Unsupervised Feature Subset Selection

Master Thesis
by

Nicolaj Søndberg-Madsen & Casper Thomsen

Faculty of Engineering and Science
Aalborg University

Department of Computer Science

TITLE:
Unsupervised Feature Subset Selection.

SEMESTER PERIOD:
DAT6,
January 20th - June 13th, 2003

PROJECT GROUP:
E4-206

GROUP MEMBERS:
Nicolaj Søndberg-Madsen, nicolaj@flaeskesteg.dk
Casper Thomsen, casper@flaeskesteg.dk

SUPERVISOR:
Jose M. Peña, jmp@cs.auc.dk

NUMBER OF COPIES: 6

NUMBER OF PAGES: 81

PAGES IN APPENDIX: 3

TOTAL NUMBER OF PAGES: 90

FRONTPAGE ILLUSTRATION BY:

Mirjam Søndberg-Madsen, mirjam@soendberg-madsen.dk

SYNOPSIS:

This master thesis has been developed in the domain of Decision Support Systems and it covers the sparsely researched area of unsupervised feature subset selection for data clustering. In the report we discuss what characterizes features that are relevant for data clustering and we propose new relevance score measures which are capable of producing a ranking of the features with respect to their relevance. The relevance scores, combined with a threshold, can be used in a filter approach where the uninformative features are discarded. The report proposes two methods for setting a threshold and the score measures are tested empirically on 3 synthetic data sets and 4 real world data sets. In a second step we propose to use the relevance rankings in a hybrid approach to performing unsupervised feature subset selection. This method allows us to perform unsupervised feature subset selection with less model inductions than ordinary wrapper approaches. Empirical tests show both the filter and hybrid approaches to perform satisfactory.

Preface

This report has been written as documentation of the second part of a master thesis at the Department of Computer Science at Aalborg University, Denmark. The report has been written during the period from the 20th of January to the 13th of July, 2003. The project will be evaluated on the 27th of July, 2003.

The inspiration for the project comes from a data mining project developed during the first part of this masters thesis. An article covering the central parts of the project has been submitted to the Fourteenth European Conference on Machine Learning and the Seventh European Conference on Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD 2003).

In the report we follow some general conventions which the reader should be familiar with before proceeding. The references in the report are marked with a number corresponding to the numbers in the literature list like [3]. Certain phrases are abbreviated. Abbreviations appear in parenthesis after the full phrase, for example "... feature subset selection (FSS) ...". *Italics* are used to mark the introduction of a new technical term and the term can be expected to be explained short after.

We would like to thank our supervisor Jose M. Peña for his great help, valuable comments and enriching discussions that have given the project more than we could hope for. In addition we would like to thank Thorsten Ottesen and Dennis Kristensen for practical and implementation specific help without which some results would have been impossible to obtain in the short time available. We are also grateful for the contribution provided by Mirjam Søndberg-Madsen who is responsible for the design of the front page.

Nicolaj Søndberg-Madsen

Casper Thomsen

Contents

1 Introduction

Research is to see what everybody else has seen, and to think what nobody else has thought.
— Albert Szent-Györgi

Decision support systems take an increasingly important role in applications today. Lately focus has turned to data mining as a new big industrial tool to use in a wide area of applications. Data mining is the process of searching through data looking for meaningful characteristics and trends. It uses statistical analysis and machine learning techniques, such as neural networks and decision trees, to find the relationships in the data that ordinary interaction with the database would not find. This allows identification of undetected relationships between items such as associations between products, sequences of events that lead to later events, and new information.

Data mining has its applications within science and research as well as in the industry and in business applications. It suits perfectly within application areas where there is a huge amount of factors each potentially capable of affecting the application area. The medical research societies which deal with a huge amount of data such as DNA-profiles, symptoms, blood-types etc., have been using data mining with success. In addition data mining has also received attention in commercial domains. The following is an often referred example of a successful application of data mining performed by an American supermarket chain. It illustrates how the process of examining raw data, drawing mature conclusions and as a consequence, deploying the result can result in an improved understanding of a business. Moreover, the resulting knowledge increased profit for the supermarket.

> “For example, one Midwest grocery chain used the data mining capacity of Oracle software to analyze local buying patterns. They discovered that when men bought diapers on Thursdays and Saturdays, they also tended to buy beer... The retailer concluded that they purchased the beer to have it available for the upcoming weekend. The grocery chain could use this newly discovered information in various ways to increase revenue. For example, they could move the beer display closer to the diaper display.” [52]

In addition the authors of this report have participated in an industrial data mining project recently [37]. In this project a thorough analysis of a textile company's data yielded a new description of how the dealers in the company behave which can be used to explain why some end their career or to target campaigns for hiring better dealers.

In general, the need for data mining is a result of a growing amount of data stored within even the smallest companies. In many cases there is a lot of hidden information in that data which can be used to predict events in the future or to make a detailed description of the present. *Clustering* and *classification* techniques are developed for the purpose of dealing with many of the tasks that appear in data mining projects.

In classification each record in the database is assumed to belong to a predefined class which is determined by one of the attributes, namely the *class label*. A *predictive model* is produced by analyzing each record in a database where the class label is known. When the model is completed it can be used to predict or classify yet unseen records. Classification is also referred to as *supervised learning* [28]. On the other hand data clustering aims to describe the group-structure which is underlying in a given data set. As opposed to classification, clustering generates a model without consulting a class label which explains why it is referred to as *unsupervised learning*. Generally, clustering is divided into two groups, *partitional* and *probabilistic* clustering. Partitional clustering yields a description by dividing the data into a partition whereas in probabilistic clustering we construct a probabilistic model of the data.

One of the critical tasks in data mining is data clustering [37]. In this part of data mining several factors potentially influence the results, for instance the number of clusters k, and the production of a meaningful description of the structure which is hidden in the data. In addition an important factor is the size of the data. It is most critical if databases consist of a huge amount of features. In many cases some of the features are not informative for the purpose of learning from the data and can be considered as noise and we say that they are irrelevant. Such features have a negative impact on learning in that they introduce distortion rendering the results less accurate. In addition the complexity of any learned model increases in the number of features, thus including irrelevant features will make the learned model harder to comprehend and increase the cost of induction. Moreover, if a part of the data base can be discovered, which can be left out without doing any harm to the learned model, this combined with the model description, can be regarded as valuable information.

Therefore in this project we focus only on the problem of reducing the number of features. This problem is usually referred to as *feature subset selection* (FSS). FSS is the process of identifying the most effective subset of the original features in a data set for a particular purpose and it is a central problem in data analysis [17, 34]. It can be performed both supervised and unsupervised. Supervised FSS is applied in classification where the class label is known and finding the optimal subset can be considered a search problem where a given subset can be tested against the class labels. Similarly, unsupervised FSS is performed in data clustering. In unsupervised FSS a test against a class label does not exist and so other techniques must be developed in order to evaluate a given subset.

In the field of unsupervised FSS there has not been performed a great deal of research currently [14, 34]. There is however a growing need for reducing the dimensionality of data for clustering which further motivates this project.

We are in this project concerned with the problem of unsupervised FSS as the identification of irrelevant features for data clustering. Therefore we wish to identify the characteristics which must account for irrelevant features and propose a method which can effectively discard irrelevant features resulting in a more comprehensible model without doing any harm to the learned model.

2 Clustering

The notion of finding 'natural groups' tends to imply that the algorithm should passively conform like a wet teeshirt.
— Michael R. Anderberg

Many problems which arise in data mining can be solved by using data clustering [3, 13, 29]. In general data clustering is regarded as an important way of summarizing data in an understandable manner [18]. Despite its widespread use there exists different definitions, interpretations and expectations of which the term *clustering* gives rise to [33, 42, 47]. Therefore in order to continue our discussion of data clustering we define the concept of data clustering. First, we outline the assumptions on which any clustering technique is based. Then we introduce 2 different clustering techniques: a partitional clustering algorithm and a model-based clustering algorithm.

2.1 Clustering

Clustering is a process of discovering groups in data [34]. It yields a description of the group structure which is hidden in the data when the group memberships are unknown [54]. The discovery process aims to discover classes in the data which are natural for the data set. It is clear that it only makes sense to identify groups if some groups exists. Therefore, clustering is based on the assumption that the data is generated by an underlying model which is responsible for such groups. Specifically, the purpose of clustering is to gain more information about this model. Figure 2.1 depicts a mechanism which is often used to explain the underlying model. It consists of a *selector*, a number of physical processes and the data set.

The assumption is that each instance in the data set is generated by this mechanism. For each instance the selector selects one and only one of the physical processes. The physical process then generates each attribute value of the

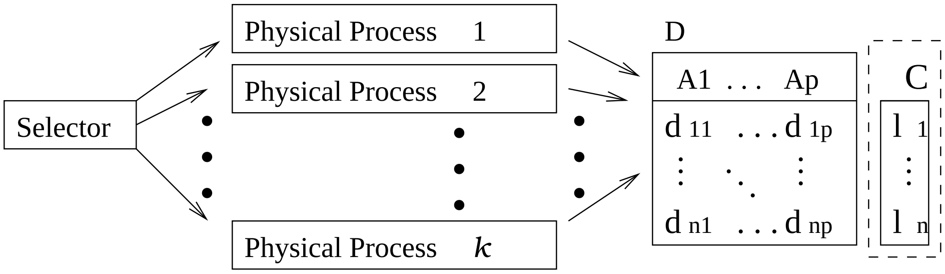

Figure 2.1: The underlying model: Each instance in the data set is generated by a physical process, selected by a selector which remains unknown for us.

instance, based on an unknown probability distribution. In the end, all the instances generated by one physical process are assumed to belong to the same cluster. The clusters and the physical processes remain unknown or hidden, i.e. it is unknown which physical process is responsible for generating a specific instance and how it has been generated (its associated probability distribution). More specifically, cluster analysis is based on the following assumptions:

1. Clustering is applied to a data set D containing n instances, such that $D = \{\boldsymbol{x}_1 \dots \boldsymbol{x}_n\}$. Each instance $\boldsymbol{x}_i$ is a vector, of p values $x_{i1} \dots x_{ip}$. In addition we let $\boldsymbol{x}_i$ be an instantiation $\boldsymbol{x}$ of the p-dimensional vector of random variables $\boldsymbol{X} = \{X_1, \dots, X_p\}$.

2. Each instance $\boldsymbol{x}_i \in D$ is a member of one and only one of the underlying hidden clusters $C = \{c_1, \dots c_k\}$. The cluster membership is represented by the label l_i assigned to each $\boldsymbol{x_i}$. Since this cluster membership is unknown (hidden cluster membership) we refer to C as the *hidden cluster membership variable* or simply the *cluster variable*.

3. D is generated by an underlying model consisting of k physical processes which, together with the selector are represented by a joint probability distribution.

We denote the joint probability distribution for the selector $P(c)$, the probability for each physical process to be selected, and the joint probability distribution for each physical process as $P(x_1, \dots x_p|c)$ or simply $P(\boldsymbol{x}|c)$, the probability of generating a case $\boldsymbol{x}$ given the cluster membership C.

In addition it is worth to mention that the instances in D can be of any type, i.e. categorical, real valued or even a mix of those kinds. However we want to make the following constrain: In this project we will only allow the type of instances in D to be categorical.

In general, clustering can be regarded as an optimization problem. Given a data set D, a feature subset and the number of clusters k must be selected which results in an optimal clustering model.

The resulting model, must be optimal with respect to some measurement function which assigns a score to each possible model. This measurement is based on an intuitive understanding of heterogeneity and homogeneity which is referred to as the *clustering criterion*. If this criterion is translated into a mathematical formula which measures the homogeneity within each cluster, the clustering problem is left as a search for the model that yields the most homogeneous clusters. There exist two classical models of the clusters, *partitional* and *probabilistic*.

A partitional data clustering (also called *partitioning*) algorithm partitions a dataset into k clusters such that instances in the same cluster are more similar than instances in other clusters. The process is required to be *exhaustive*, i.e. all instances in D are assigned to a cluster while each cluster is required to be *non-empty* and *mutually exclusive*. That is, each cluster must contain at least one instance and each instance is assigned to one and only one cluster.

The probabilistic models however, describe the clusters by modeling the mechanism that generated the data. These methods are regarded as more advanced than the partitional algorithms due to their well-founded base in statistics [13, 47, 54]. After identifying a number of clusters it recovers the probability distributions $p(c)$ and $P(\boldsymbol{x}|c)$ of the underlying model.

In unsupervised learning the number of clusters is usually unknown. As this factor can have a large impact on the result we assume it to be known for all data sets used in the remainder of the report. Finding k is out of scope for this project.

In the following we introduce the two clustering methods we are going to use in this project, a partitional technique, called the k-modes algorithm and a model-based technique called the *Naive Bayes* (NB) Model.

2.2 The k-modes Algorithm

The k-modes algorithm was introduced by Huang [31] as a variation of the well known k-means algorithm [46]. k-modes runs on categorical data and maintains the efficiency that k-means exhibits on large data sets.

The k-modes algorithm iterates through the data set and assigns each instance to the cluster that contains more similar instances than the other clusters and keeps repeating this untill convergence has been obtained, i.e. the result of iteration i is equal to the result of iteration $i-1$. The assignments make use of the clustering criterion which varies for different clustering methods.

2.2.1 Definitions

We run the k-modes algorithm on a data set D of n instances. Each instance $\boldsymbol{x} \in D$ is a vector of p nominal values $x_1 \ldots x_p$. The k-modes algorithm partitions D into k clusters $c_1 \ldots c_k$, by assigning each instance to the cluster with the most similar cases, or more correct, the least dissimilar cases. To measure the similarity between cases we use a dissimilarity measure. Let $d(\boldsymbol{x}, \boldsymbol{y})$ be the dissimilarity between a pair of cases $\boldsymbol{x}$ and $\boldsymbol{y}$. Then the dissimilarity returned by d is the total number of mismatches of the corresponding attribute categories of the two cases [31]. We have

$$d(\boldsymbol{x}, \boldsymbol{y}) = \sum_{j=1}^{p} \delta(x_j, y_j) \tag{2.1}$$

where

$$\delta(x_j, y_j) = \begin{cases} 0 & (x_j = y_j) \\ 1 & (x_j \neq y_j) \end{cases} \tag{2.2}$$

The dissimilarity measure is also known as the *Hamming distance* [6].

Each cluster is represented by a prototype, or a *mode*. A mode is a nominal vector of size p that minimizes:

$$\sum_{\boldsymbol{x} \in c_l} d(\boldsymbol{x}, \boldsymbol{q_l}) \tag{2.3}$$

for each case $\boldsymbol{x} \in c_l$, and c_l is a cluster represented by $\boldsymbol{q_l}$.

A mode $\boldsymbol{q}$ is both initiated and updated using a frequency based method. For each attribute x_j in the subset of the data set assigned to cluster c_l we search for the most frequently occurring state. The state of $\boldsymbol{q}$ at index j will thus be be updated to represent to most frequently occurring state in c_l.

2.2.2 The Algorithm

The essence of the k-modes algorithm is the search for a partitioning which is optimal with respect to a certain *cost function*. The cost function is the sum of Hamming distances from each instance to the mode of the cluster to which it is assigned.

The cost function which must be minimized is:

$$E = \sum_{l=1}^{k} \sum_{\boldsymbol{x} \in c_l} d(\boldsymbol{x}, \boldsymbol{q}_l), \tag{2.4}$$

where $\boldsymbol{q}_l$ is the cluster mode of cluster c_l and $\boldsymbol{x} \in c_l$ is the set of cases assigned to cluster c_l. The k-modes algorithm consists of the following steps:

1. Select k initial modes, one for each cluster.

2. Use Equation 2.1 to assign each instance to the cluster with the most similar mode. Each time an instance has been allocated to a cluster, recalculate the cluster mode using Equation 2.3.
3. After all instances have been allocated to clusters, retest the dissimilarity of instances against the current modes. If an instance is found such that its nearest mode belongs to another cluster rather than its current one, reallocate the instance to that cluster and update the modes for both clusters.
4. Repeat 3 until convergence has been reached, i.e. no instance has been reassigned after a full cycle test of the whole data set.

Like the k-means algorithm, the k-modes algorithm is likely to produce locally optimal solutions that are dependent on the initial modes and the order of objects in the data set [55]. Therefore it is appropriate to run the k-modes algorithm several times with different initial modes and pick the best result with respect to the cost function [31]. To pick the initial modes totally at random might not be appropriate since there might be a risk that one or more cluster modes will be assigned values such that no, or very few instances will be assigned to it. Therefore we have chosen to modify Step 1 and 2 in the above algorithm in order to obtain initial modes that are close to the data:

1. Assign each instance in the data set to one of the k clusters chosen at random, ensuring that each mode will be assigned at least one instance.
2. When all instances are assigned to a cluster, calculate the k cluster modes using Equation 2.3. If two modes are identical, restart from step 1.

2.3 Model-Based Clustering

As already mentioned there are two main approaches to clustering, namely partitional and probabilistic clustering. The latter can provide each case with a probability distribution with the probability of each cluster. The latter approach is sometimes called a soft (or *fractional*) assignment as opposed to the hard assignments performed by partitioning.

2.3.1 Finite Mixture Models

As mentioned model-based clustering is an attempt to model the process which has generated the data. Thus a model contains the probability distribution modeling the selector and a separate probability distribution for each cluster. The fact that the number of clusters is assumed to be finite and the model is a mix of models, one for each cluster, has led to the name *finite mixture models*. The aim of a finite mixture model is to model the joint probability mass function $p\left(\boldsymbol{x}|\theta\right)$ which is most likely to have generated the data D. We have:

$$\begin{aligned} p(\boldsymbol{x}|\theta) &= \sum_{i=1}^{k} p(c_i|\theta) \;\; p(\boldsymbol{x}|c_i,\theta_i) \\ &= \sum_{i=1}^{k} \pi_i \;\; p(\boldsymbol{x}|c_i,\theta_i) \qquad (2.5) \end{aligned}$$

where $\pi_i = p(c_i|\theta)$ is the marginal probability of each cluster such that $\sum_i \pi_i = 1$, $p(\boldsymbol{x}|c_i,\theta_i)$ is the probability distribution which is modeling cases in the i'th cluster, and θ are the parameters of the model where $\theta = \{\pi_1,\ldots,\pi_k,\theta_1,\ldots,\theta_k\}$.

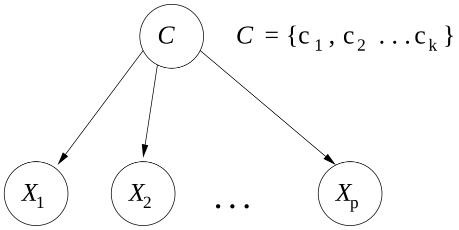

Figure 2.2: The NB Model. The variables $X_1 \ldots X_p$ are independent given the cluster variable.

In model-based clustering a model structure and the probability distributions associated with each cluster is learned from a data base. However it is common that one would stick to a fixed structure beforehand. A widely used fixed structure of finite mixture models is the NB model. The NB model is a model where an assumption of conditional independence among each pair of the variables in $\boldsymbol{X} = \{X_1, \ldots X_p\}$ given the cluster random variable C is made. Under this assumption $p(\boldsymbol{x}|c_i,\theta_i)$ can be calculated as

$$p(\boldsymbol{x}|c_i,\theta_i) = \prod_{l=1}^{p} p(x_l|\theta_i^l), \qquad (2.6)$$

where $p(x_l|\theta_i^l)$ is the probability distribution over the values for the variable X_l and θ_i^l is the set of parameters. Figure 2.2 depicts an NB model, where C is the cluster membership variable and each $X_l \in \boldsymbol{X}$ are the variables in the data base.

In order to assign a case $\boldsymbol{x}$ to a cluster c_i we need the probability of the cluster membership given $\boldsymbol{x}$, namely the *cluster membership probabilities*. We use Bayes rule to get

$$\begin{aligned} p(c_i|\boldsymbol{x},\theta) &= \frac{\pi_i \;\, p(\boldsymbol{x}|c_i,\theta_i)}{\sum_{j=1}^{k} \pi_j \;\, p(\boldsymbol{x}|c_j,\theta_j)} \\ &= \frac{\pi_i \prod_{l=1}^{p} \, p(x_l|\theta_i^l)}{\sum_{j=1}^{k} \pi_j \prod_{l=1}^{p} \, p(x_l|\theta_j^l)} \end{aligned} \tag{2.7}$$

which can be used to assign each case $\boldsymbol{x} \in D$ the most likely cluster, or to perform a soft assignment where each case $\boldsymbol{x}$ is assigned fractionally to the set of clusters according to the distribution $p(c_i|\boldsymbol{x},\theta)$.

2.3.2 Learning a Naive Bayes Model from Data

In order to learn a model from a set of data $D = \{\boldsymbol{x}_1 \ldots \boldsymbol{x}_n\}$ we search for the parameters which maximize the likelihood of the training data, $L(D|\theta)$. The most likely θ is usually denoted $\hat{\theta}$ and this approach to finding the parameters θ is called the *maximum likelihood criterion* (ML):

$$\hat{\theta} = argmax_\theta \;\; L(D|\theta) = argmax_\theta \prod_{\boldsymbol{x} \in D} p(\boldsymbol{x}|\theta). \tag{2.8}$$

Let n_i^{lj} denote the number of cases in the database which belong to the ith cluster and for which the lth variable is in state j. Similarly, let θ_i^{lj} denote the probability that, for a given case in cluster i, the lth variable is in state j. The maximum likelihood criterion is known to be:

$$\hat{\theta}_i^{lj} = \frac{n_i^{lj}}{n_i}, \tag{2.9}$$

where $n_i = \sum_j n_i^{lj}$. Similarly, the marginal probabilities of the ith cluster π_i are found as:

$$\pi_i = \frac{n_i}{n}. \tag{2.10}$$

This approach is an analysis of the frequencies of occurrences in the data only. In some cases, when one wants to incorporate prior knowledge about the probability distributions one may want to use the *maximum a posteriori* (MAP) estimate. Let $p(\theta)$ denote our prior knowledge about the parameters, then we have:

$$\begin{aligned} \hat{\theta}_{\mathsf{MAP}} = argmax_\theta \, p(\theta|D) &= argmax_\theta \, L(D|\theta) \frac{p(\theta)}{L(D)} \\ &= argmax_\theta \, L(D|\theta) \, p(\theta) \end{aligned} \tag{2.11}$$

Let α_i^{lj} denote the prior knowledge we have for cases in the ith cluster with variable l in state j, where $\alpha_i = \sum_j \alpha_i^{lj}$. The MAP estimate is then:

$$\theta_i^{lj} = \frac{\alpha_i^{lj} + n_i^{lj}}{\alpha_i + n_i}; \quad n_i = \sum_j n_i^{lj}; \quad \alpha_i^{lj}, \, \alpha_i \, \geq \, 0, \tag{2.12}$$

and similar for the marginal cluster probability for the ith cluster which is found as:

$$\pi_i = \frac{\alpha_i + n_i}{\alpha + n}; \quad \alpha = \sum_j \alpha_j; \quad \alpha_i,\, \alpha \,\geq\, 0. \tag{2.13}$$

2.3.3 Learning a Naive Bayes Model for Clustering

The above approach can be used to learn an NB model from data. However, in clustering the cluster membership is unknown. This constitutes a problem since the cluster membership variable is assumed to be known in the above approach when the values for θ are estimated. Therefore clustering can be regarded as a special case of learning a model from data with missing values. Therefore we need an algorithm which is able to deal with missing values. One well known algorithm for learning parameters of a probabilistic model from a data set with missing values is the *Expectation - Maximization* algorithm, or simply the EM algorithm [11]. It consists of two steps, namely the expectation (E) step and the maximization (M) step. In the E step each case in the database $\boldsymbol{x} \in D$ is assigned the posterior probability of its cluster membership (cluster membership distribution) using Equation 2.7. In the M step these probabilities are considered as real data and the parameters θ of the model are learned using ML estimates or MAP estimates. After each iteration the algorithm measures the performance of the parameters. The performance of the parameters θ on a data set D is given as the likelihood of the data given the parameters, $L(D|\theta)$:

$$\begin{aligned} \text{Performance}(\theta) &= L(D|\theta) = \prod_{\boldsymbol{x} \in D} p(\boldsymbol{x}|\theta) \\ &= \prod_{\boldsymbol{x} \in D} \sum_{i=1}^{k} \pi_i \prod_{l=1}^{p} p(x_l|\theta_i^l). \end{aligned} \tag{2.14}$$

It is sometimes convenient to use the logarithm of the likelihood (*log likelihood*) to measure the performance of a model:

$$\text{Performance}(\theta) = \sum_{\boldsymbol{x} \in D} log \left[\sum_{i=1}^{k} \pi_i \prod_{l=1}^{p} p(x_l|\theta_i^l) \right]. \tag{2.15}$$

The E step and the M step are repeated until a certain stopping criterion is met. As with the k-modes and k-means algorithms the stopping criterion is when the model has reached convergence, i.e. when the parameters θ have not been changed during the last iteration of the E step and the M step. Sometimes it is convenient to have a more fuzzy understanding of the term convergence, in such cases one would choose a a threshold γ as stopping criterion. If the improvement in performance, measured by the log likelihood of the model, in the last iteration of the E and M step is less than γ, convergence has been reached, and the algorithm is terminated. In this work we will stick to a threshold $\gamma = 10^{-6}$.

2.3.4 Implementing the Expectation Step

The E step performs a fractional completion of the database where each case is assigned fractionally to clusters. For this purpose we need $p(c_i|\boldsymbol{x},\theta)\,\forall\, i$. Thanks to the conditional independencies in the NB model (Equation 2.6) we can use Equation 2.7 for this purpose. Here we introduce an example.

Table 2.1 and 2.2 show an example of 2 components associated with cluster c_1 and c_2 respectively.

c_1	x_1	x_2	x_3
$p(x_i = 1)$	0.3	0.2	0.9
$p(x_i = 2)$	0.7	0.8	0.1

Table 2.1: The probability distributions for component 1 conditioned on cluster $c1$.

c_2	x_1	x_2	x_3
$p(x_i = 1)$	0.5	0.4	0.7
$p(x_i = 2)$	0.5	0.6	0.3

Table 2.2: The probability distributions for component 2 conditioned on cluster $c2$.

First we assume that we have the prior probabilities for $p(c_1) = 0.4$ and $p(c_2) = 0.6$ we then have for a given case $\boldsymbol{X} = [1, 2, 2]$:

$$p(c_1|\boldsymbol{X} = [1,2,2], \theta_1) = \frac{0.4 * (0.3 * 0.8 * 0.1)}{0.4 * (0.3 * 0.8 * 0.1) + 0.6 * (0.5 * 0.6 * 0.3)} = 0.16$$

$$p(c_2|\boldsymbol{X} = [1,2,2], \theta_2) = \frac{0.6 * (0.5 * 0.6 * 0.3)}{0.4 * (0.3 * 0.8 * 0.1) + 0.6 * (0.5 * 0.6 * 0.3)} = 0.84$$

In the E step all cases in the database we assign the cluster membership distribution as described here. In the above example all cases with the configuration $\boldsymbol{X} =[1, 2, 2]$ are assigned the probabilities 0.16 and 0.84 for cluster c_1 and c_2 respectively.

2.3.5 Implementing the Maximization Step

In the maximization step we assume the cluster membership distributions obtained in the previous E step are real data and recalculate the parameters of the model given these distributions. This is done using the ML estimate or the MAP estimate. In our case we use the MAP estimate, and since we have no reason to prefer some parameter values above others we consider all to be equally likely a priori. That is, we use a uniform prior probability distribution.

What needs to be done is to update the parameters θ of the model. That is, the probability distributions within each component $p(\boldsymbol{x}|c_i, \theta_i)$ and the marginal probabilities $p(c_i)$. This is done using Equations 2.12 and 2.13 with one minor change. Since the E step has assigned fractional cluster membership probabilities to each case instead of hard assignments the frequency analysis can not be performed by counting cases. Instead the probabilities are summed to obtain $n_i = \sum_{\boldsymbol{x} \in D} p(c_i|\boldsymbol{x}, \theta)$ and $n_i^{lj} = \sum_{\boldsymbol{x} \in D} p(x_l = j, c_i|\theta_i^{lj})$ when applying Equations 2.12 and 2.13. In fact, counting frequencies of hard assignments can be regarded as a special case of the above two sums where the probabilities are 0 or 1.

Let us assume we have a database of the same dimensionality as in the previous example (any other equality is pure coincidence) in which the cluster membership probabilities have been attached to each case in the previous E step.

x_1	x_2	x_3	$p(c_1\|\boldsymbol{x})$	$p(c_2\|\boldsymbol{x})$
1	2	1	0.6	0.4
1	1	2	0.3	0.7
2	2	1	0.2	0.8
2	2	2	0.9	0.1
1	2	1	0.6	0.4

Table 2.3: Data instances with attached cluster membership probabilities.

We consider the 5 cases in Table 2.3 and estimate the marginal probabilities for $p(c)$ using Equation 2.13. We use the MAP estimate and consider the database with the new fractional cluster assignments as real data. Since, with the fractional assignments we can not count the number of cases assigned to each cluster n_i, we sum the fractional probabilities, i.e. we let $n_i = \sum_{\boldsymbol{x} \in D} p(c_i|\boldsymbol{x}, \theta)$ when applying Equation 2.13. If we apply the MAP estimate with uniform priors we get:

$$p(c_1) = \frac{1 + (0.6 + 0.3 + 0.2 + 0.9 + 0.6)}{2 + 5} = 0.514$$

$$p(c_2) = \frac{1 + (0.4 + 0.7 + 0.8 + 0.9 + 0.4)}{2 + 5} = 0.496$$

To update the parameters θ_i^{lj} we iterate through each configuration of each of the components. We apply Equation 2.12 and like before we use the fractional cluster membership assignments as real data. Therefore we let $n_i^{lj} = \sum_{\boldsymbol{x} \in D} p(x_l = j, c_i|\theta_i^{lj})$ when estimating $p(\boldsymbol{X} = \boldsymbol{x}_1|c_1, \theta_1)$. For $\boldsymbol{x}_1 = 1$ we get: $\frac{1+0.6+0.3+0.6}{2+2.6} = 0.54$ while for $\boldsymbol{x}_1 = 2$ we get: $\frac{1+0.2+0.9}{2+2.6} = 0.46$.

One question which remains is how to find some appropriate starting parameters for learning a model. An approach which has shown its worth is one proposed by Thiesson et al. [66]. The idea in this method is to estimate the parameters in a single-component model from the data using the MAP estimate and use this component to generate k components by perturbing the parameters of the single-component model. In other words, the parameters of the single component model are changed slightly at random in order to generate k unique components.

2.4 Summary

In this chapter we have described the concepts of data clustering which form a base for the proposed FSS methods. The assumptions on which all data clustering techniques are based, have been outlined to give a basic understanding of data clustering.

In the remainder of the report we will be using 2 clustering techniques: NB model and k-modes. The methods have been chosen since they are both able to deal with categorical data which is the only type of data used in this project. In addition the methods are part of 2 fundamentally different types of clustering. The NB model belongs to the group of model based clustering techniques whereas k-modes belongs to the group of partitional clustering techniques.

One of the unknown factors in unsupervised learning is the number of clusters. As this issue is out of scope for this project we will assume the number of clusters k to be known.

3 Feature Subset Selection

It is probably the choice of variables that has the greatest influence on the ultimate results of a cluster analysis.
— Michael R. Anderberg

In general, *feature subset selection* (FSS) is motivated by a wish to reduce the dimensionality of large data sets since data analysis using induction algorithms can be both highly time and space consuming. Moreover, models of large data sets tend to be harder to comprehend than models learned on smaller data sets. In case FSS can be performed effectively decreasing the number of features in the data base, the problem would be left computationally more feasible for the induction algorithms while the learned model would be more comprehensible. But what may also be considered important is that a clear distinction between relevant and irrelevant features is also valuable information as part of a summarizing description of a data set.

Critics of FSS would state that a model learned from the whole data set will always perform equally well as a model learned from relevant features only, leaving any effort spent on FSS wasted. This is sometimes called the assumption of *monotonicity*, i.e. the performance of a theoretically ideal learning algorithm is not damaged by the presence of noise [63]. However, empirical tests in [67] show that the inclusion of noise in some cases decrease the performance of the induced model.

In this chapter we continue the discussion of FSS. First we present FSS as a search problem and outline the differences between FSS performed in supervised and unsupervised learning. Then we describe the main ideas of filter and wrapper approaches and discuss these in relation to related work. After a discussion this chapter ends with several new proposals for measuring the relevance of a feature for use in unsupervised FSS.

3.1 Feature Subset Selection as a Search Problem

The problem of selecting the optimal feature subset can be regarded as a search or optimization problem (e.g [10, 14, 23, 32, 39, 43, 27, 71]) where each subset of features is regarded a point in the search space. Any search method requires a starting point in the search space, a search strategy, an evaluation function and a stopping criterion [65]. An exhaustive search for the optimal feature subset is exponentially complex i.e. in a database of p features there exists 2^p possible subsets. In such a search space any realistic approach must rely on a heuristic search strategy.

A rough classification of search strategies for solving optimization problems could be to distinguish between *complete* and *heuristic* search. The underlying idea in complete search strategies is the systematic examination of all the solutions of the search space (e.g., depth-first, breadth-first, branch and bound, etc.). Unfortunately, complete search is usually impractical as most optimization problems involve large search spaces that make this approach computationally prohibitive. Moreover, according to [68], the majority of the most challenging optimization problems that come from the methodological development of new techniques in computer sciences as well as from real-world scenarios turn out to belong to the category of NP-hard problems [24]. These facts together with the lack of flexibility of those search strategies that are based on classical techniques of operational research and numerical analysis justify the use of heuristic search strategies [51, 53, 60]. Unlike complete search strategies, heuristic search strategies do not examine the whole search space of the problem being optimized but only those parts that are considered promising according to certain heuristic criteria. Although heuristic search strategies neither ensure that the final solution is a global optimum of the optimization problem at hand nor facilitate its mathematical modeling, they provide the user with a final solution that is near a global optimum in acceptable runtime. In other words, heuristic search strategies provide the user with a trade-off between effectiveness and efficiency, which is a question of capital importance when problem optimization is approached from an engineering perspective.

Heuristic search strategies can be further divided into *deterministic* and *non-deterministic* or *stochastic*. In deterministic heuristic search strategies (e.g., forward, backward, stepwise, hill-climbing, threshold accepting, etc.), the same final solution for a given optimization problem is always achieved under the same conditions. In other words, a deterministic heuristic search strategy maps every initial solution of the optimization problem to a single final one. On the other hand, non-deterministic heuristic search is motivated by trying to avoid getting stuck in a local optimum of the optimization problem at hand, usually by means of randomness [72]. Due to its stochastic nature, different runs of a non-deterministic heuristic search strategy might lead us to achieve different final solutions for a given optimization problem under the same conditions. While some of the stochastic heuristic search strategies store only one solution of the optimization problem at hand at each iteration (e.g., simulated annealing [40, 49]), other approaches exist. Some of these other approaches are grouped under the denomination of evolutionary algorithms. Some examples of classical evolutionary algorithms are genetic algorithms [25, 30], evolutionary programming [21, 22], and evolution strategies [59, 62]. See [5, 20, 25, 44] for reviews of these and some other.

3.2 Feature Subset Selection Overview

In the domain of FSS there are 2 main areas of interest: supervised and unsupervised FSS. Supervised FSS has for some time been the topic of much research whereas unsupervised FSS has only recently received attention due to the growing interest in the field of data mining. In this section we will give an overview of the 2 areas with specific focus on unsupervised FSS as it is the focus of this project.

3.2.1 Supervised Feature Subset Selection

The vast majority of research in FSS has been performed in the supervised learning paradigm paying little attention to the unsupervised learning paradigm [14, 65]. The main objective of FSS applied to supervised learning is to increase the classification accuracy of the learned model by removing noise. Knowing the class label of each instance makes evaluation of any feature subset possible. It is common to use a model's ability to predict the class label of yet unseen cases to measure the performance of a feature subset, e.g. John et al. [36] who use cross-validation. In other methods the presence of the class label has inspired the use of dependency based methods, where the dependency between each feature and the class label is measured in order to leave out irrelevant features (e.g. [27]).

3.2.2 Unsupervised Feature Subset Selection

Applying FSS to unsupervised learning is a challenging task of data analysis. Using the same procedure as for supervised learning is impossible due to the unknown class label. There exist no standard definition of relevance within unsupervised FSS and it will therefore be clearly stated in this report. For instance, a simple evaluation function proposed by Fisher [18] has been adapted by Talavera [65] for use in unsupervised FSS and named the *feature dependency measure* (FDM). FDM is a function that describes the average increase in the ability to guess the value of a feature given a second feature. This measure is based on the assumption that, in the absence of a class label, we can deem as irrelevant those features that exhibit low dependencies with the rest of the features. The FDM is defined as:

$$\frac{\sum_i \sum_j w \sum_{j_k} \left[P\left(X_k = x_{j_k} | X_i = x_{ij}\right)^2 - P\left(X_k = x_{j_k}\right)^2 \right]}{|\{i|X_i \neq X_k\}|} \tag{3.1}$$

Equation 3.1 takes into account the increase in predictiveness of one feature given another feature. The leftmost factor (w) is a weight which provide higher values to the most predictable values of a feature and is defined as:

$$w = P\left(X_i = x_{ij}\right). \tag{3.2}$$

The proposal has been tested using a naive filter model approach which calculates the feature dependency measure for each individual feature and then selecting the highest scoring features using a fixed predefined threshold which can differ for each case. Some other approaches can be found in the literature (e.g. [57]).

Contrarily to supervised learning no standard unified performance criterion exist in the unsupervised learning paradigm [57]. Variables such as the number of clusters k, the performance of a clustering result and the quality of the data can have an impact on the results. This means that the term "optimal FSS" differs in the interpretation of the data analyst and as such makes comparisons difficult.

3.3 Filters and Wrappers

John et al. [36] introduce the notion of *filters* and *wrappers* which constitutes two different ways of performing FSS. In this section we will outline both methods and present a discussion of their performance.

3.3.1 The Filter Approach

Figure 3.1 illustrates the filter approach. First the algorithm is passed a set of features. Then the irrelevant features are filtered out, based on the analyst's notion of relevance, and at last a subset of relevant features is passed to the learning algorithm. Therefore the main property which characterizes a filter approach is the independence of a learning algorithm.

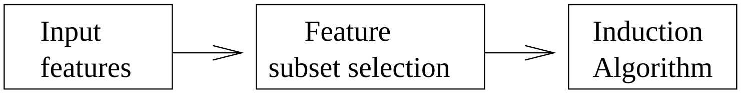

Figure 3.1: The filter model. Features are filtered out before the model is learned by the induction algorithm.

Selecting features, using a filter approach, is highly dependent on the understanding of relevance, i.e. it is necessary to have a measure of relevance for features. Such a measure depends on the machine learner's understanding of relevant features. Several works have proposed ways to measure relevance based on different definitions of relevance. John et al. [36] discuss 4 different definitions of relevance in the context of classification and show that the performance of the filter approach is highly dependent on the definition of a relevant feature.

There are several advantages for filter methods within clustering which is not a concern in classification. Since filtering is performed independently of the induction algorithm filters are independent of the performance of the learning algorithm and the success of an induced model. For instance, filters are independent of whether the optimal number of clusters k for a data set has been found.

Several proposals, such as Peña et al. [57] and Talavera [65], use a filter approach based on ranking each feature according to a *score measure* in order to be able to select a subset containing the most salient features (for Talavera's proposal see Section 3.2.2). Both proposals are based on unsupervised learning.

Peña et al. [57] propose a filter method using *conditional Gaussian networks* [45] in which they score a feature's relevance as the average likelihood ratio test statistics for excluding an edge between the measured feature and any other feature in the graphical Gaussian model [70]. The relevance measure for each feature X_i is written as:

$$\sum_{j=1, j\neq i}^{p} \frac{-n \; log\left(1 - r^2_{ij|rest}\right)}{p-1} \tag{3.3}$$

where p is the number of features in the database, n is then number of cases in the database and $r^2_{ij|rest}$ is the sample partial correlation of the features X_i and X_j adjusted for the remaining variables. The relevance measure allows to rank the features in a decreasing order with respect to relevance. The authors propose a heuristic which automatically decides on a relevance threshold. The relevance threshold is calculated as the rejection region boundary for an edge exclusion test in a graphical Gaussian model for the likelihood ratio test statistic. The features included in the learning are then those features which have a higher relevance score than the threshold.

3.3.2 The Wrapper Approach

John et al. [36] argue that it is a disadvantage that filters are independent of the induction algorithm and propose the wrapper approach to replace filters. In a wrapper the FSS algorithm is wrapped around the learning algorithm. Using a heuristic search strategy the wrapper searches through the space of feature subsets using the learning algorithm as a part of measuring the score of each feature subset. Each feature subset is evaluated by measuring the performance of the learned model. Figure 3.2 illustrates the wrapper approach. First a subset of features is selected according to some heuristic while secondly the subset is evaluated using the performance of the induction algorithm on the feature subset. Generally, subsets of features are evaluated through several iterations of this second phase. Each iteration requires a new model to be learned.

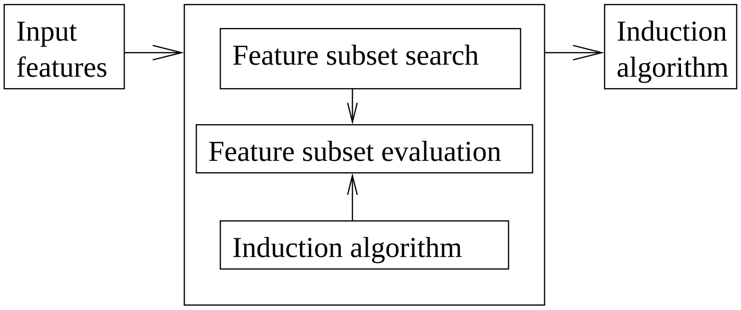

Figure 3.2: The wrapper model. FSS is performed as a "wrapper" around the induction algorithm.

The lack of a standard unified performance measure for unsupervised learning is a problem since the performance of the wrapper is highly dependent of the understanding of a successful cluster model. Another problem for wrappers for clustering is that the performance of the model depends on the number of clusters k which is unknown in most cases. Some works try to cope with this problem by considering finding k and the optimal features subset as one single optimization problem where the number of clusters and features are variables in the search. For instance, Dy et al. [14] propose to wrap FSS around the EM algorithm with order identification allowing to find the number of clusters, k, in the data. This approach solves 2 problems: It decreases the dimensionality of the database by removing irrelevant features and it finds the number of clusters which yields the optimal model with respect to a score criterion.

When comparing the two paradigms, filter and wrapper methods, one can not avoid the fact that the wrapper approach is much more time consuming than the filter approach. On the other hand wrappers tend to yield more precise models than models obtained by filter methods [36].

3.4 Score Measures

Before we move on to a discussion of a good relevance measure for unsupervised FSS, we want to discuss what we understand by good performance of a clustering model. The lack of a class label and a unified performance criterion has given rise to several proposals of how performance must be understood in clustering. This will yield a proposal of several new relevance measures to measure features relative to the rest of a data set which can be used for unsupervised FSS in a filter method.

3.4.1 Performance in Clustering

In general, a successful clustering is one that gives a description of any underlying group structure in the data if such exist. If we assume clusters exist, good clusters are clusters that are clear and easy to distinguish from the rest of the data. Fisher [18] is aware of this fact and he introduces two properties that can be measured, the *intra-cluster similarity* and the *inter-cluster dissimilarity*. They are measured by two posterior probabilities:

- The intra-cluster similarity: $P(X_i = x_{ij}|c_k)$, where X_i is a variable, x_{ij} is the jth value of X_i and c_k is a cluster. If this probability is high, the value of $X_i = x_{ij}$ is said to be *predictable* for the cluster members, and if it holds for many variables in the cluster c_k the cluster is said to be *cohesive*.
- The inter-cluster dissimilarity: $P(c_k|X_i = x_{ij})$. The higher this probability, the fewer clusters other than c_k share the value $X_i = x_{ij}$ which is then said to be *predictive*. If this probability is high for many of the variables within a cluster c_k, we say that c_k is *distinct*. [18, 65]

Dividing a data set into a good set of clusters should maximize these probabilities for a number of variables. Doing this, clusters formed on behalf of dependent features are rewarded. If a cluster c_k has a variable X_1 with high discriminating power the cluster will score a high $P(X_1 = x_{11}|c_k)$ and $P(c_k|X_1 = x_{11})$ since most of the values of X_1 will be x_{11} within the cluster and few values of X_1 will have the value x_{11} in other clusters. If X_1 is highly dependent on another variable, e.g. X_2, then most members of c_k will have the same value for X_2, say x_{21}. Hence both x_{11} and x_{21} contribute with both predictability and predictiveness making c_k more cohesive and more distinct. Thus in general, variables that are highly dependent on other variables contribute to achieve clusters that are both cohesive and distinct [65].

3.4.2 Relevance

One of the main problems in unsupervised FSS is to define relevance. In several previous proposals the definition has been based on a scoring criterion in which the score of each feature has been evaluated with respect to some measure. Using a predefined threshold each feature is then either deemed relevant or irrelevant.

We expect that if we know the performance of a feature subset S_i which consists of i relevant features, then adding one irrelevant feature to the feature subset, such that we have S_{i+1}, will not increase, or even decrease the performance of the clustering. We can evaluate the proposed score measures by evaluating the performance of the subset of features which were deemed relevant by the filter method and compare with subsets including irrelevant features. The essence of this test is that if the performance of the set of features S_i is not worse than the performance of the whole set of features $\boldsymbol{X}$, then the rest of the features, i.e. $\boldsymbol{X} \setminus S_i$ can be deemed irrelevant for clustering.

John et al. [41] discuss relevance in the context of supervised FSS but the definition can also be of interest for unsupervised FSS. They distinguish between strong and weak relevance and suggest the following definitions:

- A feature X is strongly relevant if removal of X alone will result in performance deterioration of an optimal Bayes classifier.
- A feature X is weakly relevant if it is not strongly relevant and there exists a subset, S, such that the performance of a Bayes classifier on S is worse than the performance on $S \cup \{X\}$.
- A feature is irrelevant if it is not strongly or weakly relevant.

As it is, this definition of relevance only works for wrapper approaches in supervised learning. However, we can transform this notion to filters if we know what characterizes features that would increase the performance of the classifier. Moreover, we can accept this definition of relevant features for clustering too. It only requires an agreement of what performance means in clustering. Therefore we use the above discussion of how to measure performance in clustering (that good clusters are both cohesive and distinct) in order to propose a measure of relevance for features. In allegory with the underlying model we say that for data instances $\boldsymbol{x} \in D$ only a subset of the variables are relevant. We regard as relevant those variables which are affected by the hidden cluster membership variable in the joint probability distribution $p(X|C)$. However, the cluster variable C, its number of states and impacts on the observed variables is unknown in clustering. What is known is that in the probability distribution $p(X|C)$ the cluster membership of a data instance has a different impact on some of the features. This can be modeled in a probabilistic model in the following way: We let the cluster membership be represented by the cluster variable, which has influence on the state of each variable in the model. In the case that there are features which are not influenced by the cluster membership these can be regarded as random variables, not connected to the hidden cluster membership variable. Figure 3.3 depicts an example of a model in which 4 features are influenced by the cluster membership while the 5th is a free random variable.

Elidan et al. [16] describe the impacts of hidden variables in probabilistic models and the interaction between observed variables and hidden variables. They argue that if a probabilistic model is learned with hidden variables, i.e. a variable which has influence to some of the nodes has been left out of the learning, the model will contain *semi-cliques*. Therefore semi-cliques can be regarded as an indication of the presence (or absence) of a hidden variable.

The explanation for this is that if a model is induced from a data set containing hidden variables, it will discover dependencies among the variables which

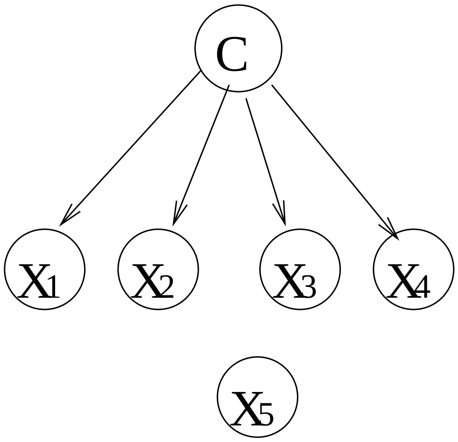

Figure 3.3: The cluster random variable has influence only on the relevant features.

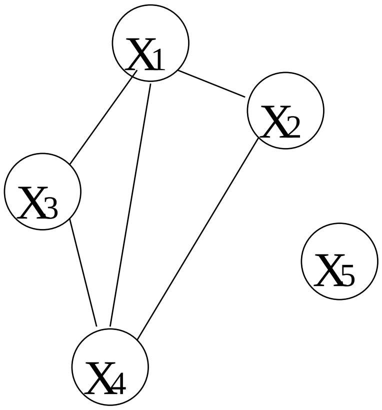

Figure 3.4: Without the cluster random variable the relevant variables form semi-cliques.

depend on the hidden variable. Logically this is caused by the dependency properties among the variables in the data set. Variables which depend on a hidden variable (i.e. its children) are d-connected if there is no evidence on the hidden variable, while the parent variables of the hidden variable are d-connected to its children under the same conditions [35]. Thus there is a dependency among variables which are influenced by the hidden cluster membership variable. Figure 3.4 depicts a model learned using the PC algorithm [64] and a data set generated by the model in Figure 3.3 from which the class label has been removed. We can use this as an illustrative example. The variables which were influenced by the class variable have connections to at least half of the rest of the variables and are said to be part of a semi-clique.

Since the cluster membership variable is hidden, we can use the argumentation that when hidden variables reside in a model the model will contain semi-cliques. But rather than restoring the model containing the hidden variable we use the above reasoning to argue which features are relevant. If we assume we have a database containing variables which depend on the cluster membership and variables which are free random variables without any dependencies we can expect the features which depend on the cluster variable to have many strong interdependencies. We use this in our definition of relevant features. Since a feature which is influenced by its cluster membership is likely to be part of a semi-clique we call a feature relevant when it is dependent of at least one other feature. Therefore, in this project we define relevance using dependencies. A feature without any dependencies is defined as irrelevant, whereas a subset of dependent features are defined as either relevant or irrelevant according to the strength of the dependencies among them. Measuring the strength of dependencies can be done either by assigning a value to connections between pairs of features or by scoring each feature based on the dependencies that exist for that feature. We propose to use the latter approach.

3.4.3 Redundancy

Untill now we have discussed the notion of relevance limited to a definition of relevant and irrelevant features. Another type of feature which must be dealt with is redundant features which also have a remarkable different way to be looked upon in classification and in clustering. It is not trivial to handle redundant features as it can be difficult to deem such features as either relevant or irrelevant. This section is meant to open a discussion on the importance of handling redundant features in the context of feature subset selection.

Merriam-Webster's [2] dictionary define redundance as: "exceeding what is necessary or normal". In the context of feature subset selection this can be applied to define a redundant feature as a feature that contributes with unnecessary information. This could be information which is already included in the clustering by another feature. That is why, in classification, features are filtered out if the information they contribute with about the class membership for each case is already present in the database. For instance, the information a feature contributes with can be redundant in terms of a copy of it or if it has a high dependency to one of the features already in the data set. As already mentioned in the previous section, in clustering the most homogeneous clusters are obtained from features which have dependencies among each other. Therefore, it can be said that in data clustering, we seek out redundancy. We do however dare to open a discussion about redundancy of features in data clustering. We say that a feature X_i is redundant in data clustering with respect to a data set D if X_i is relevant but it does not however contribute with homogeneity in models learned from D. That means that X_i has many strong dependencies to rest of the features in D and models learned from $D \setminus X_i$ are equally cohesive as models learned from D.

The term redundant can also according to Merriam and Websters mean: "serving as a duplicate for preventing failure". Consider a scenario in which a data base consist of 1000 features whereof 990 are redundant and the last 10 are relevant features. Clustering using all 1000 features give well defined cohesive clusters although perhaps hard to describe. In the event that redundant features are removed we will remove 989 feature leaving 11 left for clustering. By removing these features we also remove the weight given to these features leaving 990 features correspond to 1 feature equally important to each of 10 other features.

In a wrapper approach a feature can be considered irrelevant if it does not increase the cohesiveness of the resulting clusters (measured with respect to the clustering criterion). The impact of redundant features in the above mentioned scenario is that the cluster membership of a significantly large amount of cases change rendering the clustering results different although not more cohesive. Such features would according to a wrapper approach be deemed irrelevant although their presence have an impact on the final results.

In this project we deal only with detection of irrelevant features but even so redundancy can have an impact on the results and the performance of the results of the proposals.

3.4.4 Proposal Overview

In this report we propose to use a ranking scheme where each individual feature is assigned a score according to a measure of dependence with respect to the rest of the feature set. The main idea is that we assume that relevance can be expressed by the interdependencies in the feature set. Detecting whether 2 features in the feature set are dependent can be done in several different ways. We propose 3 different methods for determining dependency (referred to as *dependency measures*):

1. χ^2 analysis.
2. Predictive accuracy.
3. Information gain.

The χ^2 analysis is an obvious choice for testing dependency between two features. Among the many advantages by using this method is that a threshold, for distinguishing relevant from irrelevant features, is already defined in the *significance level*, and that its statistic has a well known distribution. The second measure represents the idea of measuring the change in predictive accuracy between pairs of features. This method is inspired by Talavera [65]. Inspired by Dash et al. [10] we also propose, as a third dependency measure, to use information gain, which is a measure of entropy to describe the dependencies among features. A naive approach is to measure the entropy of a single feature and the reduction in entropy based on adding a second feature in order to describe their dependencies.

Each of the 3 methods have their strengths and weaknesses. In this report we will test all 3 methods in order to test which will perform best for the proposal that will be presented.

3.4.5 χ^2 Analysis

The χ^2 *distribution* is a density distribution that is used in many hypothesis tests. The most common use of the χ^2 distribution is to test independence hypotheses. Although this test is by no means the only test based on the χ^2 distribution, it has come to be known as the χ^2 *test*. The χ^2 distribution has one parameter, its *degrees of freedom* (df).

When using χ^2 in order to test dependencies it is necessary to set up a hypothesis that can be either kept or rejected. Setting up and testing hypotheses is an essential part of statistical inference. In each problem considered, the question of interest is simplified into two competing hypotheses between which we have a choice: the null hypothesis, denoted H0, against an alternative hypothesis, denoted H1. These two competing hypotheses are not however treated on an equal basis. The null hypothesis is given priority, meaning that in order to be convinced that H1 holds we have to reject H0, whereas H0 holds if we cannot reject its existence. Thus the outcome of a hypothesis test is 'reject H0' or 'do not reject H0'. In this particular case H0 states that 'variable X_i is independent of variable X_j'.

In order to test the hypothesis using χ^2 it is necessary to extract the two attributes from the original data set and create a *contingency table* for them. A contingency table is a table of frequencies. A two-dimensional contingency table is formed by classifying subjects by two variables. One variable determines the row categories, the other variable defines the column categories. Each cell will then contain the frequency of occurrence in the data set where the variables are in the states given by the row and column category for the cell. For this to be possible both attributes are required to be categorical.

The parameter, degree of freedom, of the χ^2 distribution, originally proposed by Fisher [19], is the number of cells in the contingency table which can be manipulated without changing the marginal totals. A standard approximation of this proposal is:

$$df = (rows - 1) * (columns - 1) \tag{3.4}$$

A contingency table over the occurrence of values of the two variables is called a contingency table of *observed* values. In order to calculate the χ^2 *test statistic* it is necessary to calculate the contingency table of *expected* values. The expected values for a contingency table of observed values is calculated as:

$$E_{ij} = \frac{\left(\sum_k cell_{ik}\right)\left(\sum_k cell_{kj}\right)}{n} \tag{3.5}$$

where n represents the total number of instances in the data set.

The test statistic is a quantity calculated from the contingency tables of observed and expected values. Its value is used to decide whether or not the null hypothesis should be rejected in our hypothesis test using a threshold denoted as the *critical value*. The χ^2 test statistics is then calculated as:

$$\chi^2 = \sum_i \sum_j \frac{(O_{ij} - E_{ij})^2}{E_{ij}} \tag{3.6}$$

The critical value for a hypothesis test is a threshold to which the value of the *test statistic* in a sample is compared to determine whether or not the null

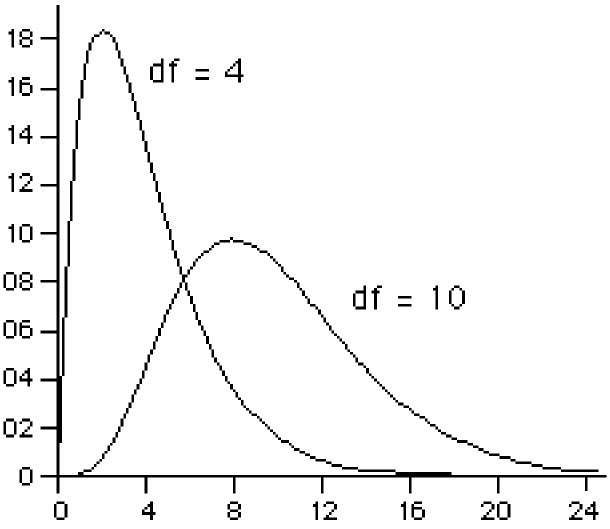

Figure 3.5: The χ^2 distribution, showing the impact of different degrees of freedom.

hypothesis is rejected. The critical value for any hypothesis test depends on the *significance level* at which the test is carried out. The significance level of a statistical hypothesis test is a fixed probability of wrongly rejecting H0. We want to make the significance level as small as possible in order to protect the null hypothesis and to prevent the result from inadvertently making false claims. The significance level is usually denoted by α and chosen to be 0.05. This means that if the value of the χ^2 test statistics is within the tail 5% of the area of the χ^2 *distribution* then H0 is rejected.

As can be seen from the Figure 3.5, the impact from the degree of freedom is the interval in which the hypothesis can be rejected. The higher the degree of freedom, the higher we allow the values of the χ^2 statistic to take and still keep the hypothesis of independence.

The outcome of a χ^2 test as a dependency measure can be the *p-value*. The p-value is the probability of getting a value of the test statistic as extreme as or more extreme than that observed by chance alone, if H0 holds. It is the probability of wrongly rejecting the null hypothesis and is calculated through the *cumulative distribution function* of the χ^2 distribution.

$$CDF(x) = \frac{\gamma\left(\frac{DF}{2}, \frac{x}{2}\right)}{\Gamma\left(\frac{DF}{2}\right)} \tag{3.7}$$

where Γ is gamma function and γ is the incomplete gamma function. For further details see [4, 15, 61].

The p-value is compared with the significance level and, if it is smaller, the result is significant. That is, if the null hypothesis were to be rejected at $\alpha =$

0.05, this would be reported as 'p < 0.05'. Small p-values suggest that the null hypothesis is unlikely to be true. The smaller it is, the more convincing is the rejection of the null hypothesis. It indicates the strength of evidence for say, rejecting the null hypothesis H0, rather than simply concluding 'reject H0' or 'do not reject H0'.

χ^2 as a Dependency Measure

Using χ^2 as a dependency measure makes a lot of sense given the definition of relevance. The p-value of a χ^2 test indicates the strength of a dependency between two attributes and can therefore be adapted as a dependency measure. The p-value is high for weak dependencies and low for strong dependencies and therefore a simple approach to applying it as a dependency measure is to subtract it from 1 which is the upper bound for the p-value of a χ^2 test.

$$DM_{\chi^2}(X_i, X_j) = 1 - p\_val(X_i, X_j) \tag{3.8}$$

where p_val is the p-value of a χ^2 test for H0 stating that X_i is independent of X_j.

3.4.6 Predictive Accuracy

The idea of using predictive accuracy is the idea of comparing the probability that the value of one single attribute can be predicted with the probability that it can be predicted given the state of another attribute. Let X be a feature in the data set with the marginal probability distribution $p(x)$. $p(x)$ can be estimated using ML estimation. Knowing this distribution the value of X can be predicted with some accuracy. The simplest way to predict the value of X when $p(x)$ is known is to always guess that X is in its most likely state $x_{most\_likely} \in X$. The accuracy of guessing the state $x_i \in X$ for a given data instance is referred to as the *predictive accuracy* of $p(x)$, denoted $PA(p(x))$.

The idea of using predictive accuracy as a dependency measure among the pair of attributes X_i and X_j is to compare the predictive accuracy of the marginal probability $p(x_i)$ with the predictive accuracy of the conditional probability $p(x_i|x_j)$, namely $PA(p(x_i|x_j))$. If X_i is dependent of X_j there should be an increase in the chance that the state of X_i can be guessed knowing the state of X_j compared to using only $p(x_i)$ when guessing the state of X_i.

To express this more formally, we let the predictive accuracy of a probability distribution $p(x_i)$, namely $PA(p(x_i))$, be the probability that the state of X_i can be guessed by guessing on the state with the highest probability.

$$PA(p(x_i)) = max_j \;\; p(x_j) \tag{3.9}$$

In the conditional case $p(x_i|x_j)$ we express $PA(p(x_i|x_j))$ as the probability that we can correctly guess the state of X_i for a given case when knowing the state of X_j. In order to do this we compute the conditional probability table for $p(x_i|x_j)$ and use it as a map in which to look up the probability of X_i in a certain state given the state of X_j. Similar to $PA(p(x_i))$ we guess on the most probable state of X_i given the conditional probability table.

Predictive Accuracy as a Dependency Measure

To measure the dependency between a pair of features using predictive accuracy we simply measure the change in predictive accuracy between $PA(p(x_i))$ and $PA(p(x_i|x_j))$. We want a score to be near to 1 if X_j has influence on X_i and a lower value if X_i is not influenced by X_j. This is done using the formula below.

$$DM_{PA}(x_i, x_j) = 1 - \frac{PA(p(x_i))}{PA(p(x_i|x_j))} \tag{3.10}$$

This equation is the base of the proposed dependency measure using predictive accuracy. However this measure is not symmetric in that $DM_{PA}(x_i, x_j) \neq DM_{PA}(x_j, x_i)$. This constitutes a problem since we want our three relevance measures to be symmetric. A simple solution to this problem is using the following equation derived from Equation 3.10.

$$DM_{PA}(x_i, x_j) = 1 - \frac{\frac{PA(p(x_i))}{PA(p(x_i|x_j))} + \frac{PA(p(x_j))}{PA(p(x_j|x_i))}}{2} \tag{3.11}$$

3.4.7 Information Gain

Information gain (or *mutual information* (MI)) is an entropy based measure known from classification by decision trees to rank the attributes according to importance [50]. Here we clarify how *entropy* and information gain can be used as a measure of dependency for attributes in unsupervised learning. We denote the entropy of an attribute X, $H(X)$.

$$H(X) = -\sum_{x_i \in X} p(x_i) \;\; log_2 \;\; p(x_i) \tag{3.12}$$

where $p(x_i)$ is the probability of X being in the state x_i. The value $H(X)$ is a real number between 0 and the binary logarithm of the number of states of X, which measures the purity of the data. The entropy is 0 if the probability that X is in a given state is 1 and the entropy is the binary logarithm of the number of states in X if and only if the probability of X being in a given state is the same for all states x_i.

MI is a measure of the difference between the marginal and the conditional case. In other words, MI is the reduction in entropy for an attribute X_i caused by partitioning the examples according to another particular attribute X_j. We measure the MI achieved from attribute X_j as the change in entropy between $H(X_i)$ and $H(X_i|X_j)$, the *conditional entropy* of X_i given X_j. For any fixed value x_j of X_j, we obtain the conditional probability $p(X_i|x_j)$ and calculate $H(X_i|x_j)$.

$$H(X_i|x_j) = -\sum_{x_i \in X_i} p(x_i|x_j) \;\; log_2 \; p(x_i|x_j) \tag{3.13}$$

We obtain the conditional entropy $H(X_i|X_j)$ by weighting the entropies $H(X_i|x_j)$ with the prior probabilities $p(x_j)$. Conditional entropy is defined as:

$$H(X_i|X_j) = \sum_{x_j \in X_j} p(x_j) H(X_i|x_j) \tag{3.14}$$

Then information gain is given as

$$MI(X_i, X_j) = H(X_i) \quad - \quad H(X_i|X_j) \tag{3.15}$$

Note that X_j might as well be a vector of attributes making $MI(X_i, X_j)$ a measure of difference in data purity within attribute X_i and in each set of attributes conditioned by X_j.

If the value of $MI(X_i, X_j)$ is significantly high it indicates that the purity of X_i increases when the state of X_i is known. In other words, it indicates that X_j can be used to improve the prediction of X_i.

Information Gain as a Dependency Measure

Results of performing information gain on two attributes give an indication of the dependency between them and the strength of such a dependency. This result can be directly applied as a dependency measure for the proposals in this report.

$$DM_{MI}(X_i, X_j) = MI(X_i, X_j) \tag{3.16}$$

3.5 Scoring the Relevance of Features

In this section we aim to show how the dependency measures can be used to score a single feature with respect to its dependencies to the rest of the feature set (referred to as a *score method*). We say that a feature is relevant if it is dependent on another feature. This is expressed formally in the following definition:

$$X_i \in Relevant \Leftrightarrow \exists X_j | X_i depends \, on \, X_j$$

That is, if for a variable X_i, we are able to identify a variable X_j, which depends on X_i, then both X_i and X_j are relevant for the purpose of induction.

In this proposal we distinguish between relevant and *most relevant*. We maintain our definition of relevance and use it to develop a score method which can be applied to score a single feature. Given such a method we are able to identify both features with high relevance and features with low relevance. Additionally we are able to rank each feature and select only the most relevant based on a threshold which will be described later. We propose 2 methods for scoring the relevance of a feature.

$$\begin{aligned} R_{max}(X_i) &= maxDM(X_i, X_j) & (3.17) \\ R_{avg}(X_i) &= \frac{\sum_j DM(X_i, X_j)}{p}. & (3.18) \end{aligned}$$

Where p is the number of features in the data set and DM is one of the above dependency measures DM_{chi}, DM_{PA} or DM_{MI}. In the remainder of the report we will refer to a *score measure* as a measure that uses either of the 2 score methods with any of the 3 dependency measures. The result of calculating the score of a given feature using a score measure is denoted a *relevance score*. In total that leaves 6 relevance scores available for testing.

Using a maximum scoring method on the dependency means that the score of a given feature will be the strongest dependency of the feature. Using such a scoring scheme assumes that random dependencies are weak and that features with many dependencies have a higher probability of having very strong dependencies. The main property of this approach is that it will reward strong dependencies rather than many dependencies, meaning that a feature can be dependent of only one other feature and still have a higher score than a feature with many dependent features.

The second score measure is an average over dependencies to all features. The scoring method sums up the values of the feature dependency measure between the tested feature and all the rest of the features. The average is over the total amount of features in the data set.

3.6 Thresholding

The current proposals assign scores to each feature in the data set. In order to do unsupervised FSS it is necessary to set a threshold that is able to effectively cut away all irrelevant features based on their scores.

3.6.1 Learning Curve Sampling Method

Here we propose a new scheme based on the learning-curve sampling method proposed by Meek et al. [48].

Given a set of features $\boldsymbol{X}$, let $S_1, S_2, .., S_p \subseteq \boldsymbol{X}$ denote the feature subsets, constrained by the relevance ranking, that are to be examined in the process of finding the appropriate threshold. We require that $S_i \subset S_{i+1}$, meaning that the subsets are nested. A given subset S_i contains the i features in $\boldsymbol{X}$ with the highest relevance score. The subsets S_i and S_{i+1} differ only in a single feature.

Utility

The main idea is to keep adding features (i.e. moving from S_i to S_{i+1}) as long as the benefit is greater than the cost. At stage i there are 2 choices available. Either stop and output the current feature subset or add a new feature and examine the new feature subset. In order to evaluate a feature subset properly one has to consider both benefit and cost. At step i of incrementation we express the utility of subset S_i as:

$$Utility(S_i) = Benefit(S_i) - Cost(S_i) \tag{3.19}$$

In order to calculate the utility at each stage we need to define the functions for benefit and cost. The benefit of S_i can be defined as the sum of the relevance scores for each feature j in S_i.

$$Benefit(S_i) = \sum_{j=1}^{i} R_j \tag{3.20}$$

Defining the benefit in this way will have several consequences that should be considered. The benefit of adding a feature to the subset is evaluated with respect to the relevance of the feature itself and not with respect to the relevance of the new subset. The impacts are that a given feature is likely to be overrated rendering the scheme conservative in the selection. In addition redundancy is not detected since several features that contribute with approximately the same information will all have the same relevance.

The cost according to [48] is defined as the running time used to obtain the current benefit. We focus on interpretability and knowledge gain, therefore the cost increases with the addition of attributes as this reduces the interpretability of the induced model. Therefore the cost is proportional to the number of features in the current subset and can be defined as:

$$Cost(S_i) = i * \alpha \tag{3.21}$$

where α is the relative importance of the number of attributes to the benefit. The value of α should be assessed by the end-user since it is problem dependent.

Stopping Criterion

As mentioned in the previous section at stage i we can either choose to stop and output the current subset, or continue to stage $i+1$. Meek et al. [48] propose to check the utility at stage i against the expected utility at stage $i+1$. We adapt this method to this thresholding scheme and define our stopping criterion as follows. We stop and output the feature subset at stage i if:

$$Utility(S_{i+1}) \leq Utility(S_i) \tag{3.22}$$

Using the previous definitions of both utility, cost and benefit Equation 3.22 can be rewritten as:

$$\begin{aligned} Benefit(S_{i+1}) - Cost(S_{i+1}) &\leq Benefit(S_i) - Cost(S_i) \\ Benefit(S_{i+1}) - Benefit(S_i) &\leq Cost(S_{i+1}) - Cost(S_i) \qquad (3.23) \end{aligned}$$

From Equation 3.21 we see that:

$$Benefit(S_{i+1}) - Benefit(S_i) \leq \alpha((i+1) - i) \tag{3.24}$$

Therefore the stopping criterion can be redefined as:

$$\frac{Benefit(S_{i+1}) - Benefit(S_i)}{((i+1) - i)} \leq \alpha \tag{3.25}$$

In this equation α is chosen to reflect how many attributes the user is willing to add in order to increase the relative benefit a certain amount. Formally we can state that α is the ratio of increase in the relative benefit to the number of attributes added from S_i to S_{i+1}. If the stopping criterion is met we go back to stage i and output the subset, otherwise we continue to stage $i+1$.

Figure 3.6 shows an example of an output of a filter method including both relevant and irrelevant features. The y-axis denotes the benefit at stage i and the x-axis denotes the first i features, given the relevance ranking, in the current subset. The graphical illustration indicates how to define α.

Although the strategy is myopic, it is optimal in the case where it is guaranteed that the benefit-increase will decrease and the cost increases as a consequence of incrementing the feature subset. In this scenario it makes sense to stop when the ratio of these two quantities falls below α. In this proposal the ranking ensures that the shape of the curve is concave, leaving the strategy to be optimal.

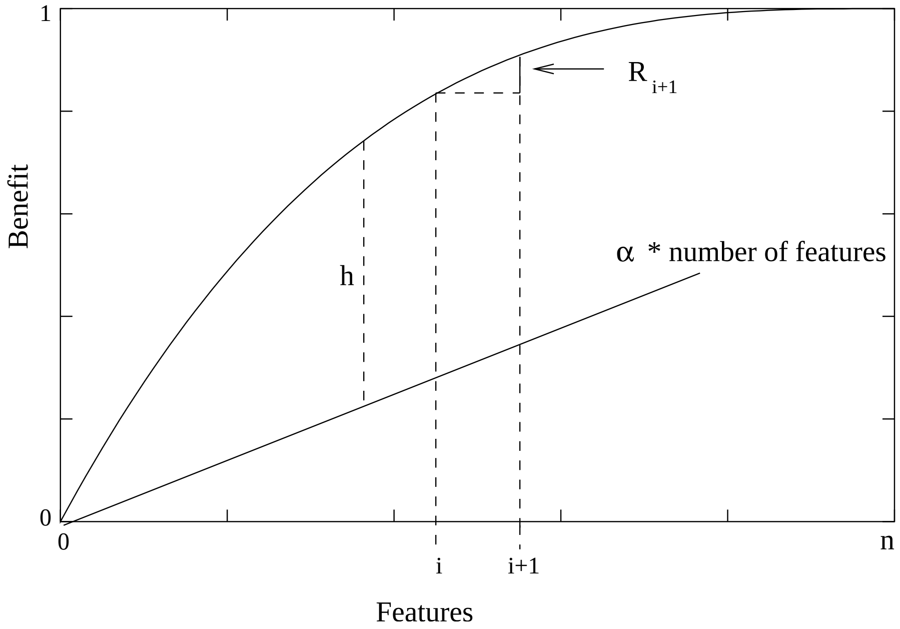

Figure 3.6: Example, plotting benefit on the y-axis. Relevance at stage $i + 1$ is the increasing benefit between stage i and $i + 1$. The impact of α shown as the coefficient for the line for which maximizing the vertical distance h defines the stopping criterion.

3.6.2 Hypothesis Testing

In a second approach for setting a threshold we want to propose the use of hypothesis tests [15]. As described in Section 3.4.5 a hypothesis test is an approach for testing whether to keep or reject a claim, namely the null-hypothesis denoted H0. If we regard feature selection as a hypothesis test we choose as null hypothesis the claim: "*feature X is irrelevant*", then we can reject H0 if the evidence against it is sufficiently strong for a given relevance measure, or we can keep it in case the evidence against it is not sufficiently strong. The alternative hypothesis H1, which is favored if H0 is rejected is the claim that "*feature X is relevant*". However the conclusion of a hypothesis test is merely to keep or reject H0.

Setting up a Hypothesis Test

The core in a hypothesis test is a test statistic. An obvious choice for a test statistic is one of the proposed score measures. Before we can use this test statistic to draw conclusions about H0 we need to know the density distribution for each of the proposed score measures under the null hypothesis. That is, we need to know what scores we can expect for an irrelevant feature in a given data set. For instance, if the scores for an irrelevant feature are normal distributed this knowledge could be a mean and a standard deviation. The hypothesis test is then to test how strongly we can believe the score of a certain feature to be among the irrelevant ones. The problem here is that we do not know the distribution of any of the proposed relevance measures. The exact distribution for those scores can be found (in theory) by sampling infinitely many examples for which H0 is true relative to a given feature set. In practice we can find an approximation to the distribution for each of the relevance measures under H0 by sampling a high number of irrelevant features.

Setting a Threshold

To test whether H0 holds for a given feature we calculate its test statistic and reject H0 if it exceeds a certain threshold, namely the critical value. The critical value is derived from the density distribution together with a decision of how high a risk with which we can accept to wrongly reject H0 if it is true. This threshold is set up with the help of the significance level. In theory the choice of a significance level is up to the user of the hypothesis test as it reflects the chance of making a wrong decision. Therefore we denote the significance level α which, like in the learning curve approach, may impact the amount of features which will be deemed irrelevant. However, a standard value for the significance level is 0.05 (5%) which we will use in our tests. From the significance level we find the set of values of the test statistic for which the null hypothesis is rejected in a hypothesis test. In our case this is set to the 5% highest values of the test statistic for the sampled irrelevant features under H0. The conclusion of our hypothesis test is that we reject H0 if for a given case the result of the test statistic is in this set. Therefore the critical value for a hypothesis test is the lowest possible value in this set. Knowing this we can accept or reject H0 for any given feature X.

Approximating the Density Distribution

To make the approximation of the density distribution of the test statistic we sample relevance scores for irrelevant features. Each sample score is generated by scoring a randomly generated feature X_{random} relative to the original feature set $\boldsymbol{X}$, i.e. $R(X_{random})$. Each randomly generated feature is sampled by filling in each value at random, maintaining the same number of states. We produce 10000 samples and sort them in increasing order. The sample set of 10000 cases is a set of values of the test statistic for which H0 is true.

We want to avoid any bias introduced by the different number of states in the tested feature and the randomly generated features which were used to produce the sample set. Therefore if we want to use a hypothesis test to test whether a feature X is relevant and the number of states in (or the *cardinality* of) X is q, we approximate the distribution of the test statistic using only features with the cardinality q.

Unlike with the learning curve sampling method approach, the advantage of this approach is that there is a clear interpretation of the value α, namely the risk of making a wrong decision about H0. Moreover, with this approach one can distinguish between relevant and irrelevant features without having to present an ordering of the features.

3.7 Summary

In this chapter we outline the idea of FSS both in supervised and unsupervised learning. We give a definition of relevance in the domain of unsupervised FSS which is based on the dependence between a feature and the cluster random variable. The cluster random variable is unknown but under the assumption that a feature is dependent on this it is likely to be dependent on other relevant features. Therefore the main idea is that a feature cannot be discarded as irrelevant if it is dependent on at least one other feature.

We propose 3 dependency measures in order to measure the dependency between 2 features:

- χ^2 analysis.
- Predictive accuracy.
- Mutual Information

The score of a single features can then be obtained in several ways. In this project we propose to measure the dependency of a feature with each of the other features in the data set. From the result we propose 2 scoring methods, maximum and average:

$$\begin{aligned} R_{max}(X_i) &= max DM(X_i, X_j) \\ R_{avg}(X_i) &= \frac{\sum_j DM(X_i, X_j)}{p} \end{aligned}$$

where p is the number of features in the data set.

In order to perform unsupervised FSS it is necessary to decide on a threshold on which features with a low relevance can be discarded as irrelevant. We propose 2 approaches to setting a threshold. The learning curve sampling method based on a ranking of the features and a cost versus benefit approach, and a hypothesis test based on 10000 randomly generated features used in a test statistics similar to that of χ^2.

4 Results

We know very little, and yet it is astonishing that we know so much, and still more astonishing that so little knowledge can give us so much power.
— Bertrand Russell

In this chapter we will present the results of applying the proposals to several data sets. First we will present a description of 3 synthetic data sets and 4 real-world data sets which will be the base for testing the methods. In Section 4.2 we then present the results of applying the proposed score measures to the described data. By showing the performance on a large variety of data sets we aim to show that the proposals perform well and to show the limitations that exist for this approach to unsupervised FSS.

Thereafter we will explain how the clustering techniques described in Chapter 2 have been applied in order to validate the results of the filter methods. Last we present the results of validating the filter results. Discussions on the results leads to further extensions of the methods proposed and Chapter 5 gives a description of a hybrid approach to unsupervised FSS in which the results in this chapter have been applied.

4.1 Data Description

In this section we will describe the 7 data sets that will be used to test and evaluate the unsupervised FSS methods proposed in this report. The data sets consist of 3 synthetic data bases and 4 real-world data sets. We first present the synthetic data sets and then the real-world data sets.

4.1.1 Sampling of Bayesian Networks

The first two synthetic data sets are based on a Bayesian Network (BN) created by Peña et al. [56]. The BN can be seen in Figure 4.1 and contains a cluster

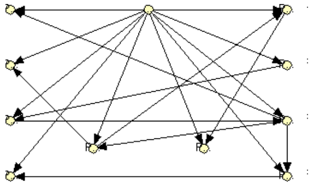

Figure 4.1: Original BN, all nodes are considered relevant for induction.

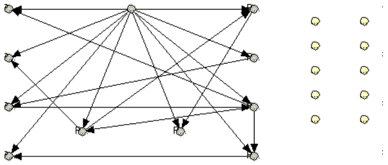

Figure 4.2: Original BN with 10 added nodes. The 10 additions are unconnected and considered irrelevant for induction purposes.

random variable with 3 states and 10 nodes of each 3 states with varying interconnectivity and all children of the cluster random variable. All 10 features in the model are relevant. We add 10 irrelevant, and therefore unconnected nodes to the network as shown in Figure 4.2. The 10 unconnected nodes also contain 3 states and each instance has been randomly generated from a specified probability distribution, chosen at random. However including the constraint that the probability distributions for each of the randomly generated features never exceed 80% nor go below 20% for any state. From the model shown in Figure 4.2 we sample 10000 cases that will be used as the first synthetic test data denoted SYN10.

A second synthetic data set has been derived using the same base BN model, but now adding 20 unconnected nodes using the same technique as for SYN10. Using this BN we have again sampled 10000 cases. We denote this data set SYN20. For both SYN10 and SYN20 we have removed the cluster random variable. The reason for creating an additional data set with the same properties, except in the number of irrelevant features, is to show how the score measures will react to an addition of irrelevant features.

As mentioned in both synthetic data sets the class random variable has 3 states and so we cluster the data set using $k = 3$.

4.1.2 Waveform

The last source of artificial data is a well known data base from the UCI repository of Machine Learning databases [7], which will be referred to as WAVE. The data consist of 40 features whereof the last 19 are noise. The data represents continuous values based on 3 generated waves over separate series of the first 21 features. Since we only consider categorical data the data set has been discretized into 3 categories. The discretization technique used is a basic method which divides the value of each feature into 3 equally sized bins and places all instances in their corresponding value interval [12].

We know that there exist 3 clusters in the data, each representing a combination of 2 waves. The first 4 and the last 4 of the 21 relevant features have been discovered as being less significant than the others, thus in some papers [8, 65] these features are considered irrelevant. Table 4.1 gives an overview of relevant and irrelevant features in the WAVE data set.

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39

Table 4.1: Overview of the WAVE data set. Relevant features are marked in bold font.

4.1.3 The Insurance Company Case

This data set is the first of the real-world data sets in our evaluation. It contains 5822 customer records kept by an insurance company that sells mobile home policies. It is known as the COIL data set. Each record consists of 85 features, containing sociodemographic data and product ownership.

An 86th feature contains the class random variable describing those who buy a mobile home insurance policy and those who do not. The data set was part of a data mining competition referred to as “the CoIL Challenge 2000” [69]. From the documentation of the results of the competition we gain some insight in what others before has gained from this data set and we can compare the results of this project to the descriptions.

One of the entries in the data description competition performs statistical analysis on the data using χ^2 in order to test dependency with the class random variable. From the statistical analysis they found 21 features within the 95% confidence level. These are shown in Table 4.2. The approach is within the domain of supervised learning which makes comparison difficult to perform.

46 58 67 4 41 42 36 17 43 33 29 30 24 31 38 15 0 64 9 11 28

Table 4.2: The 21 features that proved the best subset according to a χ^2 test performed by Kim et al. [38]. The features are ranked from best to worst.

4.1.4 Leukemia

The last real-world data set has been chosen for its extreme structure which perhaps will be able to test some of the limitations of the methods that we propose. The data set consists of 7129 features and 72 cases and will be referred to as LEUKEMIA. Each case represents a patient suffering from leukemia and the features describe gene expression level for each of the patients. The data was first introduced by Golub et al. [26]. It is well known in data mining communities and has been thoroughly analyzed in the past. From [26] we know that there are 2 clusters partitioning patients with respect to the type of leukemia that they are suffering from (AML and ALL). In the article by Golub et al. they build a predictive model using only 50 of the 7129 features based on supervised FSS in a filter approach. The resulting model obtained accurately classified 36 out of 38 patients in the test set as either type AML or ALL (the last two were classified as uncertain). This indicates a very high amount of irrelevant or redundant features which should be detected using the methods proposed in this project.

As variations of this data set 2 additional data sets have been derived using the two types of leukemia. First we have transformed the data into a new data base where each feature represents a patient and each instance represents a gene. Then we split the data into 2 separate data sets, one for each of the two types of leukemia. The first data set, denoted `AML`, contains 25 features (patients) all suffering from leukemia of type AML and 7129 cases (genes). The second data set, denoted `ALL`, contains the remaining 47 features and also 7129 cases. Due to the fact that the tables are separated given the respective type of leukemia we expect all features to be relevant (an irrelevant feature indicates that this patient has little genetically in common with the rest of the patients despite suffering from the same illness). In the `LEUKEMIA` data set there are 2 clusters (2 types of leukemia) whereas in the `AML` and `ALL` data sets there are 3 clusters (overexpressed, underexpressed and neutral genes).

4.2 Filter Results

Based on the proposals presented in Chapter 3 and the data description in Section 4.1 this section presents the results of applying the proposed relevance scores in the filter method to the 7 data sets. The order of appearance corresponds to the order of the data descriptions.

All graphs shown in this section have been normalized between 0 and 1 on both x-axis and y-axis for the purpose of applying the learning curve thresholding scheme (see Section 3.6.1). The normalization technique used is not unimportant as it can have an impact on the shape of the curve. For this project a simple normalization technique called *linear scaling* has been used. The method produces a linear relationship between instance values and normalized values. and all information is preserved and can be restored from the normalized results [58]. Equation 4.1 shows how to compute a normalized value for each instance value of an attribute.

$$x_{norm_i} = \frac{x_i - min(x_1, \ldots, x_n)}{max(x_1, \ldots, x_n) - min(x_1, \ldots, x_n)} \tag{4.1}$$

where x_{norm_i} represents the normalized value that the attribute X takes in the ith case of the database, x_i being the original value of that case.

When applying the learning curve thresholding scheme we have chosen 2 values for α, 0.3 and 0.7. For hypothesis testing we use the standard significance level, 0.05.

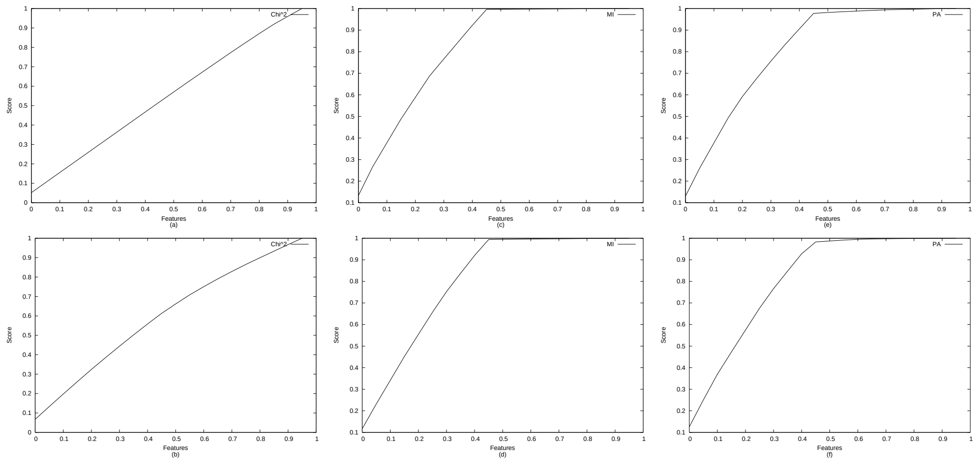

Figure 4.3: Filter results for SYN10. The top row of figures show the 3 filter methods using max scores whereas the bottom row show average scores. The first column describes χ^2, the middle MI and the right PA. Notice that the difference between the two rows is insignificant.

4.2.1 The BN Sampled Data

Figure 4.3 depicts the results of the learning curve approach applying all 6 score measures, using both maximum and average scores, to SYN10. The top row shows the results using the maximum score for each feature whereas the bottom row shows the average scores. It is clear to see that for this data set the distinction between max and average scores is surprisingly small and insignificant. MI and PA expectedly show very similar results and the distinction between relevant and irrelevant is clear for both measures even without the use of any thresholding scheme. According to PA and MI the cut point is at 0.45 features which represents 10 features. Since we constructed the data with 10 nodes directly connected to the cluster random variable this is what we expected. Examining the results for χ^2 we surprisingly observe only very weak distinction between relevant and irrelevant features.

An explanation for the weak distinction may be in the fact that generally the p-values for irrelevant features are only a fraction higher than for relevant features. In this case we should expect the rankings to be correct in which case χ^2 would still be applicable for ranking the features but not directly for unsupervised FSS.

The rankings of the features can be seen in Table 4.3 on the following page and 4.4 using learning curve thresholds and hypothesis testing respectively. All score measures rank the truly relevant features correctly even though χ^2 identifies only few irrelevant features given learning curve thresholding using $\alpha = 0.7$. Notice also that the disadvantage of the maximum score method is clearly shown for χ^2 in that all features have been deemed relevant given this score measure.

Method	$\alpha = 0.7$	$\alpha = 0.3$	Irrelevant
MI (avg)	**16 11 13 17 14 19 10 18 12 15**		5 1 0 6 8 3 7 4 9 2
MI (max)	**13 16 11 17 14 10 12 18 15 19**		5 0 7 3 1 8 6 4 9 2
PA (avg)	**13 16 10 19 18 14 12 17 11 15**		1 6 7 4 5 2 8 0 3 9
PA (max)	**16 13 10 14 12 19 18 15 17 11**		1 6 4 7 5 2 3 8 0 9
χ^2 (avg)	**12 19 15 18 10 16 11 14 13 17** 5 1 0 6 8	3 7 4 9 2	
χ^2 (max)	**12 19 15 18 10 16 11 14 13 17** 5 1 0 6 8 3 7 4 9 2		

Table 4.3: Ranking order (best to worst) of the features in the SYN10 data set. Relevance based on the learning curve thresholding scheme using both $\alpha = 0.3$ and $\alpha = 0.7$. The truly relevant are marked in bold font.

It is interesting to notice the significant differences in the rankings that the score measures produce and how the graphs can exhibit such strong similarity despite the dissimilarity in the rankings. Clustering validation on these rankings will show whether it is just the dependency among relevant features in this data set that are so close that the ordering of relevant features become insignificant and easily altered depending on the method used, or whether one or more methods do not rank the features correctly with respect to clustering. From the graphs in Figure 4.3 it can be seen that the line is almost straight from 0 and up to the last relevant features. This indicates that the score of the features are close to equal and leads us to believe that this explains the differences in the ordering.

Method	Relevant	Irrelevant
MI (avg)	**16 11 13 17 14 19 10 18 12 15**	5 1 0 6 8 3 7 4 9 2
MI (max)	**13 16 11 17 14 10 12 18 15 19**	5 0 7 3 1 8 6 4 9 2
PA (avg)	**13 16 10 19 18 14 12 17 11 15**	1 6 7 4 5 2 8 0 3 9
PA (max)	**16 13 10 14 12 19 18 15 17 11**	1 6 4 7 5 2 3 8 0 9
χ^2 (avg)	**12 19 15 18 10 16 11 14 13 17** 5 1 0 6 8 3 7 4 9 2	
χ^2 (max)	**12 19 15 18 10 16 11 14 13 17** 5 1 0 6 8 3 7 4 9 2	

Table 4.4: Ranking order (best to worst) of the features in the SYN10 data set. Relevance based on the hypothesis test thresholding scheme using significance level α= 0.05. The truly relevant are marked in bold font.

To the SYN10 data set we have also applied the hypothesis test with $\alpha = 0.05$. Figure 4.4 depicts the distributions for each of the 6 score measures derived from sampling 10000 irrelevant features and scoring them relative to the original data. The dashed line denotes the critical values for each method. In order for this method to be reliable the curve needs to flatten out before the critical value. The more flat the curve is the more likely are we to believe in our decision to reject H0. From the curves it can be seen that the χ^2 square measure combined with the maximum relevance score is not very reliable. This can be explained by

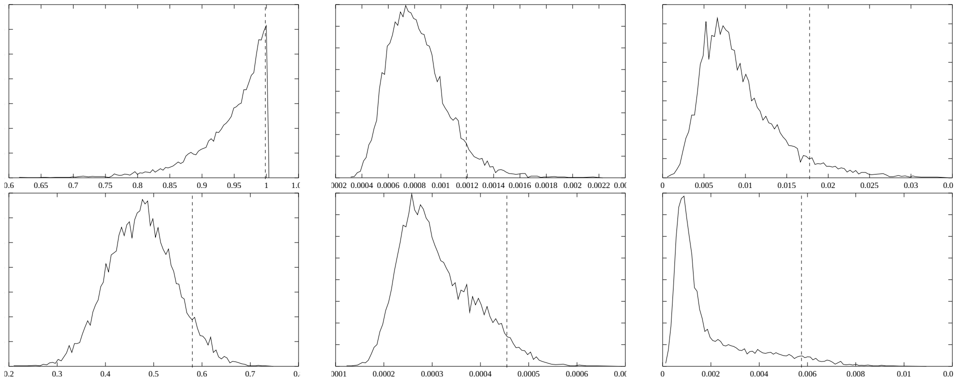

Figure 4.4: The distributions of each of the score measures applied to random features scored against the SYN10 data set. The dashed lines show the critical values. The columns from left to right are for χ^2, MI and PA respectively while the top row is for maximum score measures and the bottom row is the average score measures.

the fact that many of the randomly generated features had at least one strong dependency with one of the features in the original data set, according to the χ^2 dependency measure. It is worth to mention that the 6 graphs in Figure 4.4 are generated from the exact same sample of randomly generated features. Therefore the difference in the curves allows us to conclude that at least the χ^2 dependency measure combined with the maximum relevance score method is not very reliable. On the other hand the curves have a tendency to be flatter for the average approach for all three dependency measures with PA exhibiting the most flat shape.

The fact that all the features have the same number of states allows us to use the ordering when performing the filtering. Therefore, in stead of comparing each feature to the critical value derived from the sample sets, we benefit from the ranking and declare the features which has scored less than the critical value irrelevant. From Table 4.4 it can be seen that with PA and MI each with both average and maximum scores has successfully filtered out all the irrelevant features. The χ^2 method however has declared all the 20 features relevant and has not been able to detect any irrelevant features despite half of the features are truly irrelevant according to the true structure of the model which has generated the data. The performance of the methods used in the hypothesis test corresponds to the result of the learning curve where both the PA and MI measures are significantly better than the χ^2 measures in filtering out irrelevant features.

Figure 4.5 on the next page shows the results of applying the 6 measures to the SYN20 data set. As with the first graphs the top row represents the results using maximum scores whereas the bottom row represents the results using average scores. For these results we again observe a clear distinction between relevant and irrelevant features. According to MI and PA in Tables 4.5 and 4.6 only the

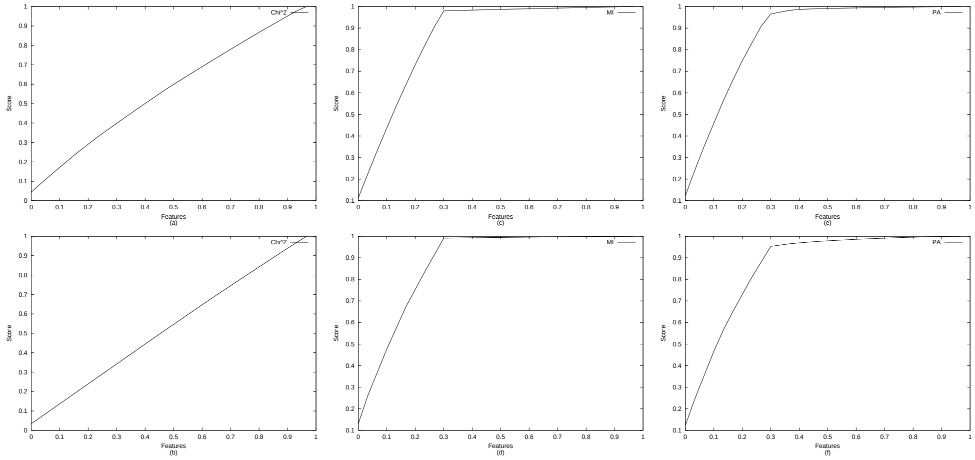

Figure 4.5: Filter results for SYN20. The top row of figures show the 3 score methods using max scores whereas the bottom row show average scores. The first column describes χ^2, the middle MI and the right PA. Notice again that the differences are insignificant.

10 truly relevant features are detected as relevant. This corresponds very well with the results of the smaller data set, which seems to indicate that the amount of irrelevant features does not have an impact on the distinctiveness between relevant and irrelevant features even by averaging scores over all features.

Method	$\alpha = 0.7$	$\alpha = 0.3$	Irrelevant
MI (avg)	**26 23 27 21 20 24 25 28 22 29**		14 15 8 3 7 1 12 2 11 18 9 0 19 5 6 4 16 10 13 17
MI (max)	**26 23 21 27 29 24 20 28 22 25**		7 15 1 5 12 14 3 0 4 2 11 8 16 6 17 18 9 19 10 13
PA (avg)	**26 23 20 24 22 29 25 28 27 21**		19 5 14 1 18 13 0 10 8 11 17 16 15 9
PA (max)	**23 26 20 28 29 24 22 21 27 25**		11 16 17 14 15 4 9 2 5 6 19 3 1 12 18 8 7 13 0 10
χ^2 (avg)	**29 26 20 22 23 27 21 24** 7 **25** 12 1 5 **28** 15 18 3 2 11 17 14 0 4 16 6 8 19 9 10	13	
χ^2 (max)	**29 26 20 22 23 27 21 24** 7 **25** 12 1 5 **28** 15 18 3 2 11 17 14 0 4 16 6 8 19 9 10 13		

Table 4.5: Ranking order (best to worst) of the features in the SYN20 data set. Relevance based on the learning curve thresholding scheme. The truly relevant are marked in bold font.

By examining the rankings as shown in Tables 4.5 and 4.6 it can be seen that they contain the same characteristics as the first data set. The truly relevant features have been detected using both MI and PA whereas χ^2 ranks the truly

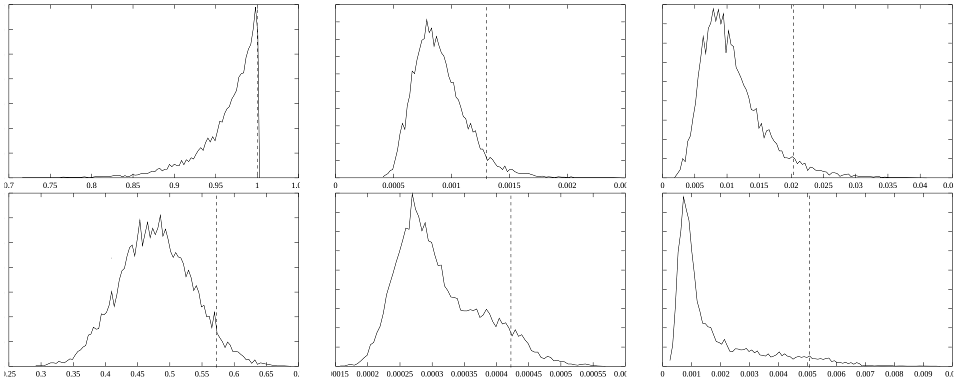

Figure 4.6: The distributions of each of the score measures applied to random features scored against the SYN20 data set. The dashed lines show the critical values. The columns from left to right are for χ^2, MI and PA respectively while the top row is for maximum score measures and the bottom row is the average score measures.

relevant features highest although the scores make the thresholding techniques unable to distinguish relevant from irrelevant. The reason for the inaccuracy that is exhibited by χ^2 must be explained by the similarity of the values that features are given whether or not they are dependent. Seemingly this method is more susceptible to noise than the other two.

The hypothesis thresholding scheme was also applied to the SYN20 data set. Again all the features have the same number of states and the ordering was used, instead of comparing each feature to the critical value. The results of performing the hypothesis test on SYN20 are shown in Table 4.6. Again it can be seen that with PA and MI each with both average and maximum scores all the true irrelevant features have been filtered out. With the significance level 0.05, the χ^2 method however has declared all the 30 features relevant. Again these results corresponds very much to the results obtained by the learning curve results. Figure 4.6 depicts the distributions of the score measures when sampling irrelevant features.

Including more irrelevant features increases the chance of random dependencies among irrelevant features which can lead to an irrelevant feature being deemed relevant. In this section we have shown our method capable of handling a number of irrelevant features without consequences for the final features subset.

4.2.2 The Waveform Data

The last of the artificial data sets is the WAVE data set which includes both irrelevant and partially relevant features. In Figure 4.7 the results of applying the 3 average score measures are shown. Previous results indicate that using average or maximum score measures does not have an impact on PA or MI, but χ^2 being

Method	Relevant	Irrelevant
MI (avg)	**26 23 27 21 20 24 25 28 22 29**	14 15 8 3 7 1 12 2 11 18 9 0 19 5 6 4 16 10 13 17
MI (max)	**26 23 21 27 29 24 20 28 22 25**	7 15 1 5 12 14 3 0 4 2 11 8 16 6 17 18 9 19 10 13
PA (avg)	**26 23 20 24 22 29 25 28 27 21**	19 5 14 1 18 13 0 10 8 11 17 16 15 9
PA (max)	**23 26 20 28 29 24 22 21 27 25**	11 16 17 14 15 4 9 2 5 6 19 3 1 12 18 8 7 13 0 10
χ^2 (avg)	**29 26 20 22 23 27 21 24** 7 **25** 12 1 5 **28** 15 18 3 2 11 17 14 0 4 13 16 6 8 19 9 10	
χ^2 (max)	**29 26 20 22 23 27 21 24** 7 **25** 12 1 5 **28** 15 18 3 2 11 17 14 0 4 16 6 8 19 9 10 13	

Table 4.6: Ranking order (best to worst) of the features in the SYN20 data set. Relevance based on the hypothesis test thresholding scheme using significance level α= 0.05. The truly relevant are marked in bold font.

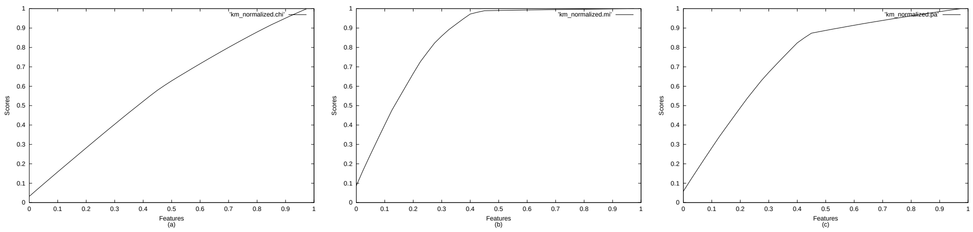

Figure 4.7: Filter results for the WAVE data set using average scores. The leftmost describes χ^2, the middle MI and the right PA.

more susceptible to noise performs poorly given maximum scores. Therefore we have chosen only to show average results for the remaining data sets. In this data set though we can see a significant difference in the curves representing PA and MI. MI indicates a more clear distinction between relevant and irrelevant features whereas PA maintain that several more features contribute although the rate of contribution is questionably small.

From previous analysis and knowledge on the construction of the data we know that the features 21-39 are noise and can be considered irrelevant features (see Table 4.1). In addition analysis has shown the features 0-3 and 17-20 maintain only little relevance. In Tables 4.7 and 4.8 it is clear that all features that are considered noise have been ranked last regardless of the method used. The only exception to this is feature 0 which has been ranked very low. All 3 score measures agree on this property.

Closer examination shows that PA and MI rankings are surprisingly similar considering the differences in the graphs. Both agree on a ranking where the features 0, 1, 19 and 20 also rank lower than any of the known relevant. This corresponds with what we know about the data already. The features 2, 3, 17 and 18 although also deemed irrelevant by previous analysis are still minor relevant and the structure of the data states that these 4 features are the most relevant of the minor relevant features. Applying the thresholds we obtain very similar results where only the features 0 and 20 of the truly relevant have been detected as irrelevant. Further comparisons with previous analysis is not applicable due to the discretization that has been performed prior to applying the 3 score measures on the data.

According to χ^2 no features are deemed irrelevant although the ranking match the results given by PA and MI and previous analysis of the data.

	$\alpha = 0.7$	$\alpha = 0.3$	Irrelevant
MI (avg)	**6 14 7 13 15 5 4 16 12 8 3 17 11 9 2 18 10 19**	**1**	29 24 34 39 33 21 25 22 28 **20** 32 37 **0** 36 26 27 30 35 38 23 31
PA (avg)	**6 14 7 13 15 5 12 16 4 8 3 17 11 18 9 2 10 19 1**		24 33 29 39 37 32 27 22 28 **0** 34 36 35 **20** 25 21 26 38 30 31 23
χ^2 (avg)	**6 12 18 14 19 5 8 3 1 17 15 11 16 13 9 10 4 7 2** 29 24 21 39 32 25 34 33 28 **20** 22 37 26 **0** 36	27 30 35 38 31 23	

Table 4.7: Ranking order (best to worst) of the features in the WAVE data set. Learning curve thresholding scheme used. Truly relevant features based on previous analysis have been marked with bold font.

	Relevant	Irrelevant
MI (avg)	**6 14 7 13 15 5 4 16 12 8 3 17 11 9 2 18 10 19 1**	29 24 34 39 33 21 25 22 28 **20** 32 37 **0** 36 26 27 30 35 38 23 31
PA (avg)	**6 14 7 13 15 5 12 16 4 8 3 17 11 18 9 2 10 19 1** 24 33 29 39 37 32 27	22 28 **0** 34 36 35 **20** 25 21 26 38 30 31 23
χ^2 (avg)	**6 12 18 14 19 5 8 3 1 17 15 11 16 13 9 10 4 7 2** 29 24 21 39 32 25 34 33 28 **20** 22 37 26 **0** 36 27 30 35 38 31 23	

Table 4.8: Ranking order (best to worst) of the features in the WAVE data set. Hypothesis test thresholding scheme used. Truly relevant features based on previous analysis have been marked with bold font.

Table 4.8 shows the result of applying the hypothesis test to the WAVE data set using PA, MI and χ^2 average scores with significance level 0.05. Furthermore the distributions for the 3 tested score measures are in Appendix A. Again we allow ourselves to benefit from the ranking when distinguishing between relevant and irrelevant features instead of comparing each feature to the critical value. MI has declared 19 features relevant all features which are relevant according to our data set while only two of the features, namely 0 and 20, which are relevant according to previous analysis have been declared irrelevant. PA has declared 25 of the original 40 features relevant with the same two relevant features left out. At last, the hypothesis test with χ^2 has not deemed any of the features irrelevant.

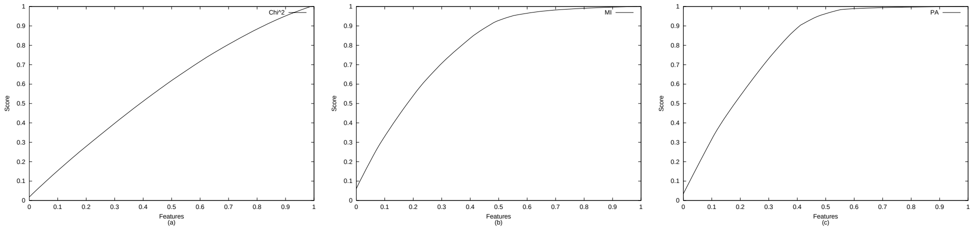

Figure 4.8: Filter results for the COIL data set using average scores. The leftmost describes χ^2, the middle MI and the right PA.

Despite that the true distinction for this data set was made on a version of the data which was not discretized both MI and PA has performed nicely on this data set. It is clear that discretization of a data set can have a large impact on the models which can be learned from the data as well as the features which are relevant for clustering.

4.2.3 The CoIL Challenge

We have shown that the proposed score measures work on 3 constructed artificial data sets which proves that the methods work in theory. We compare the results that we obtain using these 3 methods with the results obtained by [38]. Figure 4.8 shows the graphs for the 3 methods using average scores.

Method	$\alpha = 0.7$	$\alpha = 0.3$	Irrelevant
MI (avg)	**0 4 29 30 35** 34 **17 42 24** 18 **9** **11 41** 27 **15** 12 **33 36 38** 14 22 23 16 21 **28 31** 6 37 25 26 2 8 32 20 13 39 10 19 5 3 7 **58** 40 79	**43 64** **46 67** 1 54 51	53 72 75 74 45 66 48 44 69 63 82 62 47 52 50 61 56 68 55 65 60 84 73 71 77 57 83 81 78 70 76 49 59 80
PA (avg)	**27 33 38 30 29 24 17 0 36** 18 22 35 34 14 **41** 12 32 **9** 37 **11** 16 25 **4** 23 26 21 7 **42** 6 **31 15** 8 13 **28** 10 2 39 5 19 20	3 **58** 79 40 **67 43** **64 46**	1 54 53 75 51 74 72 45 44 62 48 63 82 66 69 55 56 47 52 60 50 61 83 84 73 77 68 70 71 57 65 81 76 59 49 78 80
χ^2 (avg)	**58 4 0 17 30 42 29** 27 20 16 35 34 6 18 **38 24 31** 8 66 37 **36** 32 **43** 7 **15 41** 2 21 3 13 **9** 14 23 25 12 **33** 22 19 **28** 26 5 84 51 54 72 **11** 63 **46** 10 **64** 79 39 65 40 1 45 61 75 77 83 69 68 56 **67** 50 44 47 78 71 76 74 57 48 62 53 52 82 81 80 73 55 70 59 49 60		

Table 4.9: Ranking order (best to worst) of the features in the COIL data set. Relevant according to [38] are marked in bold font.

Tables 4.9 and 4.11 show the features divided into relevant and irrelevant features using learning curve thresholding and hypothesis testing respectively. Table 4.9 additionally show the ranking of the features according to the score measure. A noticeable difference in the 2 thresholding schemes is that χ^2 does not provide results that the learning curve is capable of detecting as irrelevant, whereas hypothesis testing detects several irrelevant features given the same scores.

The CoIL data set is the only data set with varying cardinalities of the features. In fact this data set has features with cardinalities from 2 to 10 and a single feature with the cardinality 40. Therefore, for this data set we have produced 3 sample sets (one for each score measure used) for each of the different cardinalities in order to estimate the critical values. That makes a total of 30 sample sets. Table 4.10 depicts the different cardinalities and the corresponding critical values derived from the sample sets. Moreover the distributions for each score measure are depicted in Appendix A.

Cardinality	Features	Threshold (χ^2/MI/PA)
2	61 65 66 76 77 78 80 83 84	0.569588 / 0.001542 / 0.008092
3	56 57 59 **64** 68 71 81	0.567715 / 0.002704 / 0.012020
4	**43** 47 50 69 70 73 74 82	0.566007 / 0.003849 / 0.015139
5	45 48 63	0.564513 / 0.004980 / 0.017750
6	2 3 7 19 51 52 53 55 60 62 72 75 79	0.563525 / 0.006094 / 0.020116
7	40 44 **46** **67**	0.564060 / 0.007193 / 0.022048
8	10 **42** 54	0.562444 / 0.008278 / 0.024165
9	1 5 20 **28** 32 **58**	0.562407 / 0.009349 / 0.025740
10	6 **4** 8 **9** **11** 12 13 14 **15** 16 **17** 18 21 22 23 **24** 25 26 27 **29** **30** **31** **33** 34 35 **36** 37 **38** 39 **41**	0.559556 / 0.010394 / 0.027360
40	**0**	0.558365 / 0.039715 / 0.076679

Table 4.10: Cardinalities and critical values for χ^2, MI and PA scores. The critical value is influenced by the cardinality of the tested feature.

The result of applying the hypothesis test to the CoIL data is shown in Table 4.11. The relevant and irrelevant features in the table distinguishes between the features for which the null hypothesis was rejected and the features for which the null hypothesis was kept respectively. Note that due to the different number of states for the features of this data set an ordering of the features with respect to the relevance scores makes little sence in this approach. Therefore the features in Table 4.11 are ordered numerically.

If we look closer at these results we see that a hypothesis test with the MI score has declared none of the features irrelevant which our benchmark for the CoIL data set has declared relevant. This is acceptable since we want to be conservative when leaving out features. However, it must be possible to discard more than 13 out of 85 features. The approach can be made less conservative by increasing α.

The PA based hypothesis test discards 45 of the original 85 features rendering it the least conservative of the 3 approaches. Unfortunately 4 of the features which has been deemed irrelevant by this approach are among the features which

Method	Relevant	Irrelevant
MI (avg)	**0** 1 2 3 **4** 5 6 7 8 **9** 10 **11** 12 13 14 **15** 16 **17** 18 19 20 21 22 23 **24** 25 26 27 **28** **29** **30** **31** 32 **33** 34 35 **36** 37 **38** 39 40 **41** **42** **43** 44 45 **46** 47 48 51 53 54 56 **58** 61 63 **64** 65 66 **67** 68 69 71 72 74 75 77 78 79 82 83 84	49 50 52 55 57 59 60 62 70 73 76 80 81
PA (avg)	**0** 2 3 **4** 6 7 8 **9** 10 **11** 12 13 14 **15** 16 **17** 18 19 21 22 23 **24** 25 26 27 **28** **29** **30** **31** 32 **33** 34 35 **36** 37 **38** **41** **42** **64**	1 5 20 39 **43** 44 45 **46** 47 48 49 50 51 52 53 54 55 56 57 **58** 59 60 61 62 63 65 66 **67** 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84
χ^2 (avg)	**0** 2 3 **4** 5 6 7 8 **9** 10 **11** 12 13 14 **15** 16 **17** 18 19 20 21 22 23 **24** 25 26 27 **28** **29** **30** **31** 32 **33** 34 35 **36** 37 **38** 39 40 **41** **42** **43** **46** 51 54 **58** 63 **64** 65 66 72 79 84	1 44 45 47 48 49 50 51 52 53 54 55 56 57 59 60 61 62 **67** 68 69 70 71 73 74 75 76 77 78 80 81 82 83

Table 4.11: Result of applying the hypothesis test to the CoIL data set with significance level 0.05 and sampling 10000 cases. Relevant features according to [38] are marked in bold font.

has been recorded relevant by our benchmark for this data set. The fact that our benchmark is based on supervised learning makes a fair comparison unapplicable and the mismatches are not considered as errors.

At last χ^2 has deemed 42 out of the 84 features irrelevant. This time one of the features which is declared irrelevant is one of the relevant according to our benchmark for this data set. A validation of the filter results obtained in this section compared to the respective clustering models which can be learned from these feature subsets will be presented shortly.

4.2.4 Leukemia

The data sets, LEUKEMIA, AML and ALL are particular interesting for several reasons. The LEUKEMIA data set is well known in the data mining community and thoroughly analyzed in the past. Its extreme number of features can prove to be a challenge for any FSS method. Also indications show that very few features are actually relevant which further challenges the methods by including a large amount of noise. The AML and ALL data sets are interesting in that the patients all suffer from the same type of illness and their gene expression profiles should be similar and therefore also there should be very few or no irrelevant features.

Figure 4.9 on the next page shows the results of applying the score methods to the ALL and AML data set. The top row shows the 3 score measures applied to AML and the bottom row shows them applied to ALL. As before all results are shown using average scores. Expectedly the methods indicate that there are no irrelevant features in the data set. According to the rankings shown in Table 4.12 and 4.13 for the AML data MI and PA agree to some extent on the ordering of the features. Note how the first 5 features and the last 5 features are almost the

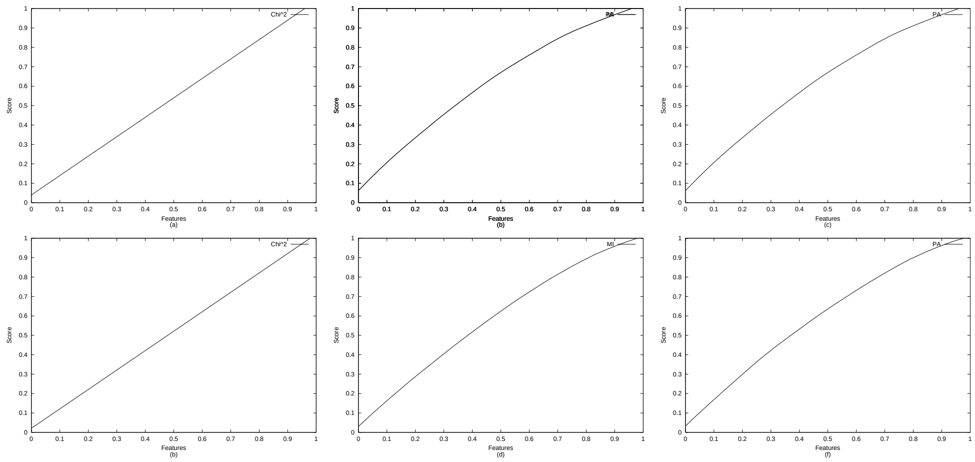

Figure 4.9: Filter results for the AML and ALL data transformed from the original leukemia data set. The top row makes out the AML data set whereas the bottom row makes out the ALL data set. The first column describes χ^2, the middle MI and the right PA. In both cases it is not surprising to see that all patients are relevant due to the fact that their gene expression profiles should bear similarities.

same for both MI and PA although not ordered completely identically. For χ^2 the ordering is trivial since all features have received the same score.

Method	$\alpha = 0.7$	$\alpha = 0.3$	Irrelevant
MI (avg)	**3 12 0 11 6 4 9 14 8 21 5 19 7 2 23 10 1 16 18 20 13 24**	**15 17 22**	
PA (avg)	**6 11 3 12 9 0 19 2 20 8 10 4 15 21 23 7 5 14**	**18 1 16 22 13 17 24**	
χ^2 (avg)	**0 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1**		

Table 4.12: Ranking order (best to worst) of the features in the AML data set. Relevance based on the learning curve thresholding scheme. Since we do not expect any irrelevant features all are marked in bold font.

Method	Relevant	Irrelevant
MI (avg)	**3 12 0 11 6 4 9 14 8 21 5 19 7 2 23 10 1 16 18 20 13 24 15 17 22**	
PA (avg)	**6 11 3 12 9 0 19 2 20 8 10 4 15 21 23 7 5 14 18 1 16 22 13 17 24**	
χ^2 (avg)	**0 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1**	

Table 4.13: Ranking order (best to worst) of the features in the AML data set. Relevance based on the hypothesis test thresholding scheme. Since we do not expect any irrelevant features all are marked in bold font.

Similar tables for the ALL data set can be seen in Tables 4.14 and 4.15. In these tables though the similarity among PA and MI is less striking. The explanation for this could be found in the clustering results in that the results could indicate a less clear ordering of the features due to additional dependencies.

Method	$\alpha = 0.7$	$\alpha = 0.3$	Irrelevant
MI (avg)	**12 45 4 18 14 25 43 29 15 46 36 44 17 24 9 10 34 41 42 35 23a 13 39 11 32 1 21 30 5 33 31 20 22 3 19 0 38 6 8**	**27 28 2 26 37 7 16 40**	
PA (avg)	**41 45 46 25 17 24 12 9 15 18 35 6 3 10 29 34 4 14 36 26 37 1 30 39 44 13 43 5 22 20 31 7 42 21 27 32 23**	**0 11 38 33 28 8 2 40 19 16**	
χ^2 (avg)	**0 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1**		

Table 4.14: Ranking order (best to worst) of the features in the ALL data set. Relevance based on learning curve thresholding scheme. Since we do not expect any irrelevant features all are marked in bold font.

Method	Relevant	Irrelevant
MI (avg)	**12 45 4 18 14 25 43 29 15 46 36 44 37 17 24 9 10 34 41 42 35 23 13 39 11 32 1 21 30 5 33 31 20 22 3 19 0 38 6 8 27 28 2 26 7 16 40**	
PA (avg)	**41 45 46 25 17 24 12 9 15 18 35 6 3 10 29 34 4 14 36 26 37 1 30 39 44 13 43 5 22 20 31 7 42 21 27 32 23 0 11 38 33 28 8 2 40 19 16**	
χ^2 (avg)	**0 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1**	

Table 4.15: Ranking order (best to worst) of the features in the ALL data set. Relevance based on hypothesis test thresholding scheme. Since we do not expect any irrelevant features all are marked in bold font.

We know from previous analysis that a large portion of the features in the LEUKEMIA data set are irrelevant. In Figure 4.10 on the facing page it can be seen that the methods do not indicate any irrelevant features. This is most likely caused by the large amount of features and the lack of cases which strongly increase the possibility of random dependencies among irrelevant features. In fact closer examination revealed that all features include strong dependencies with at least 100 other features. This fact renders any relevance measuring among these features difficult. This is also the case for the proposed scoring methods in this report. It is worth mentioning that PA performs significantly better than both MI and χ^2 by exhibiting a concave shaped graph whereas the other two are almost straight.

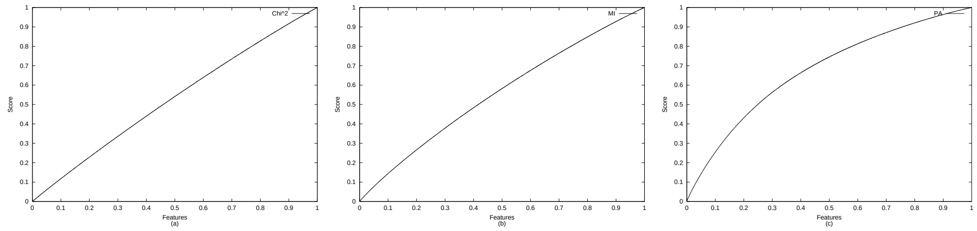

Figure 4.10: Filter results for the leukemia data set in full containing 7129 features and 72 cases. Notice that PA performs significantly different from both MI and χ^2. Considering previous analysis we can say that PA performs significantly better than the 2 other methods.

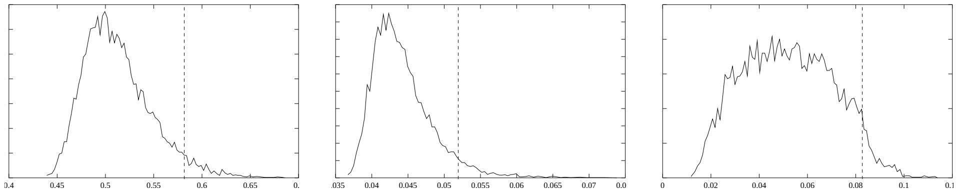

Figure 4.11: The distributions of each of the score measures applied to random features scored against the LEUKEMIA data set. The dashed lines show the critical values. The figures from left to right are for χ^2, MI and PA respectively.

The ordering of the last data set is not shown due to the large amount of features. It is however interesting to see whether the rankings that the methods have produced can prove to be correct. Section 4.4.1 on page 60 will show the results of validating the rankings of all results shown in this section.

	Relevant		
Method	Learning curve ($\alpha = 0.3$)	Learning curve ($\alpha = 0.7$)	Hypothesis test
MI (avg)	7129	6975	6803
PA (avg)	7106	3803	453
χ^2 (avg)	7129	7129	6657

Table 4.16: A summarizing table on the results obtained by the filtering methods performed on the LEUKEMIA data set. The numbers shown are the number of relevant features according to each method.

Table 4.16 shows the number of relevant features returned by each of the 3 score measures combined with both the learning curve approach and the hypothesis test. The PA measure has declared most features irrelevant for both the learning curve approach and the hypothesis test approach. With the learning curve approach it has deemed 3326 features irrelevant while with the hypothesis test it has successfully found 6676 irrelevant genes rendering it the least conservative approach. With MI only 157 was declared irrelevant together with the learning curve approach while 326 features were declared irrelevant when combined with the hypothesis test. χ^2 has detected no irrelevant features when combined with the learning curve. However, combined with a hypothesis test it has declared 472 out of the 7129 features leaving it less conservative than the MI methods. To summarize on this it is clear that PA has proven a quite convincing performance. This, together with the previous results has indicated that it has a higher difference in the scores for features with weak dependencies and the features with strong dependencies. Figure 4.11 depicts the distributions for the relevance measures for irrelevant features scored against the `LEUKEMIA` data set. It is interesting that for this data set PA shows a less peaked distribution than both χ^2 and MI. Despite this PA has filtered out significantly more features than the other two measures.

4.3 Validation of the Filter Results

In the previous section we shown that the two filter methods were capable of filtering out irrelevant features. Their performance was measured by comparing the obtained results to knowledge we were able to gain about the data from an external source. In a second step of validation, we wish to measure each score measures ability to distinguish between features which contribute to homogeneous clusters and whether they are capable of ranking the features with respect to their relevance. In order to do this we apply the 2 clustering methods described in Chapter 2 on the feature subsets that were declared relevant by each filter method. We evaluate the resulting models by measuring their homogeneity and compare it with the homogeneity of a model learned from the whole data set. This task however is far from trivial in that fair comparison between cluster results based on different sized feature subsets can prove to be a challenge. Here we aim to explain how the evaluation is performed and how the homogeneity of clustering with different subsets of features can be measured such that the results are comparable.

4.3.1 The Test Strategy

Our tests are designed to test our methods ability to select relevant features and their ability to correctly identify the order of relevant features. In a first step we learn models from each subset of relevant features according to our filter methods and measure their ability to generate homogeneous results. In a second step we wish to validate the relevance ranking of the features produced by the filter methods. Therefore we learn a model from the most relevant feature, and measure its performance. Then we learn a model from the two most relevant features and measure the performance of this second model. Thus we continue clustering with the most relevant features adding 1 feature for each iteration.

More specifically, we have an ordered set of features $\boldsymbol{X} = \{X_1, X_2, \ldots, X_p\}$, ordered with respect to the relevance measure R, such that $R(X_1) \geq R(X_2) \geq$

$\ldots \geq R(X_p)$, i.e. the most relevant features first. From $\boldsymbol{X}$ we produce p feature subsets such that $S_1 = \{X_1\}, S_2 = \{X_1, X_2\}, \ldots, S_p = \boldsymbol{X}$ with each $S_i \subset S_{i+1}$. Given a performance measure P, which measures the homogeneity of our clusters according to our clustering criterion, we will verify that $P(S_i) \leq P(S_{i+1})$. If this is true we have indices that the ordering produced by R is valid.

4.3.2 Validation Using k-modes

To test whether our relevance measures can successfully be used for FSS as a preprocessing step for the k-modes algorithm, we want to measure the performance of a model learned by a feature subset S_i of the i most relevant features with respect to our clustering criterion P. Ideally, we wish to measure the partitioning in terms of the cohesiveness and distinctiveness of the clusters obtained by a k-mode partitioning with a subset of the original features. A widely used measure when evaluating the performance of a k-means partitioning is the average distance to cluster centroids and the same applies for the k-modes algorithm. However, since we may assume that the number of instances in the data set is constant for all evaluations of feature subsets, the sum of distances to the cluster centroid is equally good. Therefore we use Equation 2.4 to evaluate the goodness of a partitioning.

We want to be able to compare the relevancy of the feature subsets S_i and S_j for $i \neq j$ with respect to our clustering criterion (the performance function P). If we apply Equation 2.4 on models learned from the two subsets and compare the results we would favor the smaller subset and so we need a more fair comparison, and we need the performance function P to be independent of the amount of features used for learning. Therefore we learn the models using the specific subsets S_i and S_j and evaluate the performance based on the full set of features S_p. This means that we use the cluster assignments (or labels l) of each instance $x_i \in D$ we got when k-modes was run on S_i and S_j respectively, and assign the labels to each instance in D. We then apply Equation 2.4 on D with the partitionings obtained by models learned from S_i and S_j respectively. This way we measure the ability of the features in S_i to partition the data base D and achieve homogeneous results. This way the performance function P yields comparable results.

4.3.3 Validation Using NB Models

To validate a probabilistic model, like the NB model, it is common to use the log-likelihood of the data given the learned model, i.e. Equation 2.15. However, like with the k-modes algorithm, it is unfair to compare the performance of different models containing different subsets of features. We wish to distinguish between the use of the performance measure that was used when the model was tested for convergence during learning and the performance measure which, when applied on a learned model, yields results comparable to models trained on a different feature subset.

To obtain a comparison that is similar to that of the k-modes we measure the performance of the whole data set after induction. In the case of the NB model the features that were not included in the learning process are included in one last maximization step in order to calculate their parameters using the current fractional partitioning of the data base. Based on the complete data we now apply the log-likelihood estimate to measure the performance. In other words, the induction of a NB model on a feature subset S_i yields a set of labels l of fractional cluster membership assignments, one for each case in the data base S_i. If we assign those labels to each case in D, run one iteration of the maximization step we have a model including all features in D but which is only learned from the feature subset S_i. On this data base we can apply Equation 2.15 and measure the performance of the features subset S_i in a comparable manner.

4.4 Experimental Evaluation of the Score Measures

In this section we present the results of applying the two clustering algorithms, the k-modes and NB to our data bases. For each score measure we measure the performance of the models which are learned from the relevant features only. The models are evaluated according to our clustering criteria, namely the total distance to the cluster modes for k-modes models and the log-likelihood for the NB models. Both measures are calculated as described above in order to obtain comparable results (the results are comparable within each data base only). Furthermore, we run both the algorithms multiple times, each time with different starting criteria and report only the best possible obtained result measured with the performance function P which takes the entire data set into account. That is, we are looking for the best model which can be learned from S_i, that when its clustermembership assignments are used on the data set D result in homogeneous clusters. In this project we have chosen to choose from 5 models. Further more we pick the starting criteria for each iteration in a deterministic manner such that the same set of starting criteria are evaluated when each feature is added.

The results of measuring the performance of k-modes models learned the features which are relevant according to the filter methods are shown in Table 4.17. The measurements are in total distance to cluster modes measured with Equation 2.1 and the number of clusters k is held constantly at the value mentioned in Section 4.1. The rightmost column shows the performance of models learned from the entire set of features for each data set. The results must be compared with the amount of features which have been discarded. For instance, when the relevant features according to the χ^2 dependency measure performs equally well as the whole data set with a hypothesis test, it must be taken into account that all features in SYN20 are relevant according to this method. If we pay attention to the results obtained with the CoIL data where PA together with the learning curve approach was able to discard 45 features with $\alpha = 0.7$ and 37 features with $alpha = 0.3$. Note that for both feature subsets, the homogeneity of the resulting model is a fraction better than the model learned by the entire data set. The same applies for MI combined with the learning curve approach where 40 and 48 features are filtered out with a small increase in the cluster homogeneity. This may indicate that the k-modes algorithm in some cases performs worse when noisy features are included in the training data.

		Learning curve		Hypothesis test	All Features
Data	Method	$\alpha = 0.7$	$\alpha = 0.3$	$\alpha = 0.05$	
SYN10	MI	85016	85016	85027	76661
	PA	85016	85016	85027	
	χ^2	76702	76661	76661	
SYN20	MI	105346	105346	105346	110883
	PA	105346	105346	105346	
	χ^2	110883	110883	110883	
WAVE	MI	109133	109040	109040	104986
	PA	105890	105890	105922	
	χ^2	105447	104986	104986	
CoIL	MI	118703	118703	118569	118995
	PA	119086	118733	118557	
	χ^2	118995	118995	118557	
AML	MI	70491	70201	70201	70201
	PA	71713	70201	70201	
	χ^2	70201	70201	70201	
ALL	MI	125245	124322	124322	124322
	PA	125321	124322	124322	
	χ^2	124322	124322	124322	
LEUKEMIA	MI	191126	191126	191126	191126
	PA	191126	191126	191126	
	χ^2	191126	191126	191126	

Table 4.17: The performance of the k-modes partitioning models learned from the features which are relevant according to the filter methods measured in Equation 2.1 to cluster modes.

Table 4.18 shows the results of learning NB models from the features which are relevant according to the filter methods. The values are log-likelihoods of the data given the learned NB model. The rightmost column contains the performance of models learned from the entire data set D. It is worth to notice that except for the LEUKEMIA data set models learned from any of the feature subsets do not perform better than the entire data set. This is, as opposed to the k-modes algorithm, an indication of more stability under the presence of noisy features. Also note that for the 3 artificial data sets all the relevant feature subsets perform equally well as the entire data set. Again we point out PAs performance on the CoIL data set. In the case where PA together with the learning curve approach filtered out 45 features the performance only degrades 5 points out of -313942, an insignificant percentage. The same accounts for the rest of the feature sets.

		Learning curve		Hypothesis test	All Features
Data	Method	$\alpha = 0.7$	$\alpha = 0.3$	$\alpha = 0.05$	
SYN10	MI	-174924	-174924	-174924	-174924
	PA	-174924	-174924	-174924	
	χ^2	-174924	-174924	-174924	
SYN20	MI	-242394	-242394	-242394	-242393
	PA	-242394	-242394	-242394	
	χ^2	-242393	-242393	-242393	
WAVE	MI	-200439	-200439	-200439	-200439
	PA	-200439	-200439	-200439	
	χ^2	-200439	-200439	-200439	
CoIL	MI	-313965	-313965	-315817	-313942
	PA	-313947	-313947	-316809	
	χ^2	-313942	-313942	-315810	
AML	MI	-171972	-171940	-171940	-171940
	PA	-172138	-171940	-171940	
	χ^2	-171940	-171940	-171940	
ALL	MI	-305395	-305304	-305304	-305304
	PA	-305407	-305304	-305304	
	χ^2	-305304	-305304	-305304	
LEUKEMIA	MI	-421303	-420990	-420990	-419768
	PA	-419768	-419768	-421262	
	χ^2	-419768	-419768	-421022	

Table 4.18: The performance of NB models learned from the features which are relevant according to the filter methods measured in log-likelihood.

These tests have shown that the filters proposed previously are capable of filtering out features which do not contribute to the clustering with respect to more homogeneous clusters. Moreover, we have seen that for the k-modes algorithm the noisy features are likely to confuse the result rendering the resulting model less homogeneous than a model learned from a subset of features, which are relevant for the clustering.

4.4.1 Relevance Ranking Validation

In this section we aim to validate the rankings of the features based on the relevance scores. In the case of the artificial data sets the ranking should be sufficient to be convinced of their capability since we know which features are truly relevant. We do however perform validation of the relevance ranking for the purpose of showing the reliability of the validation techniques. In addition this section will show the results of validating the real-world data sets. The results in most cases lead to a discussion of the performance of the unsupervised FSS methods and reliability of the clustering methods.

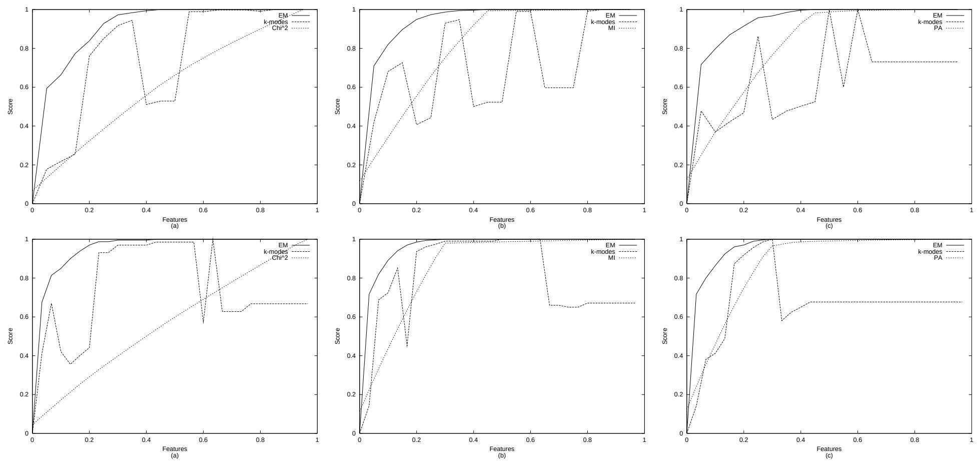

Figure 4.12: The cluster results on the SYN10 and SYN20 data sets using the filter ranking. Graphs show both the filter outputs, the k-modes and NB models results. The top row shows the results of SYN10 whereas the bottom row shows results of SYN20. The leftmost describes χ^2, the middle MI and the right PA.

The graphs shown in this section are normalized between 0 and 1 in both the x-axis and y-axis using Equation 4.1. The y-axis for the clustering techniques represent the score for the current feature subset S_i whereas the x-axis represent the features ordered according to their ranking (best to worst).

Synthetic Data Rank Validation

The cluster results of SYN10 and SYN20 can be seen in Figure 4.12. The most noticeable part of the results is the significant instability in the results of k-modes. However a trend is visible and combined with the results of the NB model the results strongly indicate the the rankings are correct.

All subsets have been clustered with k-modes 5 times and the best result has been selected. The results indicate that more iterations are necessary in order to get more stable results. In comparison the NB model perform much more stable and the results support our previous statement that the score measures are conservative. Clustering with the NB model indicates that less than 10 features are necessary for clustering.

Waveform Rank Validation

Figure 4.13 shows the results of the filters applied to the WAVE data set. In this case though the graphs have been overlaid with results of clustering using the feature subsets specified by the rankings. The validation of PA and MI using NB models provide nice graphs that are very similar indicating that the NB models agree with the rankings produced by the 2 methods. It can be seen from the

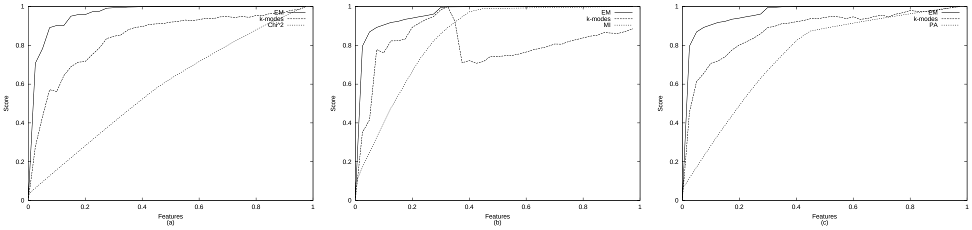

Figure 4.13: The cluster results on the WAVE data set using the filter ranking. Graphs show both the filter outputs, the k-modes and the NB model results. Again the leftmost describes χ^2, the middle MI and the right PA.

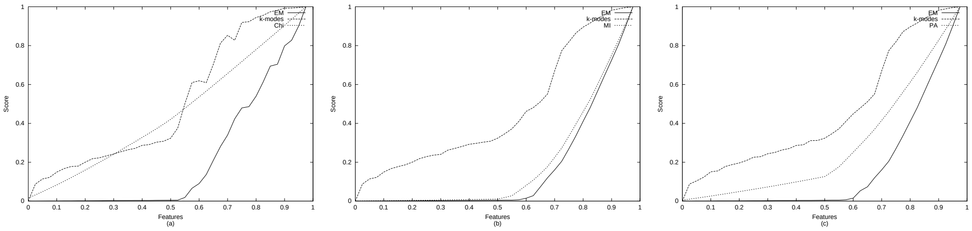

Figure 4.14: Validation of the filter results on the WAVE data set. Graphs show both the filter outputs, the k-modes and the NB model results using reverse order ranking. Again the leftmost describes χ^2, the middle MI and the right PA.

graphs that the accuracy of the model increase only slightly after 30% of the features.

The results of the clustering methods using our rankings indicate that the rankings are correct. The features that contribute with the most information for the clustering have been ranked first. We cannot however, be certain that any random ranking of the features would not produce the same output. None the less for comparison Figure 4.14 shows the same results as before, although this time the features are in reverse order according to the relevance scores proposed. The clustering results clearly show the impact the ordering has on the clustering. In the case of MI and PA the NB model clearly shows only very small improvements of the clustering from 1 feature and up to the total amount of irrelevant features.

The CoIL Challenge Rank Validation

In Figure 4.15 the cluster results for the COIL data set can be seen. The results indicate sensible rankings and in all cases no more than half the features are sufficient for clustering. In many cases significantly fewer features seem necessary.

Another interesting aspect of the graphs is that for all methods the filter is the most conservative, in the middle is k-modes and the most risky results are obtained using the NB model. By risky we refer to the fact that the feature subset that according to the NB model is sufficient for clustering includes only a minimum of features and is more likely to exclude relevant features than analysis

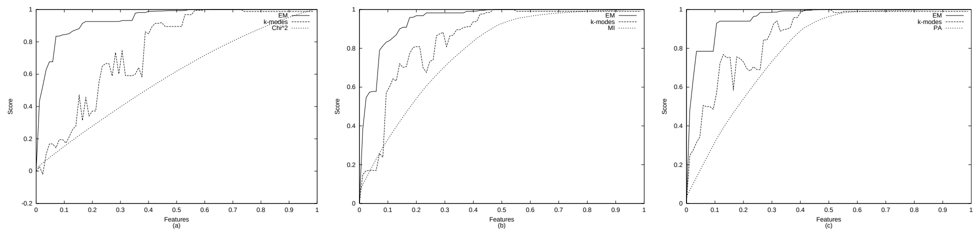

Figure 4.15: Cluster results on the COIL data set using the filter rankings. Graphs show both the filter outputs and the k-modes and the NB model results. The leftmost describes χ^2, the middle MI and the right PA.

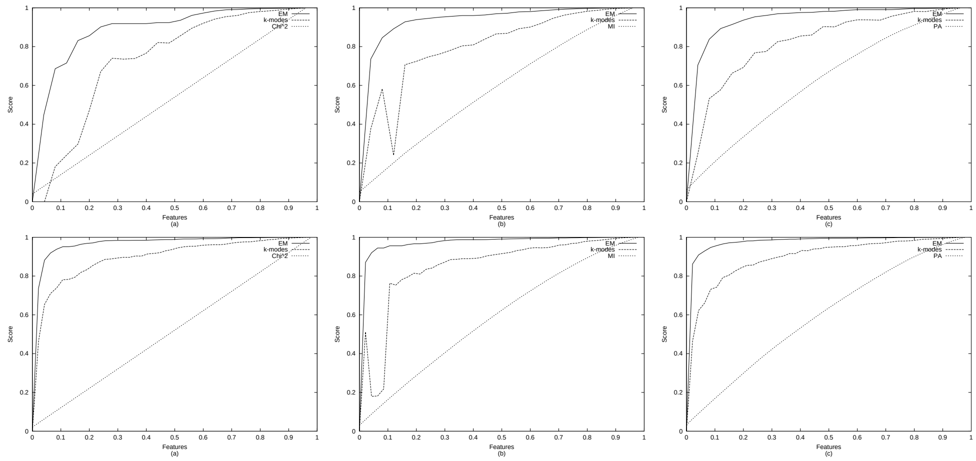

Figure 4.16: Cluster results on the AML and ALL data sets using filter ranking. Graphs show both the filter outputs, the k-modes and the NB model results. The top row illustrates the AML data set whereas the bottom row illustrates the ALL data set. The leftmost describes χ^2, the middle MI and the right PA.

performed using only the filter or k-modes.

Leukemia Rank Validation

Each of the 3 leukemia data sets have been analyzed using the 3 proposed filter methods. Recall that the results found was that the AML and ALL data sets, not surprisingly, did not contain any irrelevant features. Using the same filters it was also difficult to distinguish relevant from irrelevant in the LEUKEMIA data set. What we expect to see in this section is verification that the AML and ALL data sets do not contain irrelevant features, and that the ranking of the LEUKEMIA data set makes sense.

Figure 4.16 presents the clustering results for the AML and ALL data sets. Again it is clear to notice that k-modes is more conservative than the NB model and that the NB model provides more stable results. Comparing with the results of the filter approach a significant difference becomes apparent. According to

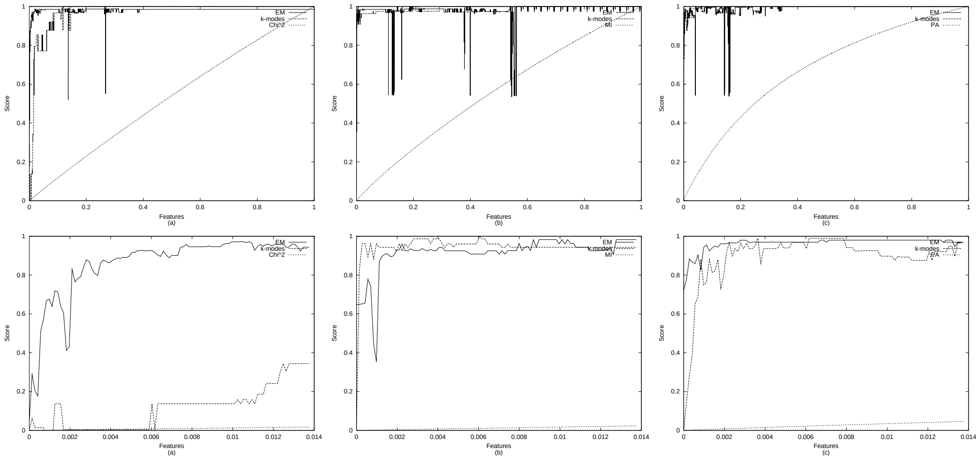

Figure 4.17: Cluster results on the LEUKEMIA data sets using filter ranking. Graphs show both the filter outputs, the k-modes and the NB model results. The top row illustrates the LEUKEMIA data set in full whereas the bottom row illustrates the same data set in which we have zoomed in to 100 features. The leftmost describes χ^2, the middle MI and the right PA.

the filter all features are relevant for clustering, but this is based on an analysis of each feature separately. Figure 4.16 gives a good indication of the difference between scoring a subset as the cumulative score of each feature and scoring a subset as a whole. The clusterings clearly indicate that less than half the features are necessary to obtain a good clustering result. Each feature in it self provides relevant information to clustering, but most of them provide the same information and therefore very few of them are sufficient. This fact is not taken into consideration in the proposed score measures and must be considered a weakness in the approach.

The results of validating the LEUKEMIA data set can be seen in Figure 4.17. The top row shows the clustering results of all 7129 feature subsets. It is clear that according to the graphs very few features can in fact provide a clustering result approximately as accurate as a result based on the full feature set. This corresponds well with previous analysis in the domain of classification.

The figures show both results of k-modes and the NB model and given the amount of features present it is difficult to separate the results. The bottom row gives a more detailed view into the first 100 which seem to indicate that less than 50 features should be sufficient to build a good model. Unfortunately the filter approach seems unable to detect this property although PA combined with the hypothesis test reduces the amount of features to 453. However according to the 2 clustering techniques the reduction is still very conservative.

Another interesting issue in Figure 4.17 is the fact that the scores tend to be more unstable compared to the results of the other data sets. The reason could be in the precision of the used data types which do not perform well for extremely low values. The values used in order to compute the log-likelihood of a NB model will be a multiplication of 7129 probabilities which represents extremely low values. The results however correspond well with expectation and can serve as approximations of the correct results.

4.5 Summary

In this chapter we have shown the proposed filter approach able to produce good results for various data sets. A summarization of these are shown in Table 4.19 which illustrates the amount of relevant features that have been deemed irrelevant (false negatives) and vice versa (false positives). The fact that the amount of false positives exceeds the amount of false negatives indicate that the approaches are conservative.

		Learning curve		Hypothesis test	
Data	Method	fp	fn	fp	fn
SYN10	MI	0	0	0	0
	PA	0	0	0	0
	χ^2	5	0	10	0
SYN20	MI	0	0	0	0
	PA	0	0	0	0
	χ^2	19	0	20	0
WAVE	MI	0	3	0	2
	PA	0	2	0	2
	χ^2	13	0	19	0
CoIL	MI	27	4	51	0
	PA	24	5	22	4
	χ^2	64	0	34	1
AML	MI	0	3	0	0
	PA	0	7	0	0
	χ^2	0	0	0	0
ALL	MI	0	9	0	0
	PA	0	12	0	0
	χ^2	0	0	0	0

Table 4.19: Summarization of the results of the proposed filter approach. The cell values describe both false positives (fp) specifying an irrelevant deemed relevant, and false negatives (fn) specifying relevant features deemed irrelevant. The latter being the most important to avoid.

For the real-world data sets it is difficult to determine truly relevant features. Based on previous analysis a set of truly relevant features have been selected although the analysis usually have been made using supervised FSS and therefore cannot be compared to unsupervised FSS. This explains the false negatives that are visible in Table 4.19. In addition the WAVE data set contains false negatives which is due to the small degree of relevance that is present for these features.

For the LEUKEMIA data set no results are shown in that the set of truly relevant features are unknown. However we can refer to Table 4.16 for details on this data set. The transformed data sets, AML and ALL contain only truly relevant features based on an intuitive understanding of the data.

5 A Hybrid Approach

We all agree that your theory is crazy, but is it crazy enough?
— Niels Bohr

It is a known fact that wrapper approaches produce more reliable results when used for FSS because they rely on the learning method to evaluate the homogeneity of the model obtained by a given feature set. However, wrapper approaches are slow. As opposed to wrapper approaches, the filter approach is faster than wrappers but their independence of the evaluated model leaves them to rely on safe conservative approaches like the filters discussed in the previous chapter.

It is clear that a perfect method for performing FSS has the reliability of the wrapper and the speed of the filter. Therefore, in this chapter we propose a method that takes advantage of both the accuracy of the wrapper and the computational speed of a filter.

In the first part of the chapter we will give a description of the proposed hybrid. Then we will show that the hybrid approach can perform as accurately as a wrapper with a considerable reduction in computational cost compared to ordinary wrappers. The results presented are based on the same data sets that has been used throughout Chapter 4.

5.1 The Method

When performing FSS, the size of the search space is 2^p where p is the number of features. Therefore much research within FSS focus on optimizing the search strategy within this search space. In any kind of problem involving search, the goal is to minimize the amount of points in the search space which has to be evaluated. This task is especially critical for wrapper approaches for FSS where a model has to be learned for each feature subset. Moreover, learning of a model is a search task in itself which involves much uncertainty which must be dealt with in order to assign a fair validation to a feature subset. For instance, evaluation of each feature subset requires multiple models to be learned if the learning algorithm has a probability of being trapped in a local maximum. The computational cost of the induction algorithm renders many advanced heuristic search techniques impossible. Especially approaches that rely on genetic algorithms have to learn a huge amount of models on their way to an accepted feature subset.

The evaluation task is much less critical for filter approaches. However, the filter approaches proposed in this report scores each feature alone and independently of the final task. That is, the score methods we have proposed rewards features which are likely to contribute to cohesive clusters but the evaluation is of each feature alone instead of evaluating an entire feature subset which may be more fair to the features. Scoring single features for relevance favors selecting features that convey the same information instead of selecting features that add independent information. Therefore, the result of evaluating each feature alone may lead to over-rating the features which lead to a too conservative approach compared to what can be obtained with a wrapper.

Here we aim to find an approach which is somewhere in between the two extremes, in order to combine the computational efficiency of the filter and the accuracy of the wrapper. For this purpose we apply a wrapper approach on the remaining feature subset returned by the proposed filter approach to significantly reduce the search space. That is, we apply the score measures presented in Chapter 3 and use the obtained rankings for the subset of relevant features as the order in which the features should be added to the pool of features that are being used for model learning and evaluation. We refer to this method as a hybrid approach in that it benefits from advantages of both filter and wrapper approaches.

5.1.1 The Search Problem

Let $S = \{S_1, S_2, \ldots, S_f\}$ denote an ordered set of feature subsets, constrained by the relevance ranking, that contain the remaining f features after the filter approach has been applied using the relevance measure. We require that each S_i contains the i most relevant features according to our relevance measure, and that $S_i \subset S_{i+1}$, meaning that the subsets are nested. Furthermore, we assume that the performance of the feature subsets is *monotonic* i.e. the performance of the best model which can be learned from S_i is lower than or equal to the performance of the best model learned from S_{i+1}. The validation of the orderings in the previous chapter indicate that this is true for NB but not for k-modes. We then use a wrapper approach to perform a search in a search space of feature subsets based on the features in S.

5.1.2 Thresholding

In this proposal we wish to apply a simple search strategy in which each result is compared to the result obtained from S_f. Prior to the comparison we decide on a margin α, stating how much performance degrading can be accepted. Specifically, we search for a subset with a fraction α of the performance obtained with S_f scaled against the performance which can be obtained with S_1. To do this we need a performance measure P. Let $P(S_i)$ be a function that learns an NB model from S_i and returns the log-likelihood of that model measured with respect to the entire data set D (see Section 4.3.3). We then benefit from Equation 4.1 and obtain

$$P_{norm}(S_i) = \frac{P(S_{(l+r)/2}) \; - \; P(S_1)}{P(S_f) \; - \; P(S_1)}. \tag{5.1}$$

The feature subset S_i we are searching for is the one with a $P_{norm}(S_i)$ closest to α, but yet always above.

Unless stated otherwise the margin set in this project will be a degrading of 3%. In this case α is set to 0.97. The reason for using linear scaling in this search criterion is simply to let α be scalable between multiple data sets.

5.1.3 Binary Search FSS

To find the feature subset which satisfies the above criterion we need a search strategy, and we are even allowed to benefit from the ranking in S. One possibility we have considered is to apply a learning curve strategy like the one proposed for the filter approach. Applying the learning curve thresholding scheme would in this case require using a standard sequential forward search technique which would require a number of learned models proportional to p, which is acceptable in a search space of size 2^p. We have also considered the possibility of applying a hypothesis test. In such a strategy we could use the measure P as test statistic and in a forward search strategy constrained by the order in S, sample a number of scores when randomly generated features are added to a subset S_i. Such a method would not be applicable in a wrapper approach due to the extreme amount of clustering models that are required to be evaluated. For instance, using a sample size of 10000 features would in a worst case scenario require 10000 inductions for each of the features in S_f.

We propose to use a simpler and computationally less heavy search strategy. Taking advantage of the ascending order in S we can apply binary search strategy for the best feature subset [9]. The binary search strategy used works by first evaluating a clustering model using the feature subset S_f. Using this result we can, as a second step, evaluate a clustering model using only half of S_f, namely $S_{f/2}$. We search for the feature subset with a $P(S_i)$ as close to α as possible yet always above the threshold. If the model learned using $S_{f/2}$ performs too

Binary Search FSS

Parameters: A threshold α, an ordered set of feature subsets S with the first element at index l and the last element at r. S_f contains the f features which have not been filtered out by the filter approach.

Returns: The index of the feature subset which performs a fraction $1 - \alpha$ worse than the entire feature subset S_f.

```
$BSFSS(\alpha, S, l, r)$ {
if ($l = r$)
        return $l$
if ( $P_{norm}(S_{(l+r)/2}) > \alpha$)
        return $BSFSS(\alpha, S, l, (r + l)/2)$
else
        return $BSFSS(\alpha, S, (r + l)/2\ + 1, r)$
}
```

Figure 5.1: The Binary Search FSS algorithm (BSFSS) applied for unsupervised FSS in our hybrid approach. It takes as argument an ordered set of feature subsets which is based on the feature rankings returned by one of the score measures in the previous chapter. The number of evaluated models with this approach is proportional to $log_2\, p$.

poorly we evaluate a new model at 75% of S ($S_{3f/4}$), and if $S_{f/2}$ performs better than the threshold, we evaluate a new model at 25% of S, the feature subset $S_{f/4}$. The number of feature subsets to be evaluated using this approach is proportional to $log_2\, p$. We call this strategy *Binary Search FSS* and the details are depicted in Figure 5.1.

5.2 Results

As mentioned the hybrid approach has been tested on the same data sets as used for testing and evaluating the filter approach. The rankings of the features for each data set have already been shown, as well as the performance of the clustering models for each evaluated subset of features. Table 5.1 gives an overview of the results compared to both filter methods.

It is clear for all results that the hybrid approach is able to remove a significant amount of features that the filter could not deem irrelevant. This corresponds well with the fact that the filter is conservative. The thresholding scheme used is very naive and applying another scheme could prove to discard even more features. Especially when examining the ALL graph on Figure 4.16 it seems that 8 to 12 features is a conservative choice and good results could be obtained

		Filter		Hybrid	
Data	Method	Learning curve	Hyp. test	Learning curve	Hyp. test
`SYN10`	MI	10	10	6	6
	PA	10	10	6	6
	χ^2	15	20	7	7
`SYN20`	MI	10	10	6	6
	PA	10	10	6	6
	χ^2	29	30	7	7
`WAVE`	MI	18	19	13	13
	PA	19	26	13	13
	χ^2	34	40	10	10
`CoIL`	MI	44	72	13	16
	PA	40	39	24	16
	χ^2	85	54	21	25
`AML`	MI	22	25	10	10
	PA	18	25	8	9
	χ^2	25	25	17	17
`ALL`	MI	38	47	12	13
	PA	35	47	8	8
	χ^2	47	47	12	12
`LEUKEMIA`	MI	6975	6809	66	66
	PA	3809	459	26	13
	χ^2	7129	6657	182	182

Table 5.1: Overview of the relevant features found using the filter and the hybrid approach.

using only the 5 best features in the data set. However 8 to 12 features is a highly significant reduction compared to the filter approach.

An interesting part of the results is to examine the features that have not been detected as irrelevant by the filter but discarded by the hybrid. Finding similar characteristics among these features could help to improve future proposals for a filter approach.

In Figure 5.2 we have included the BN for the synthetic data including only relevant features. The 4 figures show the probability distribution of each feature inside each cluster and in total. In the figure each feature is denoted `PatientX` where X is the number of the feature. The aim is to give an explanation of the features which according to the filter approach was considered relevant but according to the hybrid was deemed irrelevant. The synthetic data represent the simplest model and has thus been chosen for this purpose.

According to the filter approach for `SYN10` both MI and PA score feature 12 and 15 among the least relevant which according to the hybrid are irrelevant. Examining Figure 5.2 for the probability distributions inside each cluster for these features we notice a striking similarity with other relevant features. For `Patient3` corresponding to feature 12 in the `SYN10` data set, `Patient4` have a probability distribution inside each cluster which is almost identical. This could lead us to believe that the 2 features are redundant. The same applies for `Patient6`

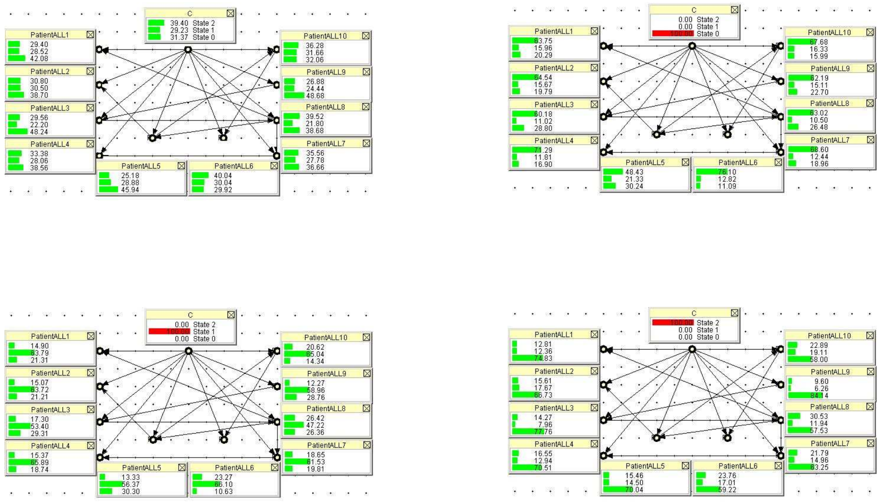

Figure 5.2: BN's of the relevant features in the first synthetic data, created using Hugin [1] to show the probability distribution of individual features inside each cluster. Top left shows the original BN without any fixed states. The other 3 show the probability distribution of all features with the class random variable fixed to one of 3 states.

corresponding to feature 15 in the SYN10 data set and Patient10.

5.2.1 Extensions to the CoIL data results

Applying hypothesis testing on the COIL data set we obtain a feature subset that is not ranked since the features have different cardinalities (see Section 4.2.3). For the hybrid approach we require a ranking of the relevant features regardless of the thresholding scheme used. Therefore in this section we show the performance of relevant features according to the hypothesis testing for the COIL data set.

In Figure 5.3 the clustering results of the relevant features according to the hypothesis testing is shown. The features are ranked given their score measure. The figure shows the clustering results for all feature subsets constrained by the rankings. The hybrid only considers $log_2\ p$ of these subsets but in order to be convinced that the ranking is valid the figure shows the performance P_{norm} for all feature subsets. The figure verifies the results in Table 5.1 in that it is clear that χ^2 drops significantly in performance several features before both PA and MI.

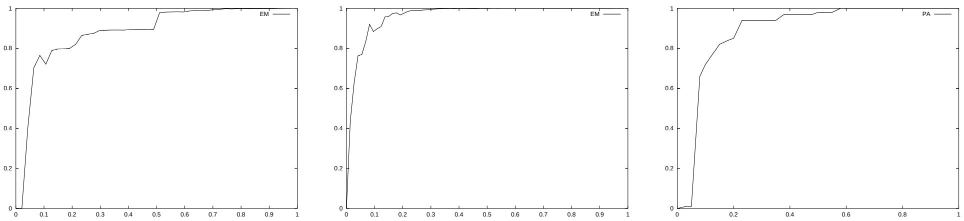

Figure 5.3: Illustration of the graphs for the CoIL data including only relevant features based on the hypothesis test. The graphs support the results of the hybrid for this data set.

5.3 Summary

In this chapter we have proposed a hybrid approach that takes advantage of both the accuracy of the wrapper and the computational cost of the filter. Not surprisingly the method reduce the amount of features significantly compared to the filter approach proposed in Chapter 3.

The results obtained are less conservative and effectively reduce irrelevant features from a given data set. Regarding computational cost we know that the wrapper is the most expensive given the amount of inductions needed to perform. In this proposal we have reduced the amount of inductions needed to $log_2\, p$.

Most of the reductions performed by the hybrid lowers the dimensionality beyond what is know to be relevant. That is, some of the relevant features for synthetic data are also discarded as irrelevant. In this chapter we have continued the discussion regarding redundant features which we believe to be the cause of this behavior. We consider redundancy an important issue to handle in the context of unsupervised FSS regardless of whether the approach is wrapper or filter.

6 Future Work

A great frustration in life is discovering that sometimes those who say something can't be done turn out to be right.
— Donald Simanek

This masters thesis has been developed given a limited amount of time which also limits the degree to which we can accomplish the tasks at hand. The focus has been on broad research in the area of unsupervised FSS and several proposals have been made. The proposals outline potentially interesting facts that could be beneficial in further research. In this section we raise questions in connection to the work done and pose new problems that have yet to be solved.

6.1 The Filter Approach

In this domain 3 score measures have been developed for scoring each feature in a data base and ranking them. FSS has been performed based on 2 proposed thresholding techniques. However, several problems have been raised during the development of the methods.

All 3 proposed score measures are based on a myopic strategy in which the score of each feature is based on individual performance independently of its membership to any feature subsets. A better approach could be propose a score measure which is able to evaluate a subset of features in order to measure their combined score. Given the current approach the score of each feature is independent other features, which is highly likely to overrate a feature resulting in conservative FSS. It has been our experience that features that are part of a semi-clique are very relevant whichi leads us to suggest a score measure that rewards features for being part of a semi-clique. One way of viewing the problem is as a search in a connectivity graph. In the proposals of this report the connectivity graph is assumed to be complete. Another approach was to search for an optimal connnectivity graph in which only true edges are present. This can be accomplished using the proposed score measures to score the edges.

It has been discovered during the tests that some features although relevant according to the filter approach, prove to contribute with very little new information for clustering purposes.

It is our opinion that a discussion of redundance in data clustering is needed. Therefore work could be done in the near future aiming at a discussion and a definition of redundance in data clustering. We suggest research that would indicate how to handle redundant features.

6.2 The Hybrid Approach

The hybrid extends the filter approach by applying a wrapper on the output of the filter. The search space however is constrained given the ranking of the features. We take advantage of previous analysis using the filter in order to further reduce the amount of inductions to $log_2\ p$. The wrapper performs an online search strategy in which we decide after each induction whether or not to continue.

The thresholding scheme used in this proposal is naive although in most cases it performs satisfactory. However referring to the validation of the ALL data set we can observe that the threshold is still conservative. The validation reveals the full shape of all inductions and serves as indications of where the threshold should be. According to the validation of the ALL data set the threshold should be around 4 features which is the end of a steep climb on the graph and where it flattens to a very slow increase given the rest of the features. The task of developing a thresholding scheme however is not trivial in that several properties must be considered.

- The threshold must support an online strategy.
- Performance cannot be assumed to decrease in the number of features.
- Unless no features are irrelevant, induction on the full data set is prohibited.

The current approach requires induction on the full feature subset that is output from the filter approach. In most cases this will not be the full data set but still the induction can be expensive. One can also argue that applying post processing on this feature subset could make further inductions obsolete. Therefore it is desirable to perform inductions only on feature subsets that do not exceed the size of the resulting feature subset.

As the required number of inductions have been greatly reduced it becomes applicable to perform as search for the optimal model in which the assumption that the number of clusters k is unknown. Future research could explore this area and perform empirical tests.

7 Conclusion

It's never too late to give up.
— Ronny Ericson

In Chapter 1 we motivated and limited this project to be within the field of data clustering which we consider a critical task in data mining. In addition to this broad area, we have limited our work to be concerned with unsupervised feature subset selection (FSS), which we consider the process of identifying irrelevant features which can be left out without doing any harm to the resulting model.

The most critical part of our approach is how we wish to characterize features which are relevant for data clustering. In this report we argue that relevant features must depend on the cluster random variable and hence, they must be d-connected given no evidence on the cluster variable. In the absence of the cluster random variable we define a relevant feature as a feature that is dependent on at least one other feature. For the purpose of measuring dependence between pairs of features we have applied 3 dependency measures:

- χ^2 analysis.
- Predictive accuracy.
- Mutual Information

In order to further measure the relevance of features we apply relevance measures based on pairwise dependencies among features in the data set in order to measure, compare and rank the features with respect to relevance. For this purpose 2 relevance measures were proposed:

$$\begin{aligned} R_{max}(X_i) &= maxDM(X_i, X_j) \\ R_{avg}(X_i) &= \frac{\sum_j DM(X_i, X_j)}{p} \end{aligned}$$

Based on the above relevance measures we have proposed and tested the following methods to identify irrelevant features. First we propose a filter approach which works by ranking the features according to their relevance score and selecting only the most relevant features by setting a threshold. The threshold is set using a learning curve sampling method using a cost versus benefit approach.

In a second approach for performing FSS with the proposed relevance measures we propose a filter method based on a hypothesis test known from statistics. The hypothesis test uses the relevance measures as a test statistic obtained by scoring a sample set of randomly generated features against the database. A feature is declared relevant if its test statistic provides sufficient evidence against a hypothesis of independence.

Lastly, a verification that the score measures are truly capable of ranking the features with the most informative features first, has motivated a hybrid approach. The hybrid approach can be seen as a wrapper that takes advantage of the rankings provided by the proposed relevance measures. The use of this ranking significantly reduces the number of feature subsets in the search space this approach has to inspect. For the purpose of induction we have used the Naive-Bayes model. Additionally we propose to reduce the number of inspected feature subsets by discarding those features that have been deemed irrelevant by a filter approach.

Experimental evaluation has been performed on the 3 proposed FSS methods using 3 synthetic and 4 real-world data sets. The relevance measures were tested in their ability to correctly identify the irrelevant features. By comparing the obtained results with the knowledge we have about the data the relevance measures showed capable of ranking the truly relevant features first. Table 7.1 gives an overview of the performance of each of the proposed methods for all data sets.

The hybrid approach is in most cases able to make significant additional reductions which corresponds to our belief that the filter approach is conservative. This is especially noticeable for the 3 leukemia data sets (`AML`, `ALL` and `LEUKEMIA`). Considering the knowledge we have on the artificial data sets the rankings produced, correctly rank all relevant features first. With regards to the `WAVE` data set the rankings also reflect correct ranking within relevant features. All rankings are supported by validation performed using the Naive-Bayes and k-modes clustering techniques.

Most remarkable are the results of the PA relevance measure. In case of the high dimensional `LEUKEMIA` database it is, together with the hypothesis test, able to discard 3320 features. If the hybrid is applied on the remaining feature subset, only 13 features remain. A model learned on this small feature subset validates that the removal of the 7116 features inflicts almost no harm to the learned model.

		Filter		Hybrid	
Data	Method	Learning curve	Hyp. test	Learning curve	Hyp. test
SYN10	MI	10	10	6	6
	PA	10	10	6	6
	χ^2	15	20	7	7
SYN20	MI	10	10	6	6
	PA	10	10	6	6
	χ^2	29	30	7	7
WAVE	MI	18	19	13	13
	PA	19	26	13	13
	χ^2	34	40	10	10
CoIL	MI	44	72	13	16
	PA	40	39	24	16
	χ^2	85	54	21	25
AML	MI	22	25	10	10
	PA	18	25	8	9
	χ^2	25	25	17	17
ALL	MI	38	47	12	13
	PA	35	47	8	8
	χ^2	47	47	12	12
LEUKEMIA	MI	6975	6809	66	66
	PA	3809	459	26	13
	χ^2	7129	6657	182	182

Table 7.1: Overview of the relevant features found using the filter and the hybrid approach.

A Hypothesis Distributions

Here we present the graphs for the hypothesis tests performed on the data sets: WAVE, ALL, AML and COIL. On all figures the dashed lines show the critical values. On all the figures the graphs are from left to right, χ^2, MI and PA respectively.

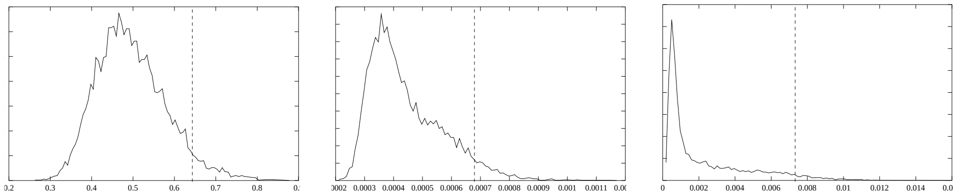

Figure A.1: The distiributions of each of the score measures applied to random features scored against the ALL data set.

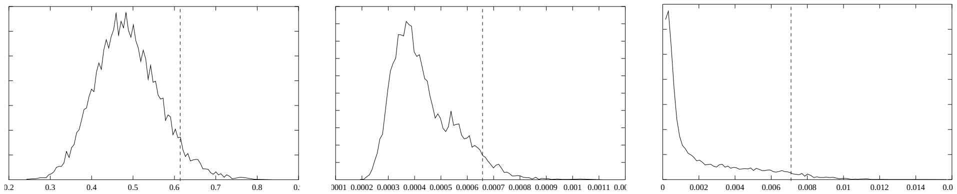

Figure A.2: The distiributions of each of the score measures applied to random features scored against the AML data sets.

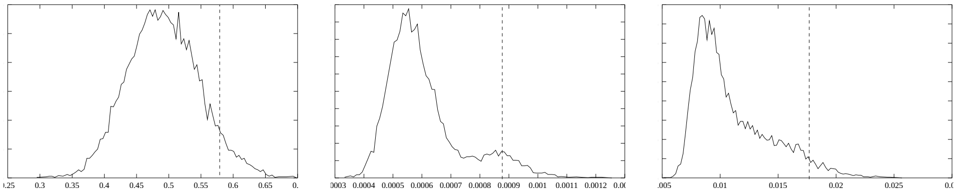

Figure A.3: The distiributions of each of the score measures applied to random features scored against the `WAVE` data set.

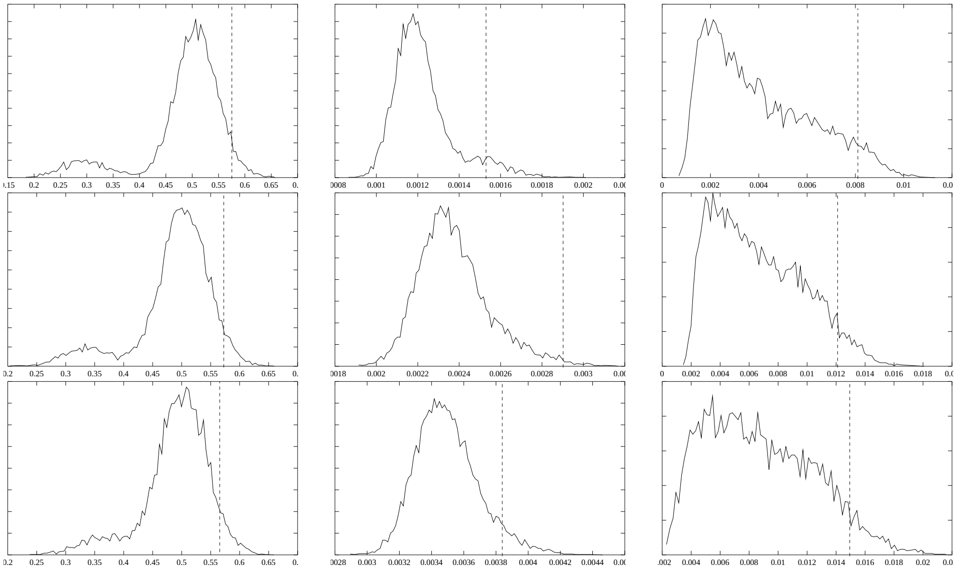

Figure A.4: The distiributions of each of the score measures applied to random features scored against the `COiL` data set. The rows are from above, cardinality 2, cardinality 3 and for cardinality 4.

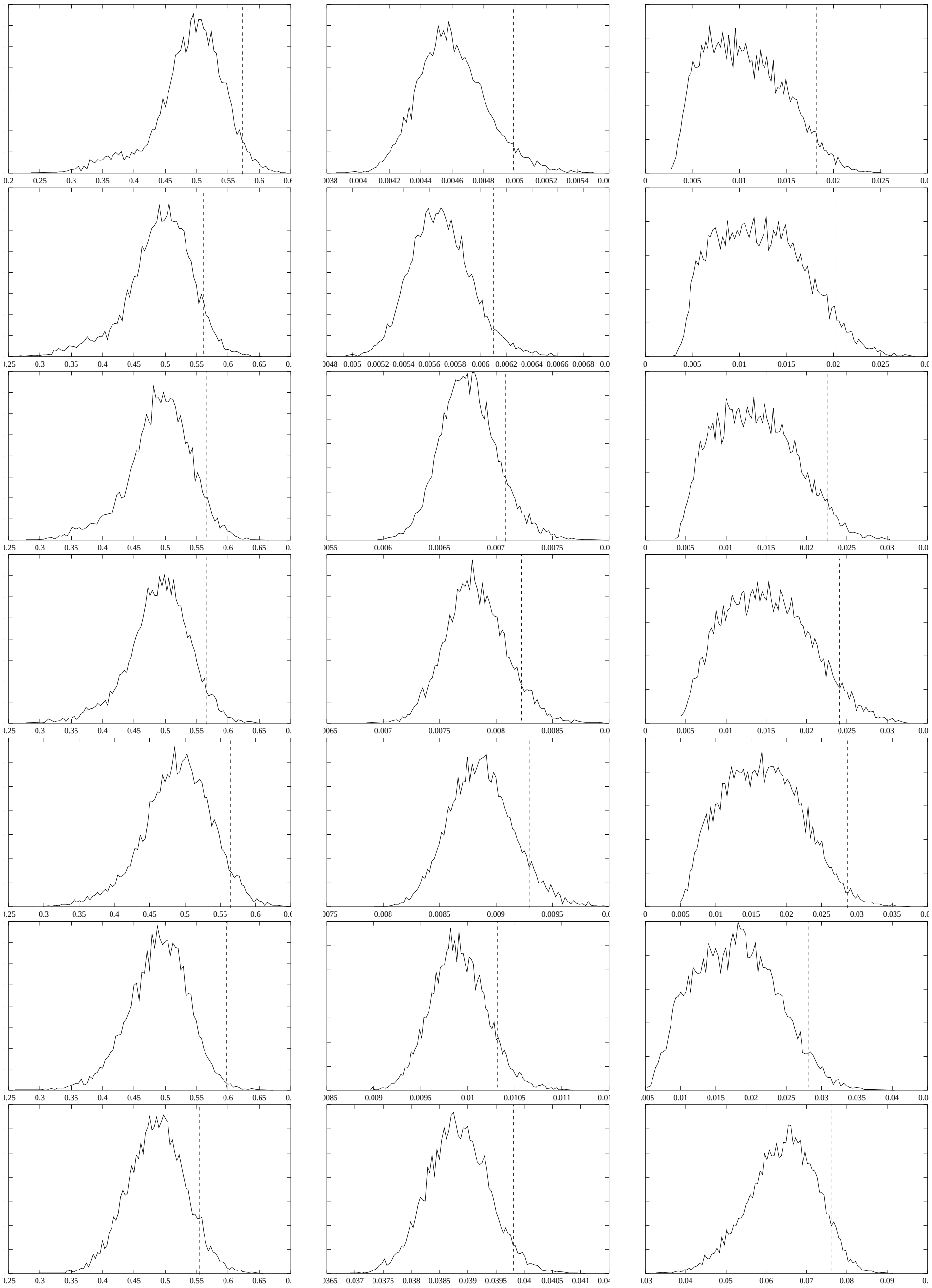

Figure A.5: The distiributions of each of the score measures applied to random features scored against the `COiL` data set. The rows are from above, cardinality 5, 6, 7, 8, 9, 10 and for cardinality 40.

Bibliography

[1] Hugin expert. http://www.hugin.com, 2003.

[2] Merriam-webster dictionary. http://www.m-w.com, 2003.

[3] M. R. Anderberg. *Cluster Analysis for Applications*. Academic Press, 1973.

[4] B. M. Ayyub and R. H. McCuen. *Probability, Statistics, and Reliability for Engineers*. CRC Press, 1997.

[5] T. Bäck. *Evolutionary Algorithms in Theory and Practice*. Oxford University Press, 1996.

[6] P. E. Black. Dictionary of Algorithms and Data Structures. http://www.nist.gov/dads/HTML/hammingdist.html, 2003.

[7] C. L. Blake and C. J. Merz. UCI Repository of Machine Learning Databases. http://www.ics.uci.edu/∼mlearn/MLRepository.html, 1998.

[8] L. Breinman, J. H. Friedman, R.A. Olshen, and C.J. Stone. *Classification and Regression Trees*. Belmont, CA, 1984. Wadsworth International Group, 1984.

[9] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press, 1st edition, 1990.

[10] M. Dash, H. Liu, and J. Yao. Dimensionality Reduction for Unsupervised Data. In *Ninth IEEE International Conference on Tools with AI, ICTAI '97*, 1997.

[11] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society (Series B)*, (39):1–39, 1977.

[12] J. Dougherty, R. Kohavi, and M. Sahami. Supervised and Unsupervised Discretization of Continuous Features. In *International Conference on Machine Learning*, pages 194–202, 1995.

[13] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. John Wiley and Sons., 1973.

[14] J. G. Dy and C. E. Brodley. Feature Subset Selection and Order Identification for Unsupervised Learning. In *Seventeenth International Conference on Machine Learning*, 2000.

[15] V. J. Easton and J. H. McColl. Statistics Glossary v. 1.1. http://www.cas.lancs.ac.uk/glossary_v1.1/hyptest.html, 2000.

[16] G. Elidan, N. Lotner, N. Friedman, and D. Koller. Discovering Hidden Variables: A Structure-Based Approach. In *Neural Information Processing Systems*, pages 479–485, 2000.

[17] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From Data Mining to Knowledge Discovery: an Overview. In *Advances in Knowledge Discovery and Data Mining*, pages 1–34. AAAI Press, Cambridge, Massachusetts, 1996.

[18] D. H. Fisher. Knowledge Acquisition Via Incremental Conceptual Clustering. *Machine Learning*, (2):139–172, 1987.

[19] R. A. Fisher. *Statistical Methods for Research Workers*. Oliver and Boyd, Edinburg, 6th edition, 1936.

[20] D. B. Fogel. An Introduction to Simulated Evolutionary Optimisation. *IEEE Transactions on Neural Networks*, 5(1):3–14, 1994.

[21] L. J. Fogel. Autonomous Automata. *Industrial Research*, 4:14–19, 1962.

[22] L. J. Fogel. *On the Organization of Intellect*. PhD Thesis, University of California, 1964.

[23] H. Frigui, N. Boujemaa, and S. Lim. Unsupervised Clustering and Feature Discrimination with Application to Image Database Categorization. In *Joint 9th International Fuzzy Systems Association World Congress and 20th North American Fuzzy Information Processing Society Conference, Vancouver, Canada*, 2001.

[24] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.

[25] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.

[26] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caliguiri, C. D. Bloomfield, and E. S. Lander. Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring. *Science Magazine*, 286, 1999.

[27] M. Hall and L. A. Smith. Practical Feature Subset Selection for Machine Learning. In *Proceedings of the 21st Australian Computer Science Conference*, pages 181–191, Anaheim, California, 1998.

[28] J. Han and M. Kamber. *Data Mining, Concepts and Techniques*. Academic Press, 2001.

[29] J. A. Hartigan. *Clustering Algorithms*. John Wiley and Sons, 1975.

[30] J. H. Holland. *Adaptation in Natural and Artificial Systems*. The University of Michigan Press, 1975.

[31] Z. Huang. A Fast Clustering Algorithm to Cluster Very Large Categorical Data Sets in Data Mining. In *Research Issues on Data Mining and Knowledge Discovery*, 1997.

[32] I. Inza, P. Larrañaga, and B. Sierra. Feature Subset Selection by Bayesian Networks based Optimization. Technical Report EHU-KZAA-IK-2/99, Dept. of Computer Science and Artificial Intelligence. University of the Basque Country. http://www.sc.ehu.es/ccwbayes/postscript/AI2000Inaki.ps.gz. 1999.

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

[34] A. K. Jain, M. N. Murty, and P. J. Flynn. Data Clustering: A Review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[35] F. V. Jensen. *Bayesian Networks and Decision Graphs*. Springer-Verlag, 2001.

[36] G. H. John, R. Kohavi, and K. Pfleger. Irrelevant Features and the Subset Selection Problem. In *International Conference on Machine Learning*, pages 121–129, 1994.

[37] R. Jurgelnaite, A. O. M. Addin, N. Søndberg-Madsen, and C. Thomsen. Industrial Data Mining — A Data Mining Project in Cooperation with Green House. Master Thesis from Aalborg University, Denmark. 2002.

[38] Y. Kim and W. N. Street. The CoIL Challenge 2000: Choosing and Explaining Likely Caravan Insurance Customers. Technical Report 200009, University of Iowa. Sentient Machine Research and Leiden Institute of Advanced Computer Science. http://www.wi.leidenuniv.nl/∼putten/library/cc2000/. 2000.

[39] Y. Kim, W. Nick Street, and F. Menczer. Feature Selection in Unsupervised Learning via Evolutionary Search. In *Proceedinmgs of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-00)*, pages 365–369, N. Y., 2000.

[40] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by Simulated Annealing. *Science*, 220:671–680, 1983.

[41] R. Kohavi and G. H. John. Wrappers for Feature Subset Selection. *Artificial Intelligence*, (1–2):273–324, 1997.

[42] L. Kaufman and P. J. Rousseouw. *Finding Grups in Data*. John Wiley and Sons, 1990.

[43] P. Langley. Selection of Relevant Features in Machine Learning. In *American Association for Artificial Intelligence Fall Symposium on Relevance*, pages 140–144, 1994.

[44] P. Larrañaga and J. A. Lozano. *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, 2001.

[45] S. L. Lauritzen. Propagation of Probabilities, Means, and Variances in Mixed Graphical Association Models. *Journal of the American Statistical Association*, 87(420):1098–1108, 1992.

[46] J. B. MacQueen. Some Methods for Classification and Analysis of Multivariate Observations. In *the 5th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[47] G. J. McLachlan and K. E. Bashford. *Mixture Models: Inference and Applications to Clustering*. Marcel Dekker, 1988.

[48] C. Meek, B. Thiesson, and D. Heckerman. The Learning-Curve Sampling Method Applied to Model-Based Clustering. *Journal of Machine Learning Research*, 2:397–418, 2002.

[49] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of State Calculations by Fast Computing Machines. *Journal of Chemical Physics*, 21:1087–1092, 1953.

[50] T. M. Mitchell. *Machine Learning*. McGraw-Hill, 1st edition, 1997.

[51] I. H. Osman and J. P. Kelly. *Meta-Heuristics: Theory and Applications*. Kluwer Academic Publishers, 1996.

[52] B. Palace. Data mining: What is data mining? http://www.anderson.ucla.edu/faculty/jason.frand/teacher/technologies/palace/datamining.htm. 1996.

[53] J. Pearl. *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley, 1984.

[54] J. M. Peña. *On Unsupervised Learning of Bayesian Networks and Conditional Gaussian Networks*. Ph.D. Thesis, University of the Basque Country, 2001.

[55] J. M. Peña, J. Lozano, and J. Larrañaga. An empirical Comparison of Four Initialization Methods for the k-Means Algorithm. *Pattern Recognition Letters*, 20(50):1027–1040, 1999.

[56] J. M. Peña, J. Lozano, and P. Larrañaga. Unsupervised Learning of Bayesian Networks Via Estimation of Distribution Algorithms: An Application to Gene Expression Data Clustering. In *Proceedings of the First European Workshop on Probabilistic Graphical Models*, pages 144–151, 2002.

[57] J. M. Peña, J. Lozano, P. Larrañaga, and I. Inza. Dimensionality Reduction in Unsupervised Learning of Conditional Gaussian Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23 (6):590–603, 2001.

[58] D. Pyle. *Data Preparation for Data Mining*. Morgan Kaufmann Publishers, Inc., 1999.

[59] I. Rechenberg. *Evolutionstrategie: Optimierung Technischer Systeme Nach Prinzipien der Biologischen Evolution*. Fromman-Holzboog Verlag, 1973.

[60] C. R. Reeves. *Modern Heuristic Techniques for Combinatorial Problems*. Blackwell Scientific Publications, 1993.

[61] S. M. Ross. *Introduction to Probability and Statistics for Engineers and Scientists*. Academic Press, 2nd edition, 2000.

[62] H. P. Schwefel. *Numerical Optimization of Computer Models*. John Wiley and Sons, 1981.

[63] M. Sebban and R. Nock. A Hybrid Filter/Wrapper Approach of Feature Selection using Information Theory, 2001.

[64] P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. Springer-Verlag, 1993.

[65] L. Talavera. Dependency-Based Feature Selection for Clustering Symbolic Data. *Intelligent Data Analysis*, 4:19–28, 2000.

[66] B. Thiesson, C. Meek, D. Chickering, and D. Heckerman. Computationally efficient methods for selecting among mixtures of graphical models, with discussion. In *Proceedings of the Sixth Valencia International Meeting. Bayesian Statistics*, volume 6, pages 631–656, 1999.

[67] S. B. Thrun, J. Bala, E. Bloedorn, I. Bratko, B. Cestnik, J. Cheng, K. De Jong, S. Džeroski, S. E. Fahlman, D. Fisher, R. Hamann, K. Kaufman,

S. Keller, I. Kononenko, J. Kreuziger, R. S. Michalski, T. Mitchell, P. Pachowicz, Y. Reich, H. Vafaie, W. Van de Welde, W. Wenzel, J. Wnek, and J. Zhang. The MONK's Problems: A Performance Comparison of Different Learning Algorithms. Technical Report CS-91-197, Pittsburgh, PA, 1991.

[68] A. A. Törn and A. Žilinskas. *Global Optimization*. Springer-Verlag, 1989.

[69] P. van der Putten and M. van Someren. CoIL Challenge 2000: The Insurance Company Case. In *Sentient Machine Research*, 2000.

[70] J. Whittaker. *Graphical Gaussian Models in Applied Multivariate Statistics*. Wiley Publishers, 1990.

[71] J. Yang and V. Honavar. Feature Subset Selection Using a Genetic Algorithm. *IEEE Intelligent Systems*, 13:44–49, 1998.

[72] A. A. Zhigljavsky. *Theory of Global Random Search*. Kluwer Academic Publishers, 1991.