cancer	94.12	5	95.84 ++	7	95.70	9
Standardized Audiology	62.00	35	62.00 *	35	64.00	69
Wine	96.67	6	96.67 ++	6	95.76	13
Solar	75.63	5	68.18	11	68.64	12
Vote	95.85	8	95.85 ++	10	95.37	16
Zoo	85.00	8	86.00 *	10	91.00	17
Inf2	59.52	4	63.33++	5	35.24	9
Dermatology	93.45	17	98.63	25	98.63	34
Crx	86.39	7	86.26 *	10	86.53	15
Iris Plants	96.00	2	96.00 ++	2	93.33	4

Table 3.4:- *Final Accuracy for our combined algorithm under instance baesd algorithm (IBL) (Backward elimination).*

	Flittering approach Above Zero		Backward Wrapper Approach.		Base case	
Name	Accuracy %	Avg no of att	Accuracy %	#att	Accuracy %	#att
Breast cancer	95.84	7	95.84 ++	7	95.70	9
Standardized Audiology	64.00	41	64.00	43	64.00	69
Wine	97.5	11	97.5 *	12	95.76	13
Solar	69.11	8	69.11 ++	8	68.64	12
Vote	95.62	7	95.62 ++	6	95.37	16
Zoo	91.00	16	91.00	17	91.00	17
Inf2	68.57	5	68.57 ++	5	35.24	9
Dermatology	98.63	32	98.91 ++	31	98.63	34
Crx	85.51	2	85.51 *	2	86.53	15
Iris Plants	93.33	4	95.33 ++	2	93.33	4

Table 3.5:- *Final accuracy for initial set of attributes under Bayesian classifier.*

Name	Above Zero			Above Median			Above Average.			
	Train. Data %	Test data %	Avg No of Att	Train. Data %	Test Data %	Avg no of att	Train. Data %	Test data %	Avg No of att	Total No. Of Att
Breast cancer	83.3	84.3	7	84.4	85.2	5	83.0	82.6	4	9
Standardized Audiology	96.4	71.0	41	96.1	69.5	35	91.6	69.0	18	69
Wine	91.0	91.7	11	80.2	79.2	6	76.9	78.3	6	13
Solar	73.2	69.1	8	73.1	69.2	5	75.3	75.9	2	12
Vote	94.0	93.8	7	94.0	94.1	8	95.6	94.6	5	16
Zoo	100.0	95.00	16	100.0	92	8	100.0	91.0	7	17
Inf2	46.9	39.5	5	44.0	41.4	4	45.1	41.9	4	9
Dermatology	99.3	95.6	32	96.7	94.0	17	91.0	88.9	14	34
Crx	85.7	85.3	2	81.1	79.4	7	69.4	67.5	11	15
Iris Plants	80.0	77.3	4	49.3	40.7	2	49.3	40.7	2	4

Table 3.6:- *Final Accuracy for our combined algorithm under Bayesian classifier (BCR) (Forward selection).*

	Flittering approach Above Median		Forward Wrapper Approach.		Base case	
Name	Accuracy %	Avg No of Att	Accuracy %	#att	Accuracy %	#att
Breast cancer	85.2	5	84.5 ++	6	83.2	9
Standardized Audiology	69.5	35	74.0	69	74.0	69
Wine	79.2	6	90.8	13	90.8	13
Solar	69.2	5	67.3 *	5	71.8	12
Vote	94.1	8	94.8 ++	8	89.2	16
Zoo	92	8	95.00 ++	16	94.0	17
Inf2	41.4	4	41.0 ++	8	39.5	9
Dermatology	94.0	17	98.6 ++	27	98.3	34
Crx	79.4	7	77.9 ++	8	67.1	15
Iris Plants	40.7	2	72.0 *	3	77.3	4

Table 3.7:- *Final accuracy for initial set of attributes under Bayesian classifier (BCR) (Backward elimination).*

	Flittering approach Above Zero		Backward Wrapper Approach.		Base case	
Name	Accuracy %	Avg No of Att	Accuracy %	#att	Accuracy %	#att
Breast cancer	84.3	7	84.8 ++	5	83.2	9
Standardized Audiology	71.0	41	71.00 *	41	74.0	69
Wine	91.7	11	91.7 ++	10	90.8	13
Solar	69.1	8	69.1 *	8	71.8	12
Vote	93.8	7	94.6 ++	4	89.2	16
Zoo	95.00	16	95.0 ++	16	94.0	17
Inf2	39.5	5	39.5	5	39.5	9
Dermatology	95.6	32	98.6 ++	32	98.3	34
Crx	85.3	2	85.3++	2	67.1	15
Iris Plants	77.3	4	77.3	4	77.3	4

Table 3.8 :- *Final accuracy For initial set of attributes under decision tree algorithm*

	Above Median		Above Zero		Base Case Using RELIEFF	
Name	Accuracy Rate %	Size of Tree	Accuracy Rate %	Size of tree	Accuracy Rate %	Size of tree
Breast cancer	95.29	13	94.42	13	95.29	13
Standardized Audiology	68.5	45	68.5	45	68.5	45
Wine	78.03	7	78.03	7	78.03	7
Solar	70.95	29	70.36	29	72.36	33
Vote	93.29	11	93.39	11	9329	11
Zoo	91.00	13	93.00	16	93.00	16
Inf2	69.05	39	96.67	62	96.67	62
Dermatology	84.44	18	85.6	18	85.60	18
Crx	85.95	15	85.22	3	85.96	28
Iris Planet	94.00	7	94.00	7	94.00	7

Table 3.9:- *Final Accuracy for our combined algorithm using decision tree algorithm (Forward selection).*

	Above Median		Forward Selection		Base Case using Gain Ratio	
Name	**Accuracy Rate %**	**Size of Tree**	**Accuracy Rate %**	**Size of tree**	**Accuracy Rate %**	**Size of Tree**
Breast cancer	95.29	13	95.29	13	93.41	19
Standardized Audiology	68.5	45	68.5	45	70.00	45
Wine	78.03	7	78.03	7	90.42	9
Solar	70.95	29	72.36	33	71.43	26
Vote	93.29	11	93.29	11	91.45	19
Zoo	91.00	13	93.00	16	97.00	13
Inf2	69.05	39	96.67	62	82.86	59
Dermatology	84.44	18	85.60	18	92.92	36
Crx	85.95	15	85.96	18	84.06	57
Iris Plants	94.00	7	94.00	7	94.00	7

Table 3.10 :- *Final Accuracy for our combined algorithm under decision tree algorithm (Backward elimination).*

	Above Zero		Backward Elimination		Base Case using RELIEFF	
Name	Accuracy Rate %	Size of Tree	Accuracy Rate %	Size of tree	Accuracy Rate %	Size of tree
Breast cancer	94.42	13	93.28	61	93.41	19
Standardized Audiology	68.5	45	68.50	86	70.00	45
Wine	78.03	7	74.55	17	90.42	9
Solar	70.36	29	67.73	44	71.43	26
Vote	93.39	11	92.16	17	91.45	19
Zoo	93.00	16	93.00	17	97.00	13
Inf2	96.67	62	96.67	62	82.86	59
Dermatology	85.6	18	86.91	97	92.92	36
Crx	85.22	3	85.22	3	84.06	57
Iris Plants	94.00	7	94.67	12	94.00	7

Finally, the results induced from the new algorithm are more accurate for the most of the data sets on both of the learning algorithms and using RELIEFF as a selection criteria is better than using Gain Ratio for decision tree learning algorithms.

As a result, the main characteristic of this algorithm is to get the starting set of attributes filtered out from the first part of our algorithm (i.e. the attributes with weight above median). The search control is an ordered search according to weight of each attribute and finally the halt criterion is when the first decrease of accuracy occurred.

Chapter 4: Constructive Induction on Continuous Classes

4.1 Introduction

Constructive induction can be defined as the process of changing the representation of the data set by creating new features from existing attributes(Mark, and Jude,1993). The existing attributes are called the primitive attributes. (Michalski, and Larson,1978) is behind the idea of constructive induction. Since that time many constructive induction algorithms have been developed. Constructive induction is done to improve the accuracy of the learned concept. Most constructive induction algorithms generate new Boolean attributes.

Creating new attributes in constructive induction is done by using some operators. The most common operators used in constructive induction are ***AND and OR*** operations which will result in a new Boolean attribute.

The need for constructive induction appears because in some real – world problems the attributes that describe these problems are not appropriate for describing the class, which makes the classification accuracy very poor and makes the understanding of the classifier hard. Another problem that may appear at decision tree is the replication problem (Pagallo, and Haussler , 1988). This problem is defined as having a duplication of sequence of tests in deferent paths that will lead to produce a tree with low prediction accuracy. Constructive induction methods attempt to solve these problems.

We will discuss a method for constructing new attributes on data sets with continuous classes.

4.2 Strategies for constructive induction

As we mentioned above most constructive induction algorithms use conjunction and/or disjunction as a constructive operator to generate new attributes. In terms of new attribute constructive strategies, almost all these algorithms use either the Hypothesis – Driven strategy or Data – Driven strategy. The following two sub sections discuss the two strategies in more details.

4.2.1 Hypothesis – driven strategy

Algorithms of this strategy construct new attributes by conjunction or disjunction, using the fixed path-based strategy (Michalski, and Wenk,1991). They interleave two processes: - Building a decision tree and constructing of new attributes. Each iteration constructs by Conjunction or Disjunction of only two primitive attributes.

Fringe (Pagello, and Haussler, 1989) is an example of this strategy of constructive induction. At this algorithm, the new attributes are conjunction for each pair of conditions at the parent and the grandparent nodes of a positive leaf. This algorithm is for two class problems.

This algorithm has been modified several times to deal with more than two class problems Sfring.

4.2.2 Data – sriven strategy

At this strategy the algorithm constructs new attributes directly based on training data when building a decision tree(. The algorithms under this category have only one iteration, so only one tree is built at each run. During the building of decision tree, the new attributes generated and used as primitive attributes to build the decision tree.

LFC(Rendell, and Ragavan,1993) algorithm uses two constructive operators (Conjunction and Negation) and uses information gain as an evaluation function for building the decision tree. This algorithm can work on Boolean , multi valued and continuous attributes. According to (Rendell,and Ragavan,1993), this algorithm performs well and gives high accuracy's for real – world data sets.

The above two strategies do constructive induction on data sets with discrete classes.

4.3 Constructive induction on continuous classes

In this section we will discuss a method for constructing new attributes from a set of primitive attributes describing data set with continuous classes. This method is part of our work in this thesis. The main characteristics for this method are:

1) The operators used to construct new attributes are AND *and* OR operators.
2) The Primitive attributes used to construct new attributes are Boolean attributes.
3) The number of new constructive attributes is half the number of primitive attributes.
4) The criterion used to choose a new constructive attribute is based on standard deviation. We prefer the attribute with lowest.
5) The new constructive attributes are the conjunction or dis-conjunction of only two primitive attributes.
6) The new constructive attribute is accepted if its Standard deviation is less than the standard deviation of the two primitive attributes used to construct this attribute.

7) This method is a Data – driven constructive induction method, and works as a preprocessing method meaning that the new constructive attributes are generated before going into the induction algorithm.

8) This method is a one-phase method.

Because we use the Boolean primitive attributes that have the values of 0 or 1, we calculate a weighted STD of 0 values of the attribute and calculate the weighted STD of 1 values of the new attribute. Figure 4.1 shows the idea of finding the Weighted STD of the new constructive attributes. After finding the STD of new constructive attributes, we choose the N/2 of these attributes that have the lowest STD. Where N is the number of primitive attributes. Then we add the new attributes to the primitive attributes and then we evaluate the accuracy using one of the learning algorithms that will be discussed later. Then we compare the result with the base case without using the constructive method. Figure 4.2 shows the details of this method.

```
P          No Of Primitive Attributes In The Data Set.
B          No Of Boolean Primitive Attributes.
N          No Of Cases In The Data Set.
N0         No Of Cases For New Attribute With Value 0.
N1         No Of Cases For New Attribute With Value 0.
Mean(0)    Mean value Of Value 0 for New Constructive Attribute .
Mean(0)    Mean value Of Value 0 for New Constructive Attribute .
STD(0)     Standard Deviation Of Value 0 for New Constructive Attribute
STD(1)     Standard Deviation Of Value 1 for New Constructive Attribute
STD        Final Standard Deviation  for New Constructive Attribute.
Output     Standard Deviation Of All Boolean Attributes.
(1)        For Each (B) Do
(2)        begin
(3)         Find    STD(A0).
                $STD(0) = \sqrt[2]{\frac{1}{N0}\sum_{i=1}^{N}(Class(item[i]) - Mean0)}$
                For all cases with value 0 for new attribute.
(4)         Find STD(1)
                $STD(1) = \sqrt[2]{\frac{1}{N1}\sum_{i=1}^{N}(Class(item[i]) - Mean1)}$
                For all cases with value 1 for new attribute.
(5)        STD=(N0 * STD(0) + N1 * STD(1) )/N
(6)End
```

Fig.4.1 *Weighted standard deviation for new constructive attributes.*

```
P           No Of Primitive Attributes In The Data Set.
F           Final Set Of Attributes After Constructive
C           The Constructive Attributes Selected
Err         Final Error Rate After Constructive.
Begin
♦     Get STD of all possible New attributes. (Fig. 4.1)
♦     Sort STD for all new attributes.
♦     Select P/2 of the smallest STD of New attributes.
♦     F = P + C
♦     Evaluate The error rate using the induction algorithms
♦     Err = Evaluate (Dtest,F).
End.
```

Fig 4.2. *The Constructive induction method on continuous classes*

4.4 Empirical evaluation

To test if the method improves the accuracy of a data set under certain learning algorithms, we performed an empirical evaluation for this new method. Within machine learning, the standard experimental method (Kibler, and Langley, 1988) involves running a learning algorithm on a set of training data then making a prediction on test cases and getting the accuracy. To evaluate the accuracy of this method, we run it under two learning algorithms; Regression tree and Regression Rule. We use 5 data sets.

4.4.1 The algorithms used to evaluate the method.

In our experiments, we use two learning algorithms to evaluate the performance of our method, these two algorithms are:

1) Regression rule learning algorithm

This algorithm constructs rules using specific- to – general search direction, each rule consists of a conjunction of antecedents and a liner model. Each rule in the training set assigns a rule and constructs a liner model for that rule using the case only. Then for each rule this induction algorithm finds the nearest case with a target value near to this rule, and it must not be covered by another rule.

The final results by applying the data sets under these two induction algorithms will be discussed at experimental study section discussed later in this chapter.

2) Regression tree learning algorithm

This algorithm has developed by (Quinlan,1992). This algorithm carries out a greedy search through the space of possible trees to build a piecewise liner model in the form of model tree. The idea is to split the training cases in such a way as growing a decision tree .The decision tree learning algorithm uses an evaluation function to select the attribute that has the best ability to minimize intra – subset of class values rather than maximizing the information gain. A leaf of a model tree contains a linear model relating the class values of the training cases to their attribute values rather than a class name as in decision trees.

4.4.2 Description of data sets.

During our experiments, we have used artificial and natural data sets. Natural data sets were brought from the UCI repository. We have used several families of artificial data sets to check the behavior of our method.

Module – 8 : each domain contains a set of attributes, value of each attribute is an integer value 0 or 1. Half of the attributes are continuous and the second half are boolean attributes. The class value is a random number in the range 0-1.In our experiments we used two types of t this data set

Parity : Each domain consists of discrete and boolean attributes. The I informative attributes define parity problem. The class value is :

$$C = \begin{Bmatrix} \text{rand}(0,0.5) :; C = (\sum_{j=1}^{I} Aj) \bmod 2 = 0 \\ \text{rand}(0.5,1) :; C = (\sum_{j=1}^{I} Aj) \bmod 2 = 1 \end{Bmatrix}$$

In our experiments we used two types of this kind of data sets.

Hepatitis : This data set is a real world data set from the UCI repository, it consists of five continuous attributes and thirteen boolean attributes. The class value is one of the continuous attributes.

4.4.3 Experimental method

The above learning algorithms were used to measure the performance of the new method; Regression Rule and regression Tree learning algorithms. The measure is the average error magnitude (or residual) on unseen cases.

4.4.4 Results

We run the new method under two learning algorithms. The results are summarized in tables 4.1 and 4.2. Each entry in the table contains the residual error before doing constructive induction, residual error after doing the constructive induction and the number of new attributes that added to the primitive attributes. The results show that the new method gives better accuracy in three-data sets and degrades in only one and remains the same in one under the two learning algorithms.

Table 4.1 :- *Residual error rate using regression tree algorithm.*

Data Sets	Residual Error Rate Before Constructive	Residual Error Rate After Constructive	No. Of New Attributes
Modu (And)	8.96	10.125	2
Modu (Or)	8.96	7.76	2
Par (And)	12.52	12.37	2
Par (Or)	12.52	12.52	2
Hepatitis (And/Or)	5.94	5.06	3

Table 4.1 :- *Residual error rate using regression rule algorithm.*

Data Sets	Residual Error Rate Before Constructive	Residual Error Rate After Constructive	No. Of New Attributes
Modu (And)	8.96	10.125	2
Modu (Or)	8.96	7.76	2
Par (And)	12.4	12.00	2
Par (Or)	12.4	12.30	2
Hepatitis (And/Or)	6.10	5.25	3

In conclusion, doing constructive induction on continuous classes give good results.

Chapter 5 : Conclusion and Future Work

5.1 Conclusion

Machine learning has become an important area of study within the field of artificial intelligence. Among the wide variety of issues in the area of machine learning, the issue of selecting the most relevant features is extremely important. It affects not only the prediction in accuracy but also the comprehensibility of the produced classifier.

In the first part of this thesis we developed a new method to do feature selection that is based the -RELIEF algorithm which was originally used to weight features according to their relevancy.

We performed empirical evaluation on three well-known learning algorithms: Instance -based, bayesian classifier and decision tree. We used 10 different data sets in our evaluation.

Empirical evaluation using instance based learning algorithm shows that the new method improves the accuracy on five-data sets and degrades in three-data sets and remains the same in two-data sets for forward selection. For backward elimination, the new method improves the accuracy on six-data sets and degrades in two-data sets and remains the same in two-data sets.

Empirical evaluation using the Bayeisan classifier shows that the new method improves the accuracy on six-data sets and degrades in two-data sets and remains the same in two-data sets for forward selection. For backward elimination, the new method improves the accuracy on six-data sets and degrades in two-data sets and is remains the same in two-data sets.

Empirical evaluation using decision tree learning algorithm shows that using RELIEFF as an evaluation measure instead of gain ration gives good accuracy especially for data sets with strong dependencies between attributes. The advantage of new method is most obvious with data sets that contain a high dependency between features.

The second part of this thesis is concerned with learning tasks involve continuous rather than discrete classes. In this thesis, we have applied and evaluated a constructive induction method on problems with continuous classes.

The results in chapter 4 show that this method improve the accuracy on three-data sets and degrades the accuracy in one and is still the same in one. The experiments were performed using two machine learning algorithms; Regression tree and regression rules. The new representation of data set after constructive induction remains simple because the new method constructs a small number of new features.

5.2 Future work

The area of feature selection area still has more room for future work. One direction is to use other search methods (other than forward and backward elimination) like stepwise forward and backward elimination that allows us to add/remove an attribute that was removed or added before. Another direction for future work is to use anther attribute weighting algorithm instead of RELIEFF.

Constructive induction on continuous classes is still a new area of work, so there is a lot to be done in this area. One can develop a method to do constructive induction on different types of attributes, not only can boolean attributes be used, but also nominal and discrete attributes can be used. Another direction for future work is to use another operators such as ***XOR, NOT*** to construct new attributes.

Finally, one can develop a new method to do feature selection on data set after or before applying the constructive induction method.

References

Aha, D. W. , Kibler, D. and Albert, M. K. 1991. *Instance – based learning algorithms.* Machine Learning, 6, pp. 37 – 66.

Almuallim, H. and Detterich, T. G. 1992 . *Efficient Algorithms for Identifying Relevant Features*. Proceedings of the Ninth Canadian Conference on Artificial Intelligence , PP. 38 – 45 .Morgan Kaufmann.

Barr, A. and Feigenbum, E. A. 1982. *The Handbook Artificial Intelligence*. Morgan Kaufmann , Los Altos, California.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. 1984. *Classification and Regression Trees*. Belmont,CA: Wadsworth.

Caruana, R. and Fring, D. 1994.*Greedy Attribute Selection.* Cohen,W. and Hirsh ,H. ,(eds),Machine Learning :Proceeding of the Eleventh International Conference .Morgan Kaufmann.

Mark, W. C. and Jude, W. S. 1993.*Learning to represent condons :A Challenge for Constructive Induction* . Proceeding of the 13th international joint conference on artificial intelligence .

Dietterich, T. G. and Almuallim, H. 1991.*Learning with many Irrelevant Features*. The Ninth National Conference on Artificial Intelligence , PP. 247 – 552 .MIT Press.

Duba, R. O. and Hart, P. E. 1973. *Pattern Classification and system analysis*. New York, NY: Wilsey.

Hu, Y. and Kibler, D. 1996 .*Generation of Attributes for Learning Algorithms.* proceeding of the 13th International conference on artificial intelligence, PP. 806 – 811 .

Issa, R. ,1998.*Knowledge Discovery Using a Devaloped Learning Algorithm.* MS thesis. Jordan university 1998

Kia, M. T. 1995.*Common Issues in Instance Based and Naïve Bayesian Classifier* .A PHD Thesis at University of Sydney.

Kibler, D. and Langley, P. 1988. *Machine Learning as an experimental science.* Proceedings of the Third European Working Session on Learning. Pitman publishing, London, UK, pp. 81-99.

Kira, K. and Rendell, L. A. 1992.*A Practical Approach to Feature Selection.* Proceedings of the Ninth International Conference on Machine Learning .Morgan Kaufmann.

Kohavi, R. 1995.*A Study of Cross – Validation and Bootstrap for Accuracy Estimation and Model Selection* . Mellish, C.C. ,Editor, Proceedings of the 14th International joint Conference on Artificial Intelligence ,PP. 1137 – 1143. Morgan Kaufmann.

Kohavi, R. and George, J. 1998 .*The Wrapper Approach.* .Book chapter for feature selection for knowledge discovery and Data Mining (Kluwer International Series in Engineering and computer Science) .Huan Liu and Hiroshi Motoda, editors.

Kohavi, R. , George, J. and Pfleger, K. 1994 . *Irrelevant Features and the Subset Selection Problem.* Machine Learning :Proceedings of the Eleventh International Conference ,PP. 121 – 129.Morgan Kaufmann.

Kohavi, R. and Sommerfield, D. 1995. *Feature Subset Selection Using the Wrapper Model :Overfitting and Dynamic Search Space Topology.* The First International Conference on Knowledge Discovery and Data Mining ,PP. 192 – 197.

Kononenko, I. 1994. *Estimating Attributes :Analysis and Extensions of RELIEF* . Bergadona F. and Raedt L. D. ,editors ,Proceeding of the European Conference on Machine Learning.

Kononenko, I. and Simec, E. 1994. *Induction of Decision Trees using RELIEFF.* ISSEK Workshop, Udine, 6-8.9.1994 (To appear in: R.Kruse, R.Viertl, G.Della Riccia (eds.),CISM Lecture Notes, Springer Verlag)

Langley, P. and Blum, A. 1992.*Selection of Relevant Features and examples in Machine learning* .Artificial Intelligence .PP 245 – 271.

Langley, P. , Iba, W. and Thompson, K. 1992. *An Analysis of Bayesian Classifier* .Proceedings of the Tehnth National Conference on Artificial Intelligence , PP.223-228 .San Jose, CA:AAAI Press.

Lavrac, N. , Gamberger, D. and Turney, 1998.*A Relevancy Filter for Constructive Induction.* IEEE Intelligence systems and their application, Vol. 13 No2 March/Aprial 1998.

Liu, H. and Setiono, R. 1996.*A Probabilistic Approach to Feature Selection –A Filter Solution.* L. Saitta(Ed) Proceedings of International Conference on Machine Learning (ICML-96),July 3-6 ,1996 ,PP. 319 – 327.Bari,Italy :San Francisco :Morgan Kaufmann Publishers,CA.

Mantaras, R. L. 1989. *A distance based creterion for attribute selection.* Proc. Int. Symb. Methodologies for intelligent systems, Charlotte, North Carolina, USA. , Oct. 1989.

Michalski, S. and Bloedron, E. 1998 *.Data - Driven Constructive Induction.* IEEE Intelligence systems and their application, Vol. 13 No2 March/April 1998.

Michalski, S. and Larson, J. B. 1978. *Selection of most representative Training examples and Incremental Generation of UL1.* Report No. 862 Dept. of computer science , University of Illinois Urban 1978.

Michalski, S. and Wenk, J. 1991. *Hypothesis – Driven Constructive Induction in A!17 :A method and Experiments.* MII Report 91- 9 ,Center for artificial Intelligence ,George Mason university ,Fairfax Va. 1991.

Murphy, P. M. and Aha, D. W. 1994. *"UCI repository of Machine Learning Databases"*, Available by anonymous ftp to **ics.uci.edu** in the **pub/machine-learning – databases** directory .

Pagallo, G. and Haussler, D. 1988. *Boolean Feature Discovery in Empirical Learning: Technical report* . Dept of Computer Science, Univ. of California, Santa Cruz.

Pagallo, G. and Haussler, D. 1989. *Two Algorithms that Learn DNF by discovering relevant features*. Proceedings of the 6th international Workshop on Machine Learning ,San Mateo,CA:Morgan Kaufmann, pp. 119 – 123.

Quinlan, J. R. 1986. *Induction of Decision Trees*, Machine Learning 1 , PP. 81-106

Quinlan, J. R. 1992. *Learning with Continuous Classes*. Proceedings of the 5th Australian Joint Conference on Artificial Intelligence, Singapore :world science, pp. 343 – 348.

Quinlan, J. R. 1993. *C4.5 : A Program for Machine Learning* .San Matheo, CA:Morgan Kauffmann.

Quinlan, J. R., Compton, P. J., Horn, K. A., and Lazarus, L. A. 1987. *Inductive Knowledge Acquisition: A case study* . In J. R. Quinlan ed., Applications of expert systems. PP.157 – 173. Wokingham, UK: Addison-Wesley.

Rendell, L. and Ragavan, H. 1993.*Lookahead Feature Construction for Learning Hard Concepts*. Proceedings of the 10th International Conference on Machine Learning ,San Mateo, Ca :Morgan Kaufmann, pp. 252 – 259.

Wilson, R. 1997. *Advances in Instance Based Learning Algorithms*. Ph.D. Dissertation, Computer Science Department, Brigham Young University, August 1997.

ملخص

تطوير طرق مختلفة لتمييز صفات الأصناف المتصلة والمنفصلة لأغراض التنبؤ

أعداد

فراس علي عليان البلص

المشرف

الدكتور خليل هندي

المشرف المشارك

الدكتور أحمد الجابر

تعتبر عملية اختيار وتمييز الصفات من المواضيع التي تأخذ قدرا" كبيرا" من الأهمية في مجال الذكاء الصناعي في علم الحاسب ، نظرا لاستخدام هذه المواضيع في الحياة العملية.

في هذا البحث قمنا بتطوير طرق جديده لاختيار الصفات. وتتمثل الطريقة الأولى بدمج طريقتين تم استخدامهما سابقا بطريقة واحدة. قمنا بعمل الاختبارات على هذه الطريقة باستخدام مجموعه مكونه من عشرة نماذج مختلفة من المعلومات وقد أبدت الطريقة الجديدة تحسنا" عن الطرق السابقة في اختيار الصفات وتحسين على الدقة المحسوبة. كما قمنا بتطوير طريقة لدمج الصفات في النماذج التي تستخدم الأصناف المتصلة. وقمنا بعمل اختبارات على مجموعة من النماذج وأبدت نتائج هذه الاختبارات تحسنا" في الدقة مع الإبقاء على عدد قليل من الصفات الجديدة.

ونتيجة للعمل في هذا البحث يمكن الاستنتاج بان وجود عدد من الصفات غير الضرورية في بعض النماذج يؤدي إلى صعوبة في فهم مكونات النموذج كما يؤدي إلى عدم الدقة في الحكم على النتائج عند دراسة النموذج.لذلك وجدت طرق لاختيار أو دمج بعض الصفات للتحسين من دقة النتائج ولزيادة فهم النماذج المختلفة.