JMLR Workshop and Conference Proceedings
Volume 10: Feature Selection in Data Mining

Proceedings of the Fourth International Workshop on Feature Selection in Data Mining, June 21st, 2010, Hyderabad, India

Editors: Huan Liu, Hiroshi Motoda, Rudy Setiono, Zheng Zhao

Preface


Feature Selection: An Ever Evolving Frontier in Data Mining


Feature Selection, Association Rules Network and Theory Building


A Statistical Implicative Analysis Based Algorithm and MMPC Algorithm for Detecting Multiple Dependencies


Attribute Selection Based on FRiS-Compactness


Effective Wrapper-Filter hybridization through GRASP Schemata


Feature Extraction for Machine Learning: Logic-Probabilistic Approach


Feature Extraction for Outlier Detection in High-Dimensional Spaces


Feature Selection for Text Classification Based on Gini Coefficient of Inequality


Increasing Feature Selection Accuracy for L1 Regularized Linear Models


Learning Dissimilarities for Categorical Symbols

Preface

Welcome to FSDM’10

Knowledge discovery and data mining (KDD) is a multidisciplinary field that researches and develops theories, algorithms and software systems to mine gold nuggets of knowledge from data. The increasingly large data sets from many application domains have posed renewed challenges to KDD; in the meantime, new types of data are evolving such as social media, text, and microarray data. Researchers and practitioners in multiple disciplines and various IT sectors confront similar issues in feature selection, and there is still a pressing need for continued exchange and discussion of challenges and ideas, exploring new methodologies and innovative approaches in search of breakthroughs.

Feature selection is effective in data preprocessing and reduction, thus is an essential step in successful data mining applications. Feature selection has been a research topic with practical significance in many areas such as statistics, pattern recognition, machine learning, and data mining (including Web, text, image, and microarrays). The objectives of feature selection include: building simpler and more comprehensible models, improving data mining performance, and helping prepare, clean, and understand data. The Workshop on Feature Selection in Data Mining (FSDM) aims to further the cross-discipline, collaborative effort in feature (a.k.a. variable) selection research and application. This year, FSDM is held with the 14th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2010) in Hyderabad, India.

FSDM’10 consists of one keynote speech and 8 peer-reviewed papers, in which four papers are on developing new algorithms or improving existing algorithms of feature selection; two papers on designing effective feature selection algorithms for real-world problems; and three papers on exploring novel problems in feature selection research.

It has been an enjoyable journey for us to work together with program committee members and authors to make this workshop a reality. We would like to convey our immense gratitude to the PC members for spending their precious time helping review and select papers, and to all the authors for their contributions and efforts in generating the FSDM’10 proceedings. Last but not least, we would like to thank Neil Lawrence from JMLR, and the organizers of PAKDD for their guidance and help in producing the proceedings and in organizing this workshop.

Huan Liu, Hiroshi Motoda, Rudy Setiono, Zheng Zhao

June 21, Hyderabad, India

©2010 Liu, Motoda, Setiono and Zhao.
FSDM’10 Workshop Chairs

- Huan Liu (Arizona State University)
- Hiroshi Motoda (Osaka University)
- Rudy Setiono (National University of Singapore)
- Zheng Zhao (Arizona State University)

Workshop Program Committee

- Constantin Aliferis (Langone Medical Center, USA)
- Leonardo Auslender (SAS Institute, USA)
- Selin Aviyente (Michigan State University, USA)
- Gianluca Bontempi (Universite Libre de Bruxelles, Belgium)
- Zheng Chen (Microsoft Research Asia, China)
- Soon Chung (Wright State University, USA)
- Anirban Dasgupta (Yahoo! Research, USA)
- Manoranjan Dash (Nanyang Technological University, Singapore)
- Petros Drineas (Rensselaer Polytechnic Institute, USA)
- Pierre Dupont (University catholique de Louvain, Belgium)
- Wei Fan (IBM Watson, USA)
- Assaf Gottlieb (Tel Aviv University, Israel)
- Mark Hall (University of Waikato, New Zealand)
- Michael E. Houle (National Institute of Informatics, Japan)
- D. Frank Hsu (Fordham University, USA)
- Inaki Inza (University of the Basque Country, Spain)
- Rong Jin (Michigan State University, USA)
- Irwin King (The Chinese University of Hong Kong)
- Jacek Koronacki (Institute of Computer Science, Polish Acad. Sci., Poland)
- Igor Kononenko (University of Ljubljana, Slovenia)
- Mineichi Kudo (Hokkaido University, Japan)
Preface

- James Kwok (Hong Kong University of Science and Technology, China)
- Yanjun Li (Fordham University, USA)
- Huan Liu (Arizona State University, USA)
- xiaohui liu (Brunel University, UK)
- Fabricio M. Lopes (Federal University of Technology - Parana, Brazil)
- Kezhi Mao (Nanyang Technological University, Singapore)
- Elena Marchiori (Radboud University, Netherlands)
- Hiroshi Motota (Osaka University and AFOSR/AOARD, Japan)
- Satoshi Niijima (Kyoto University, Japan)
- Tao Qin (Tsinghua University, China)
- Chai quek (Nanyang Technological University, Singapore)
- Marko Robnik-Sikonja (University of Ljubljana, FRI, Slovenia)
- Yvan Saeys (Ghent University, Belgium)
- Rudy Setiono (National University of Singapore)
- Jian-Tao Sun (Microsoft Research Asia, China)
- Ioannis Tsamardinos (University of Crete, Greece)
- Fei Wang (IBM Almaden Research Center, USA)
- Lei Wang (The Australian National University, Australia)
- Louis Wehenkel (University of Liege, Belgium)
- Zenglin Xu (Chinese University of Hong Kong, China)
- Jieping Ye (Arizona State University, USA)
- Lei Yu (Binghamton University, USA)
- Kun Zhang (Xavier University of Louisiana, USA)

Workshop web site

http://featureselection.asu.edu/fsdm10/index.html
Feature Selection: An Ever Evolving Frontier in Data Mining

Huan Liu
Computer Science and Engineering,
Arizona State University, USA

Hiroshi Motoda
Asian Office of Aerospace Research and Development,
Air Force Office of Scientific Research,
US Air Force Research Laboratory, JAPAN
and Institute of Scientific Research,
Osaka University, JAPAN

Rudy Setiono
School of Computing,
National University of Singapore, SINGAPORE

Zheng Zhao
Computer Science and Engineering,
Arizona State University, USA

Editor: Neil Lawrence

Abstract

The rapid advance of computer technologies in data processing, collection, and storage has provided unparalleled opportunities to expand capabilities in production, services, communications, and research. However, immense quantities of high-dimensional data renew the challenges to the state-of-the-art data mining techniques. Feature selection is an effective technique for dimension reduction and an essential step in successful data mining applications. It is a research area of great practical significance and has been developed and evolved to answer the challenges due to data of increasingly high dimensionality. Its direct benefits include: building simpler and more comprehensible models, improving data mining performance, and helping prepare, clean, and understand data. We first briefly introduce the key components of feature selection, and review its developments with the growth of data mining. We then overview FSDM and the papers of FSDM10, which showcases of a vibrant research field of some contemporary interests, new applications, and ongoing research efforts. We then examine nascent demands in data-intensive applications and identify some potential lines of research that require multidisciplinary efforts.

Keywords: Feature Selection, Feature Extraction, Dimension Reduction, Data Mining

1. An Introduction to Feature Selection

Data mining is a multidisciplinary effort to extract nuggets of knowledge from data. The proliferation of large data sets within many domains poses unprecedented challenges to data mining (Han and Kamber, 2001). Not only are data sets getting larger, but new types of data become prevalent, such as data streams on the Web, microarrays in genomics...
and proteomics, and networks in social computing and system biology. Researchers are realizing that in order to achieve successful data mining, feature selection is an indispensable component (Liu and Motoda, 1998; Guyon and Elisseeff, 2003; Liu and Motoda, 2007). It is a process of selecting a subset of original features according to certain criteria, and an important and frequently used technique in data mining for dimension reduction. It reduces the number of features, removes irrelevant, redundant, or noisy features, and brings about palpable effects for applications: speeding up a data mining algorithm, improving learning accuracy, and leading to better model comprehensibility. Various studies show that some features can be removed without performance deterioration (Ng, 2004; Donoho, 2006). Feature selection has been an active field of research for decades in data mining, and has been widely applied to many fields such as genomic analysis (Inza et al., 2004), text mining (Forman, 2003), image retrieval (Gonzalez and Woods, 1993; Swets and Weng, 1995), intrusion detection (Lee et al., 2000), to name a few. As new applications emerge in recent years, many challenges arise requiring novel theories and methods addressing high-dimensional and complex data. Feature selection for data of ultrahigh dimensionality (Fan et al., 2009), steam data (Glocer et al., 2005), multi-task data (Liu et al., 2009; G. Obozinski and Jordan, 2006), and multi-source data (Zhao et al., 2008, 2010a) are among emerging research topics of pressing needs.

Figure 1: A unified view of a feature selection process

Figure 1 presents a unified view for a feature selection process. A typical feature selection process contains two phases: feature selection, and model fitting and performance evaluation. The feature selection phase contains three steps: (1) generating a candidate set containing a subset of the original features via certain research strategies; (2) evaluating the candidate set and estimating the utility of the features in the candidate set. Based on the evaluation, some features in the candidate set may be discarded or added to the selected feature set according to their relevance; and (3) determining whether the current
set of selected features are good enough using certain stopping criterion. If it is, a feature selection algorithm will return the set of selected features, otherwise, it iterates until the stopping criterion is met. In the process of generating the candidate set and evaluating it, a feature selection algorithm may use the information from the training data, current selected features, target learning model, and given prior knowledge (Helleputte and Dupont, 2009) to guide their search and evaluation. Once a set of features is selected, it can be used to filter the training and test data for model fitting and prediction. The performance achieved by a particular learning model on the test data can also be used to as an indicator for evaluating the effectiveness of the feature selection algorithm for that learning model.

In the process of feature selection, the training data can be either labeled, unlabeled or partially labeled, leading to the development of supervised, unsupervised and semi-supervised feature selection algorithms. In the evaluation process, a supervised feature selection algorithm (Sikonja and Kononenko, 2003; Weston et al., 2003; Song et al., 2007; Zhang et al., 2008) determines features’ relevance by evaluating their correlation with the class or their utility for achieving accurate prediction, and without labels, an unsupervised feature selection algorithm may exploit data variance or data distribution in its evaluation of features’ relevance (Dash and Liu, 2000; Dy and Brodley, 2004; He et al., 2005). A semi-supervised feature selection algorithm (Zhao and Liu, 2007c; Xu et al., 2009) uses a small amount of labeled data as additional information to improve unsupervised feature selection.

Depending on how and when the utility of selected features is evaluated, different strategies can be adopted, which broadly fall into three categories: filter, wrapper and embedded models. To evaluate the utility of features, in the evaluation step, feature selection algorithms of filter model rely on analyzing the general characteristics of data and evaluating features without involving any learning algorithm. On the other hand, feature selection algorithms of wrapper model require a predetermined learning algorithm and use its performance on the provided features in the evaluation step to identify relevant feature. Algorithms of the embedded model, e.g., C4.5 (Quinlan, 1993), LARS (Efron et al., 2004), 1-norm support vector machine (Zhu et al., 2003), and sparse logistic regression (Cawley et al., 2007), incorporate feature selection as a part of the model fitting/training process, and features’ utility is obtained based on analyzing their utility for optimizing the objective function of the learning model. Compared to the wrapper and embedded models, algorithms of the filter model are independent of any learning model, therefore do not have bias associated with any learning models, one advantage of the filter model. Another advantage of the filter model is that it allows the algorithms to have very simple structure, which usually employs a straightforward search strategy, such as backward elimination or forward selection, and a feature evaluation criterion designed according to certain criterion. The benefit of the simple structure is two-folds. First, it is easy to design, and after it is implemented, it is also easy to understand for other researchers. This actually explains why most feature selection algorithms are of the filter model. And in real world applications, many most frequently used feature selection algorithms are also filters. Second, since the structure of the algorithms is simple, they are usually very fast. On the other hand, researcher also recognized that compared to the filter model, feature selection algorithms of the wrapper and embedded models can usually select features that result in higher learning performance for a particular learning model, which is used in the feature selection process. Comparing with the wrapper model, feature selection algorithms of embedded model are usually more
efficient, since they look into the structure of the involved learning model and use its properties to guide feature evaluation and search. In recent years, the embedded model is gaining increasing interests in feature selection research due to its superior performance. Currently, most embedded feature selection algorithms are designed by applying $L_0$ norm (Weston et al., 2003; Huang et al., 2008) or $L_1$ norm (Liu et al., 2009; Zhu et al., 2003; Zhao et al., 2010b) as a constraint to existing learning models to achieve a sparse solution. When the constraint is of $L_1$ norm form, and the original problem is convex, existing optimization techniques can be applied to obtain the unique global optimal solution for the regularized problem in a very efficient way (Liu et al., 2009).

Feature selection algorithms with the filter and embedded models may return either a subset of selected features or the weights (measuring features' relevance) of all features. According to the type of the output, feature selection algorithms can be divided into either feature weighting algorithms or subset selection algorithms. Feature selection algorithms of the wrapper model usually return feature subsets, therefore are subset selection algorithms. To the best of our knowledge, currently, most feature selection algorithms are designed to handle learning tasks with single data source. Researchers have started exploring the capability of using multiple auxiliary data and prior knowledge sources for multi-source feature selection (Zhao and Liu, 2008) to effectively enhance the reliability of relevance estimation (Lu et al., 2005; Zhao et al., 2008, 2010a).

Given a rich literature exists for feature selection research, a systematical summarization and comparison studies are of necessity to facilitate the research and application of feature selection techniques. Recently, there have been many surveys published to serve this purpose. A comprehensive surveys of existing feature selection techniques and a general framework for their unification can be found in (Liu and Yu, 2005). Guyon and Elisseeff (2003) reviewed feature selection algorithms from statistical learning point of view. In (Saeyes et al., 2007), the authors provided a good survey for applying feature selection techniques in bioinformatics. In (Inza et al., 2004), the authors reviewed and compared the filter with the wrapper model for feature selection. In (Ma and Huang, 2008), the authors explored the representative feature selection approaches based on sparse regularization, which is a branch of embedded feature selection techniques. Representative feature selection algorithms are also empirically evaluated in (Liu et al., 2002; Li et al., 2004; Sun et al., 2005; Lai et al., 2006; Ma, 2006; Swartz et al., 2008; Murie et al., 2009) under different problem settings and from different perspectives. We refer readers to these survey works to obtain comprehensive understanding on feature selection research.

2. Toward Cross-Discipline Collaboration in Feature Selection Research

Knowledge discovery and data mining (KDD) is a multidisciplinary effort. Researchers and practitioners in multiple disciplines and various IT sectors confront similar issues in feature selection, and there is a pressing need for continuous exchange and discussion of challenges and ideas, exploring new methodologies and innovative approaches. The international workshop on Feature Selection in Data Mining (FSDM) serves as a platform to further the cross-discipline, collaborative effort in feature selection research. FSDM 2005

and 2006\textsuperscript{2} were held with the SIAM Conference on Data Mining (SDM) 2005 and 2006, respectively. FSDM 2008\textsuperscript{3} was held with the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD) 2008. And FSDM 2010\textsuperscript{4} is the fourth workshop of this series, and is held at the 14th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD) 2010. This collection consists of one keynote and 8 peer-reviewed papers, among which, there are three on exploring novel problems in feature selection research; four on developing new feature selection algorithms or improving existing ones; two on designing effective algorithms to solve real-world problems. Below we give an overview on the papers of FSDM 2010.

Two novel feature selection research problems are investigated. In the keynote paper (Chawla, 2010), the author studies the interesting research problem of detecting feature dependence, which is also the topic of (Salehi et al., 2010). Both works are based on the techniques related to association rule mining. A concept that is closely related to feature dependence is feature interaction, in which a set of features cooperate with each other to define the target concept. The problem of feature interaction is studied in (Jakulin and Bratko, 2004; Zhao and Liu, 2007b). Besides detecting feature dependence, the problem of feature extraction for heterogeneous data with ontology information is also studied in (Gorodetsky and Samoylov, 2010). It is an interesting feature extraction problem related to information fusion and multi-source feature selection (Zhao and Liu, 2008).

The filter, wrapper, and embedded models are the major models used in feature selection for algorithm design. In (Esseghir, 2010), an interesting hybrid approach is proposed to combine the wrapper with the filter model through a so-called greedy randomized adaptive search procedure (GRASP). The advantage of the method is that it can inherit the strength of both models to improve the performance of feature selection. In (Jaiantilal and Grudic, 2010), a new feature selection algorithm based on the embedded model is proposed. The intrinsic point of the paper is to develop a random sampling framework, which can effectively estimate feature weights for weighted $L_1$ penalty based sparse learning models (Zou, 2006). The pairwise sample similarity is an important way to depict the relationships among samples, and has been widely used in designing feature selection algorithms (Zhao and Liu, 2007a). Improving the quality of similarity measurements is beneficial to the feature selection algorithms by taking sample similarity as their input. In (Zagoruiko et al., 2010), the authors propose to apply FRiS-function to improve similarity estimation. And in (Xie et al., 2010), the authors proposed to construct continuous variables from categorical features to achieve better similarity estimation.

Text mining is an important research area, where feature selection is widely applied for dimension reduction. In (Singh et al., 2010), the authors develop a new feature evaluation criterion for text mining based on Gini coefficient of inequality. Their empirical study shows that the proposed criterion significantly improve the learning performance compared to several existing criteria in feature selection, including mutual information, information gain and chi-square statistic. Besides text mining, feature extraction for outlier detection is also studied. In (Nguyen and Gopalkrishnan, 2010), the authors propose to use weight adjusted scatter matrices in feature extraction to address the class unbalance issue in outlier detection.

\begin{itemize}
  \item \textsuperscript{2} http://enpub.fulton.asu.edu/workshop/2006/
  \item \textsuperscript{3} http://www.psb.ugent.be/yvsae/fsdm08/index.html
  \item \textsuperscript{4} http://featureselection.asu.edu/fsdm10/index.html
\end{itemize}
Feature selection, and empirical results show that the proposed method can bring about nontrivial improvement over the existing algorithms.

3. Advancing Feature Selection Research

The current development in scientific research will lead to the prevalence of ultrahigh dimensional data generated from the high-throughput techniques (Fan et al., 2009) and the availability of many useful knowledge sources resulting from collective work of cutting-edge research. Hence one important research topic in feature selection is to develop computational theories that help scientists to keep up with the rapid advance of new technologies on data collection and processing. We also notice that there is a chasm between symbolic learning and statistical learning that prevents scientists from taking advantage of data and knowledge in a seamless way. Symbolic learning works well with knowledge and statistical learning works with data. Explanation-based learning is one such example that would provide an efficient way to bridge this gap. The technique of explanation-based feature selection will enable us to use the accumulated domain knowledge to help narrow down the search space and explain the learning results by providing reasons why certain features are relevant. Below are our conjectures about some interesting research topics in feature selection of potential impact in the near future.

Feature selection for ultrahigh dimensional data: selecting features on data sets with millions of features (Fan et al., 2009). As high-throughput techniques keep evolving, many contemporary research projects in scientific discovery generate data with ultrahigh dimensionality. For instance, the next-generation sequencing techniques in genetics analysis can generate data with several giga features on one run. Computation inherent in existing methods makes them hard to directly handle data of such high dimensionality, which raises the simultaneous challenges of computational power, statistical accuracy, and algorithmic stability. To address these challenges, researchers need to develop efficient approaches for fast relevance estimation and dimension reduction. Prior knowledge can play an important role in this study, for example, by providing effective ways to partition original feature space to subspaces, which leads to significant reduction on search space and allows the application of highly efficient parallel techniques.

Knowledge oriented sparse learning: fitting sparse learning models via utilizing multiple types of knowledge. This direction extends multi-source feature selection (Zhao and Liu, 2008). Sparse learning allows joint model fitting and features selection. Given multiple types of knowledge, researchers need to study how to use knowledge to guide inference for improving learning performance, such as the prediction accuracy, and model interpretability. For instance, in microarray analysis, given gene regulatory network and gene ontology annotation, it is interesting to study how to simultaneously infer with both types of knowledge, for example, via network dynamic analysis or function concordance analysis, to build accurate prediction models based on a compact set of genes. One direct benefit of utilizing existing knowledge in inference is that it can significantly increase the reliability of the relevance estimation (Zhao et al., 2010a). Another benefit of using knowledge is that it may reduce cost by requiring fewer samples for model fitting.

Explanation-based feature selection (EBFS): feature selection via explaining training samples using concepts generalized from existing features and knowledge. In many real-
world applications, the same phenomenon might be caused by disparate reasons. For example, in a cancer study, a certain phenotype may be related to mutations of either genes A or gene B in the same functional module M. And both gene A and gene B can cause the defect of M. Existing feature selection algorithm based on checking feature/class correlation may not work in this situation, due to the inconsistent (variable) expression pattern of gene A and gene B across the cancerous samples. The generalization step in EBFS can effectively screen this variation by forming high-level concepts via using the ontology information obtained from annotation databases, such as GO. Another advantage of EBFS is that it can generate sensible explanations to show why the selected features are related. EBFS is related to the research of explanation-based learning (EBL) and relational learning.

Feature selection remains and will continue to be an active field that is incessantly rejuvenating itself to answer new challenges.

Acknowledgments

This work is, in part, supported by NSF Grant (0812551).

References


5. For a cancerous sample, either gene A or gene B has abnormal expression, but not both.


Feature Selection: An Ever Evolving Frontier in Data Mining


Feature Selection, Association Rules Network and Theory Building

Sanjay Chawla
School of IT, University of Sydney
NSW 2006, Australia

Editor: Huan Liu, Hiroshi Motoda, Rudy Setiono, and Zheng Zhao

Abstract

As the size and dimensionality of data sets increase, the task of feature selection has become increasingly important. In this paper we demonstrate how association rules can be used to build a network of features, which we refer to as an association rules network, to extract features from large data sets. Association rules network can play a fundamental role in theory building - which is a task common to all data sciences - statistics, machine learning and data mining.

The process of carrying out research is undergoing a dramatic shift in the twenty first century. The cause of the shift is due to the preponderance of data available in all almost all research disciplines. From anthropology to zoology, manufacturing to surveillance, all domains are witnessing an explosion of data. The availability of massive and cheap data has opened up the possibility of carrying out data-driven research and data mining is the discipline which provides tools and techniques for carrying out this endeavour.

However much of these vast repositories of data generated are observational as opposed to experimental. Observational data is undirected and is often collected without any specific task in mind. For example, web servers generate a log of client activity. The web log can then be used for a myriad of tasks ranging from tracking search engine spiders to personalization of web sites. Experimental data, on the other hand, is directed and is generated to test a specific hypothesis. For example, to test the efficacy of a new drug, randomized trials are conducted and specific data is collected to answer very specific questions.

1. Feature Selection and Experimental Data

In order to appreciate the role of feature selection we first have to understand the role of experimental data in a scientific discovery process.

Taking a reductionist viewpoint, much of scientific discovery reduces to identifying relationship(s) between variables in a domain. For example, Einstein postulated that the relationship between energy and mass is governed by the equation $e = mc^2$. In order to validate the relationship, scientist will carry out experiments to test if the relationship is indeed true. The resulting data is called experimental data.

Scientist often also postulate relationship between variables which are not necessarily governed by a mathematical equation. For example, research has shown that there is a smoking is a leading cause of lung cancer. Trials are conducted to test the validity of
the relationship between the variable smoking and cancer. Experimental data does not have to be large and because there is an underlying theory which leads to an experiment the number of variables is also typically small. Thus feature selection, or the process of selecting variables which maybe related to a target variable is generally not necessary.

2. Feature Selection and Observational Data

As noted above, observational data is often collected with no specific purpose in mind. For example, a biologist maybe interested in determining which gene or a set of genes control certain physiological process \( P \). Now modern technology allows the ability to collect the expression levels of all genes in a genome. In this setting a feature selection exercise is often carried out to filter the candidate variables which correlate with the process \( P \). The reason that feature selection is generally hard and complex is because it is possible that complex relationships may exist between a set of features and the target \( P \). For example two features \( f_1 \) and \( f_2 \) maybe individually correlated with \( P \) but together they may not be. Or two features may not be related with \( P \) but together they may be related.

From a structural perspective, observational data tends to be large and high dimensional and experimental data is relatively small and low dimensional. An objective of feature selection is to shape observational data in order to extract potential relationship that may exist in the data.

However the ultimate objective of feature selection in data mining is for theory building. A theory is a set of postulates which explains a phenomenon. Whether we can learn or even begin to learn a phenomenon from data is a controversial idea.

However, as data is now being collected at unprecedented rates, data mining provides new opportunities to facilitate the learning of theories from data. This is an ambitious task because the existence of large (and high dimensional) data is neither necessary nor sufficient to explain or postulate a theory. Still, examples abound where an unexpected manifestation in raw or transformed data triggered an explanation of the underlying phenomenon of interest. Data is known to throw up “surprises” whether these can be systematically harnessed to explain the data generating process is to be seen.

3. Association Rule Mining

Association rule mining is a data mining task to find candidate correlation patterns in large and high dimensional (but sparse) observational data (Agrawal and Srikant, 1994).

Association rules have been traditionally defined in the framework of market basket analysis. Given a set of items \( I \) and a set of transactions \( T \) consisting of subsets of \( I \), an Association Rule is a relationship of the form \( A \Rightarrow B \) where \( A \) and \( B \) are subsets of \( I \) while \( s \) and \( c \) are the minimum support and confidence of the rule. \( A \) is called the antecedent and \( B \) the consequent of the rule. The support \( \sigma(A) \) of a subset \( A \) of \( I \) is defined as the percentage of transactions which contain \( A \) and the confidence of a rule \( A \rightarrow B \) is \( \frac{\sigma(A \cup B)}{\sigma(A)} \). Most algorithms for association rule discovery take advantage of the anti-monotonicity property exhibited by the support level: If \( A \subset B \) then \( \sigma(A) \geq \sigma(B) \).

Our focus is to discover association rules in a more structured and dense relational table. For example suppose we are given a relation \( R(A_1, A_2, \ldots, A_n) \) where the domain of
Figure 1: An example how a set of association rules with a singleton consequents can be represented as a backward hypergraph (B-graph)

$A$, \( \text{dom}(A_i) = \{a_1, \ldots, a_{n_i}\} \), is discrete-valued. Then an \emph{item} is an attribute-value pair \( \{A_i = a\} \). The ARN will be constructed using rules of the form

\[
\{A_{m_1} = a_{m_1}, \ldots, A_{m_k} = a_{m_k}\} \rightarrow \{A_j = a_j\} \text{ where } j \notin \{m_1, \ldots, m_k\}
\]

4. Directed Hypergraphs

In this section we briefly describe directed hypergraphs and there relationship with association rules.

A hypergraph consists of a pair \( H = (N, E) \) where \( N \) is the set of nodes and the set \( E \) is a subset of the power set \( 2^N \). Each element of \( E \) is called a hyperedge.

In a directed hypergraph the nodes spanned by a hyperedge \( (e) \) are partitioned into the head \( H(e) \) and the tail \( T(e) \). The hypergraph is called \emph{backward} if \( |H(e)| = 1 \) for all \( e \in E \) (Gallo et al., 1993; G. Ausiello and Nanni, 1990). We will only consider association rules whose consequent are singletons and therefore the set of single consequent association rules can be identified by a B-graph (i.e., a backward hypergraph).

**Example:** Consider the following set of association rules:

\[
\begin{align*}
    r_1 : & \quad \{b, c\} \rightarrow a \\
    r_2 : & \quad d \rightarrow b \\
    r_3 : & \quad c \rightarrow d \\
    r_4 : & \quad e \rightarrow c
\end{align*}
\]

These set of rules constitutes a B-graph and can be graphically represented as shown in Figure 1.
5. Association Rules Network

In this section we formally define an Association Rules Network (ARN). Details about the ARN, the algorithm to generate them, ARN properties and examples are given in (Pandey et al., 2009).

**Definition 1** Given a set of association rules $R$ and a frequent goal item $z$ which appears as singleton in a consequent of a rule $r \in R$. An association rule network, $ARN(R, z)$, is a weighted $B$-graph such that

1. There is a hyperedge which corresponds to a rule $r_0$ whose consequent is the singleton item $z$.
2. Each hyperedge in $ARN(R, z)$ corresponds to a rule in $R$ whose consequent is a singleton. The weight on the hyperedge is the confidence of the rule.
3. Any node $p \neq z$ in the ARN is not reachable from $z$.

6. Association Rules Network Process

We can use ARN as a systematic tool for feature selection. The steps involved are:

1. Prepare the data for association rule mining. This entails transforming the data into transactions where each transaction is an itemset. Data where variables are continuous-valued, will have to be discretized.

2. Select and appropriate support and confidence threshold and apply an association rule mining algorithm to generate the association rules. Note that ARNs are target driven so only those association rules are of interest which are directly or indirectly related to the target node. An association rule algorithm can be customized to generate only the relevant rules. Selecting the right support and confidence threshold is non-trivial. However, since our objective is to model the norm (rather than the exception), higher values of the threshold are perhaps more suitable.

3. Build the Association Rule Network. Details are provided in (Pandey et al., 2009). This step has several exceptions which need to be handled systematically. For example, what happens if for the given support and confidence there is no association rule generated with the target node as the consequent? In which case either the support or the confidence threshold or both have to be lowered. We may also choose to select the top-k rules (by confidence) for the given target node. The advantage here is that we don’t have to specify the confidence (or sometimes even the support) but now we have to specify the “top-k.” Another advantage is generally we can also use the “top-k” approach to find rules in higher levels of the ARN.

4. Apply a clustering algorithm on the ARN to extract the relevant features (in the context of the target domain). The ARN is essentially a directed hypergraph. The intuition is that first level nodes have an immediate effect on the target node while higher level nodes have an indirect influence. We can use a hypergraph clustering algorithm as illustrated in (Han et al., 1997).
Figure 2: The ARN Process for theory building. Association rules are generated and a target item is selected which serves as the goal node of the ARN. An ARN is a weighted B-graph. A hypergraph clustering algorithm is applied to the B-graph and each cluster represents one feature. The features are used as independent variables in a statistical model where the goal node is the dependent variable.
Figure 3: ARN for the Contact Lens data. Clearly tear production rate does not seem like a good feature.

5. The elements of the cluster are a collection of items (features) which are correlated. Choose one element of the cluster as the candidate feature. The number of clusters selected is a parameter and will require carefully calibration.

6. Build and test a statistical model (e.g., regression) to formally test the relationship between the dependent and the candidate variables.

7. ARN Examples

We give two examples of ARN and show how they can be used for feature selection.

7.1 Contact Lens Example

We use a relatively simple data set from the UCI archive (Blake and Merz, 1998) to illustrate how ARNs can be used for feature selection. The ARN for the Lenses data are shown in Figure 3. The dependent variable is whether a patient should be fitted with hard contact lenses, soft contact lenses or should not be fitted with contact lenses. There are four attributes. We built an ARN where the goal attribute is the class. Support and confidence was chosen as zero. It is clear that both ARNs (for hard and soft lenses) can be used to elicit features which are important to distinguish between the two classes.

7.2 Open Source Software Example

We have carried out an extensive analysis of the Open Source Software domain using ARN. Details can be obtained from (S.Chawla et al., 2003; Pandey et al., 2009). The OSS data was obtained to understand why certain software products available from sourceforge.net...
Figure 4: The ARN for Open Source data (Pandey et al. (2009)). The goal node is Downloads=high. After the ARN was formed, the directed hypergraph was clustered. The clusters constitute the features, and in this example, suggest meaningful features which may suggest an explanation of popular projects.

become popular. This is a cross-sectional study. Perhaps a future problem is to use ARNs for study of longitudinal data.

8. Conclusion

In this paper we have briefly illustrated a new framework that we have introduced to elicit candidate features for theory building in a domain. Association Rules Network (ARN) are built from association rules and can be used as first step to understand the interaction amongst variables in a domain. ARNs are particular suitable in settings where the data available is observational (as opposed to experimental).

References


Eui-Hong Han, George Karypis, Vipin Kumar, and Bamshad Mobasher. Clustering based on association rule hypergraphs. In *Proceedings SIGMOD Workshop Research Issues on Data Mining and Knowledge Discovery(DMKD ’97)*, 1997.


A Statistical Implicative Analysis Based Algorithm and MMPC Algorithm for Detecting Multiple Dependencies

Elham Salehi
Jayashree Nyayachavadi
Robin Gras

department of Computer Science
University of Windsor
Windsor, Ontario, N9B 3P4

Abstract

Discovering the dependencies among the variables of a domain from examples is an important problem in optimization. Many methods have been proposed for this purpose, but few large-scale evaluations were conducted. Most of these methods are based on measurements of conditional probability. The statistical implicative analysis offers another perspective of dependencies. It is important to compare the results obtained using this approach with one of the best methods currently available for this task: the MMPC heuristic. As the SIA is not used directly to address this problem, we designed an extension of it for our purpose. We conducted a large number of experiments by varying parameters such as the number of dependencies, the number of variables involved or the type of their distribution to compare the two approaches. The results show strong complementarities of the two methods.

Keywords: Statistical Implicative Analysis, multiple dependencies, Bayesian network.

1. Introduction

There are many situations in which finding the dependencies among the variables of a domain is needed. Therefore having a model describing these dependencies provides significant information. For example, which variable(s) affect(s) the other variable(s) may be very useful for the problem of selection of variables; decomposition of a problem to independent sub-problems; predicting the value of a variable depending on other variables to solve the classification problem; finding an instantiation of a set of variables for maximizing the value of some function, etc (A. Goldebberg, 2004; Y. Zeng, 2008).

The classical model used for the detection of dependencies is the Bayesian network. This network is a factorization of the probability distribution of a set of examples. It is well known that the construction of a Bayesian network from examples is a NP-hard problem, thus different heuristic algorithms have been designed to solve this problem (Neapolitan, 2003; E. Saheli, 2009). Most of these heuristics are greedy and/or try to reduce the size of the exponential search space by a filtering strategy. The filtering is based on some measures that aim to discover sets of variables that have high potentiality to be mutually dependent or independent.
Detecting Multiple Dependencies

These measures rely on an evaluation of the degree of conditional independency. However, other measures exist which are not based on conditional probability measurements that have the ability to discover dependencies. Using another measure that is not based on conditional dependencies can provide another perspective about the structure of dependencies of variables of a domain. Statistical Implicative Analysis (SIA) has already shown a great capability in extracting quasi-implications also called as association rules (R. Gras, 2008). We present a measure for multiple dependencies based on SIA and then use this measure in a greedy algorithm for solving the problem of multiple dependencies detection. We have compared our new algorithm for finding dependencies with one of the most successful conditional dependencies based heuristic introduced so far, MMPC (I. Tsamardinos, 2006). We have designed a set of experiments to evaluate the capacity of each of them to discover two kinds of knowledge: the fact that one variable conditionally depends on another one and the sets of variables that are involved in a conditional dependencies relation. Both of this information can be used to decompose the NP-hard problem of finding the structure of a Bayesian network into independent sub-problems and therefore can reduce considerably the size of corresponding search space.

This paper organized as follows: In the next section we describe the MMPC heuristic. In section 3 we present our SIA based measure and algorithm for finding multiple dependencies and the experimental results of the algorithms are presented in Section 4. Finally we conclude in section 5 with a brief discussion.

2. The MMHC Heuristic

Discovering multiple dependencies from a set of examples is a difficult problem. It is clear that this problem cannot be solved exactly when the number of variables approaches few dozens. However, for some problems, the number of variables can be several hundred or several thousand. Therefore, it is particularly important to have some methods to obtain an approximate solution with good quality. A local search approach is usually used in these problems. In this case the model of dependencies is built incrementally by adding or removing one or more dependencies at each step. The dependencies are chosen to be added or removed using a score that assesses the quality of the new model according to the set of examples (E. Saheli, 2009). In this approach the search space is exponential in terms of maximum number of variables on which a variable may depend. Therefore, there is a need to develop methods to increase the chances of building a good quality model without exploring the whole search space exhaustively. One possible approach is to use a less computationally expensive method to determine a promising subset of the search space on which we can subsequently apply a more systematic and costly method.

The final model is usually a Bayesian network in which the dependencies represent conditional independencies among variables. It is possible to build this model using information from other measures besides conditional probability. Indeed, the measurements in the first phase are used as a filter to eliminate the independent variables or bring the variables with shared dependencies together in several sub-groups. The second phase uses this filtered information to build a Bayesian network. The goal of our study is to compare the ability of two approaches for the detection of dependencies for the first phase. In this section a
measure based on conditional probability is described and in the section 4 this measure will be compared with a SIA based measure.

2.1 Definition and Notation

A Bayesian network is a tool to represent the joint distribution of a set of random variables. Dependency properties of this distribution are coded as a direct acyclic graph (DAG). The nodes of this graph are random variables and the arcs correspond to direct influences between the variables.

We consider a problem consisting of $n$ variables $v_1, v_2, \ldots, v_n$. Each variable $v_i$ can take any values in set $M_i = m_{i,1}, m_{i,2}, \ldots, m_{i,k}$. For the detection of dependencies a set of $N$ examples is available. Each example is an instantiation of each of the $n$ variables in one of $k$ possible ways.

$Par_i$, the set of all variables on which variable $v_i$ depends, is the parent set of $v_i$. Any $v_j \in Par_i$ is a parent of $v_i$ and $v_i$ is a child of $v_j$. A table of conditional probability distribution (CPD), also known as the local parameters, is associated for each node of the graph. This table represents the probability distribution $P(v_i|Par_i)$.

2.2 MMPC Approach

Although learning Bayesian networks might seem a very well-researched area and even some exact algorithms have been introduced for networks with less than 30 variables (M. Koivisto, 2004), applying them to many domains such as biological or social networks, faces the problem of high dimensionality. In recent years several algorithms have been devised to solve this problem by restricting the space of possible network structures using various heuristics (N. Friedman, 1999; I. Tsamardinos, 2006). One of these algorithms, which has a polynomial complexity is "Sparse Candidate" algorithm (N. Friedman, 1999). The principle of this method is to restrict the parent set of each variable assuming that if two variables are almost independent in the set of examples, it is very unlikely that they are connected in the Bayesian network. Thus, the algorithm builds a small fixed-size candidate parent set for each variable. A major problem of this algorithm is to define the size of the possible parent sets and another one is that the algorithm assumes a uniform sparseness in the network. More recently, another algorithm called Max-Min Hill Climber (MMHC) has been proposed to solve these two problems and obtain better results on a wider range of network structures (I. Tsamardinos, 2006). This algorithm, uses a constrained based method to discover possible parents-children relationships and then uses them to build a Bayesian network. The first step of this algorithm, the one we use in this section to detect dependencies, is called Max-Min Parent Children (MMPC). The MMPC algorithm uses a data structure called parent-children set, for each variable $v_i$ that contains all variables that are a parent or a child of $v_i$ in any Bayesian network faithfully representing the distribution of the set of examples. The definition of faithfulness can be found in (Neapolitan, 2003; I. Tsamardinos, 2006). MMPC uses $C^2$ statistical test (P. Spirtes, 2000) on the set of examples to determine the conditional independency between pairs of variables given a set of other variables. The MMPC algorithm consists of two phases. In the first phase, an empty set of candidate parents-children (CPC) is associated with $v_i$. Then it tries to add more nodes one by one to this set using MMPC heuristic. This heuristic selects the variable $v_j$ that maximizes
Detecting Multiple Dependencies

the minimum association with $v_i$ relative to current CPC and add this variable to it. The minimum association of $v_j$ and $v_i$ relative to a set of variables CPC is defined as below.

$$MinAssoc(v_i; v_j | CPC) = \arg\min_{S} Assoc(v_i; v_j | S)$$

for all subset $S$ of CPC.

$Assoc(v_i, v_j | S)$ is an estimate of the strength of the association between $v_i$ and $v_j$ knowing the CPC and is equal to zero if $v_i$ and $v_j$ are conditionally independent given the CPC. The function Assoc uses the p-value returned by the $G^2$ test of independence: the smaller the p-value the higher the association. The first phase of MMPC stops when all remaining variables are considered independent of $v_i$ given the subset of CPC. This approach is greedy, because a variable added in one step of this first phase may be unnecessary after other variables were added to the CPC. The second phase of MMPC tries to fix this problem by removing those variables in CPC which are independent of $v_i$ given a subset of the CPC. Since this algorithm looks for candidate parents-children set for each node, if node $T$ is in CPC of node $X$, node $X$ should also be in CPC of node $T$.

What is not clear about these methods are their capabilities to discover any kind of structures and how different conditional probabilities and structures of real networks influence on the quality of results. We present the result we have obtained using the MMPC algorithm on examples generated from various Bayesian networks in Section 4.

3. SIA Based Approach

Statistical Implicative Analysis (SIA) (R. Gras, 2008) is a data analysis method that offers a framework for extracting quasi-implications also called as association rules. In a dataset $D$ of $N$ instances, each instance being a set of $n$ Boolean variables, the implicative intensity measures to what extent variable $b$ is true if variable $a$ is true. The quality measure used in SIA is based on the unlikelihood of counter-examples where $b$ is false and $a$ is true. We are interested in the capabilities of SIA for finding multiple dependencies especially in situations that are difficult for conventional methods that are based on other measurements. For example, a situation in which two variables are independent but often take the same value in a large number of examples. We want to study the efficiency of SIA to refute the hypothesis of dependence by taking into account the counter examples. In order to use the SIA in general, some modifications are necessary. Indeed, we do not restrict ourselves to the binary variables and generalize the method for variables with higher cardinalities. We also want to be able to detect a situation where a combination of variables implies another variable, using an overall measure. In other word we want to measure one or more combinations of variables as the parents of a child variable. For example for variables $A$, $B$ and $C \in \{0, 1, 2\}$, we want to define a measure which is able to detect a dependency from $B$ and $C$ to $A$ because when $B = 0 \land C = 2$, $A = 1$ is abnormally frequent and when $B = 0 \land C = 0$, $A = 0$ is abnormally frequent. Current version of the SIA cannot be used for this purpose.

3.1 Definition and Notation

We use the following definitions and notations besides those presented in section 2.1. All the definitions presented here and the proofs for the rational of the measures and their properties can be found in (R. Gras, 2008). Let $Card(m_{i,j})$ be the number of times the
variable \( v_i \) takes the value \( m_{i,j} \) in \( N \) examples. \( \text{Card}(\overline{m}_{i,j}) \) is the number of times the variable \( v_i \) takes a value different from \( m_{i,j} \) and \( \text{Card}(m_{i_1,j_1}, m_{i_2,j_2}) \) the number of times the variable \( v_i \) takes the value \( m_{i_1,j_1} \) and variable \( v_i \) takes value \( m_{i_2,j_2} \) in \( N \) examples.

Let \( \pi_i \) be an instantiation of the parents of \( v_i \) chosen from \( \Pi_i \), the list of all combinations of instantiation of \( v_i \) parents. For example, in the previous example with the variables \( A, B \) and \( C \), \( \Pi_A = (0,0), (0,1), (0,2), (1,0), (1,1), (1,2), (2,0), (2,1), (2,2) \). If \( k = |M_i| \) then for each \( \pi_i \) \( |\Pi_i| = k|\Pi_i| \).

Let \( \text{Card}(\pi_i) \) be the number of times all parents of \( v_i \), \( \pi_i \) takes value \( m_{i,j} \). Then the measure \( q(\pi_i, m_{i,j}) \) is extended from SIA is

\[
q(\pi_i, m_{i,j}) = \frac{\text{Card}(\pi_i \wedge \overline{m}_{i,j}) - \text{Card}(\pi_i) \times \text{Card}(\overline{m}_{i,j})}{N}. 
\]

And the inclusion index \( i(\pi_i, m_{i,j}) \) for measuring the imbalances is extended from SIA is

\[
i(\pi_i, m_{i,j}) = (\hat{I}_{\alpha}^{m_{i,j}/\pi_i} - \hat{I}_{\alpha}^{\overline{m}_{i,j}/\pi_i})^{1/2}. 
\]

If we define function \( f \) as below

\[
f(a, b) = \frac{\text{Card}(a \wedge b)}{\text{Card}(a)} 
\]

Then

\[
\hat{I}_{\alpha}^{m_{i,j}/\pi_i} = 1 + ((1 - f(\pi_i, m_{i,j})) \log_2((1 - f(\pi_i, m_{i,j}))) + f(\pi_i, m_{i,j}) \log_2((f(\pi_i, m_{i,j})). 
\]

If \( \text{Card}(\pi_i \wedge \overline{m}_{i,j}) \in [0, \frac{\text{Card}(\pi_i)}{2}] \); otherwise, \( \hat{I}_{\alpha}^{m_{i,j}/\pi_i} = 0 \); and

\[
\hat{I}_{\alpha}^{\overline{m}_{i,j}/\pi_i} = 1 + ((1 - f(\overline{m}_{i,j}, \pi_i)) \log_2((1 - f(\overline{m}_{i,j}, \pi_i))) + f(\overline{m}_{i,j}, \pi_i) \log_2((f(\overline{m}_{i,j}, \pi_i)). 
\]

In above equations \( \alpha = 1 \). The score we try to maximize is

\[
s(\pi_i, m_{i,j}) = -i(\pi_i, m_{i,j}) \times q(\pi_i, m_{i,j}). 
\]

### 3.2 Extension of SIA

Unfortunately, the current SIA measure considers only one instantiation of the parent set at a time. If we want to consider all possible instantiations of the parent set we will obtain as many different dependency measures as there are different possible combination of instantiation. However, for each variable \( v_i \), we need a single measure that represents its degree of dependency with its parent set. Therefore we must consider all the combination of variables for \( \Pi_i \) and use the measures \( s(\pi_i, m_{i,j}) \), to see how they imply all the possible values of \( v_i \). Consequently we build a table \( T_i \) containing the set \( \Pi_{si} \) of measures \( s \) for all the combination of \( \Pi_i \) and \( M_i \) with size

\[
k \times |\pi_i| = k|\text{par}(v_i)|. 
\]

We tried various methods to combine the information of this table to a single measure. The simplest way is to consider just the maximum of \( \Pi_{si} \). Other possibilities are to take the average of \( \Pi_{si} \) or the average of the \( x\% \) of highest scores. We conducted many test with these approaches and none of them has yielded satisfactory results. In the first series of measures we considered the scores of one instantiation of \( \pi_i \), but different values of \( M_i \).
Detecting Multiple Dependencies

<table>
<thead>
<tr>
<th></th>
<th>B</th>
<th>C</th>
<th>A=0</th>
<th>A=1</th>
<th>A=2</th>
<th>Sup</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1.3</td>
<td>0.6</td>
<td>1.3</td>
<td>0.272</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0.4</td>
<td>0.2</td>
<td>0.5</td>
<td>0.5</td>
<td>0.45</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1.1</td>
<td>0</td>
<td>1.1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1: An example of table $T_i$ with $A,B$ and $C \in \{0, 1, 2\}$ and $A=\{(0, 0), (0, 1), (0, 2), (1, 0), (1, 1), (1, 2), (2, 0), (2, 1), (2, 2)\}$.

independently. What we want to detect is that a value of $\pi_i$ imply one specific instantiation of $v_i$ and we want that it is true for several different instantiations of $\pi_i$. Therefore a measure is needed to detect that $s$ is high for a couple $(\pi_i, m_{i,j})$ with $m_{i,j} \in M_i$ and low for all the others $m_{i,j} \in M_i$ and that it is true for several $\pi_i$. We have therefore defined a score which combine, for a given $\pi_i$, the maximum value $Sup_{\pi_i}$ of $s$ for all $m_{i,j} \in M_i$ and the entropy $E_{\pi_i}$ of $s$ for all the values $m_{i,j} \in M_i$.

$$\text{Sup}_{\pi_i} = \max(s(\pi_i, m_{i,j})) \text{ where } 1 \leq j \leq k,$$

$$E_{\pi_i} = -\sum_{j=1}^{k} p(s(\pi_i, m_{i,j})) \log(p(s(\pi_i, m_{i,j})) \log(k))$$

where

$$p(s(\pi_i, m_{i,j})) = \frac{s(\pi_i, m_{i,j})}{\sum_{j=1}^{k} s(\pi_i, m_{i,j})}.$$

For calculating a measure associated with a table $T_i$, we consider a set $H$ of those $\pi_i$ corresponding to the highest $x\%$ of $\text{Sup}_{\pi_i}$ values in the table. Then the score of the table is

$$S_{i,\text{Par}_{\pi_i}} = \frac{\sum_{\pi_i \in H} \text{Sup}_{\pi_i}}{\sum_{\pi_i \in H} E_{\pi_i}}.$$

This is the measure we want to maximize. Table 1 presents $T_A$ for the example with variables $A,B,C$. If you select the highest 20% Sup, only lines 1 and 3 will be selected and $S_A$ will be equal to 8.48. In the following section we give an algorithm that uses this measure to determine the major dependencies of a problem.

3.3 SIA Based Algorithm

In previous section we defined a measure $S_i$ for each variable $v_i$ knowing its parent set. To determine the dependencies of a problem we should consider different possible configurations of parent sets for all variables and choose the configuration that leads to a maximum total
Since the number of possible configurations is exponential in the number of variables, we need a heuristic approach. We chose a greedy approach for this heuristic. In the beginning of the algorithm we set the parent set of each variable to empty. Then at each step a new variable is chosen to be added to any of the parent sets using measure $S$. We stop adding variables when a fixed number of edges, $\text{maxEdge}$, has been added. The calculation of the table $T_i$ is also exponential in the number of parents of variables so we restrict the maximum number of parents for each variable to four. The next variable to be added to a parent set is chosen by comparing the highest score of four different tables. The algorithm is presented in Table 2. This algorithm avoids calculating the score for all combinations of $2$, $3$ and $4$ variables in a parent set. Only combinations that include $x$ parents can be selected to calculate the score with $x + 1$ parents. The variable structMax includes: the score of the variable regarding its parent set, the child variable and the candidate parent variable to be added to the parent set. After initialization, table $\text{max}_1$ contains a list of the scores in descending order of all the combinations including one parent and one child. So there is $n^2$ scores in it. Tables $\text{max}_2$, $\text{max}_3$ and $\text{max}_4$ are initially empty. They are used to store the scores of child-parents combination when there are $2$, $3$ and $4$ parents in the parent sets respectively. Thus at each stage of the algorithm, the variable to be added to the parent set of another variable will be determined by selecting the highest score of 4 tables. If $\text{Max}_i$ is the selected table, the parent set of the variable associated with the maximum score for this table goes from $i-1$ to $i$ variables. The score is then removed from the table and a new max score is calculated and inserted in the table $\text{max}_{i+1}$. The four tables are kept sorted in descending order so the maximum value of each table is always in position 0.

4. Experimental Study

In this section we study the capabilities of the MMPC heuristics and our SIA algorithm in finding the conditional dependencies and dependent variables involved in conditional dependencies.

4.1 Experimental Design

In our experiments, we use artificial data produced by sampling from randomly generated Bayesian networks. Each network has $A$ arcs and $n = 100$ variables divided into two sets: a set of $D$ variables for which there are direct dependency relations with at least one of the $n-D-1$ other variables; a set of variables $I$ with no dependency relationship with any of the other $n-1$ variables. The CPD of each variable is randomly generated taking into account the possible dependency relations. Each variable can take 3 different values.

We represent the distribution of independent variables as a triplet such $(p_1, p_2, p_3)$. For example $(80, 10, 10)$ means that each random variable has a probability of 0.8 for one of its three possible values, and a probability of 0.1 for the other two. The value with a probability of 0.8 is chosen randomly among the three random variables. For distributions called 'random', each variable has a different distribution $(p_1, p_2, p_3)$. 


for all \( V_i \)
\[ Par_i = \{ \emptyset \} \]
\[ structMax = \{0, 0, 0\} \]
\[ max_1 = \emptyset \]
for all \( V_i \) {
  for all \( v_j \neq v_i \) {
    if \((V_i, Par_i + v_i) > structMax.score\) {
      structMax.score = \(V_i, Par_i + v_i\)
      structMax.child = \(i\)
      structMax.parent = \(j\)
    }
  }
  \[ max_1 = max_1 + structMax \]
}
DescendSort \((max_1)\)
\[ max_2 = \emptyset, max_3 = \emptyset, max_4 = \emptyset \]
\[ nbEdge = 0 \]
while \((nbEdge < maxEdge)\) {
  \[ k = getIndexOfTableWithMaxScore(max_1, max_2, max_3, max_4) \]
  \[ enf = max[k][0].child \]
  \[ par_child = par_child + max[k][0].parent \]
  if \((k < 4)\) {
    structMax = \(\{0, 0, 0\}\)
    for all \( v_j \notin par_child \) {
      if \((V_i, Par_i + v_i) > structMax.score\) {
        structMax.score = \(V_i, Par_i + v_i\)
        structMax.child = \(i\)
        structMax.parent = \(j\)
      }
    }
    \[ max_{k+1} = max_{k+1} + structMax \]
  }
  DescendSort \((max_{k+1})\)
  \[ max[k][0] = \{0, 0, 0\} \]
  DescendSort \((max_k)\)
  \[ nbEdge = nbEdge + 1 \]
}

Table 2: SIA based Algorithm.

4.2 Evaluation of MMPC Heuristic
In this section, we study the ability of the MMPC algorithm to discover good parent-child sets of variables from data generated from Bayesian networks.

In our study, we vary the characteristics of the networks to analyze the consequences of this variation on the effectiveness of the MMPC algorithm. These changes include the distribution of independent variables \( I \), the number of dependent variables \( D \) and the number of dependencies among the variables \( D \) (i.e. the number of arcs \( A \) in the network). The results are presented in Tables 3 to 4. Each row of these tables represents an average of results for 10 different sets of examples generated from 10 different networks but with the same characteristics. In each experiment, we calculate the mean and standard deviation of the number of true positive (TP), False Positives (FP), False Negative (FN) and the computational time. TP is the number of parent-children relationships correctly predicted by the algorithm. Thus, the number of TP at most can be twice the number of arcs of the network because if there is an arc between node \( X \) and node \( T \) it means each of them
4.2.1 Finding the Dependencies

In this section, first we investigate the effects of the distribution of independent variables on the effectiveness of the MMPC algorithm. Bayesian networks used for this purpose include $I = 75$ independent variables and $D = 25$ dependent variables. The distribution used to generate the independent variables varies from almost uniform to completely random. The results are presented in Table 3. The number of arcs for all these networks is $A = 40$. One can see from these results that the distribution of independent variables has virtually no effect on the efficiency of the MMPC algorithm. The algorithm, under these conditions, was able to discover about 37% of dependencies. It may be noted that the number of FP is high, which means that the algorithm tends to predict many more dependencies than that really exists. In order to investigate the effect of the proportion of independent variables, we keep the ratio $A / D$ almost the same while changing the numbers $D$ and $I$ ($n$ remains equal to 100). As it can be seen from the results presented in the first three rows of Table 4, when the network contains only the dependent variables ($D = 100$), the MMPC algorithm performs much better and is able to find almost 80% of dependencies. However, where the number of dependent variables is equal to 25, only about 35% of the dependencies are discovered. The number of FP is also very low when all variables are dependent. It seems this method has difficulty in determining the independent variables. However, it can be noted that the run time increases considerably in the case where all variables are dependent. This can be problematic when the number of variables in the problem is much higher than 100.

If we vary the number of arcs in the networks with $n$ dependent variables ($D = n$, $I=0$), like in previous section, the TP is high. However, the percentage slightly decreases when the complexity of the networks increases. But it seems that the complexity is less important than the proportion of dependent and independent variables. Although, it should be noticed that the complexity of the network influences the computation time.

4.2.2 Problem of Selection of Variables

We mentioned in the introduction the possibility of methods that detect dependencies for the selection of variables involved in dependency relations. The idea is to decompose the original problem by locating the independent variables (those with empty candidate parent-children sets) for which the optimization can be performed independently. As the search space is reduced, the chance of finding a good quality solution is increased. The problem here

<table>
<thead>
<tr>
<th>Distribution of I</th>
<th>Average of TP</th>
<th>SD of TP</th>
<th>Average of FP</th>
<th>SD of FP</th>
<th>Average of FN</th>
<th>SD of FN</th>
<th>Run times (S)</th>
<th>precision= TP/(TP+FN)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(80, 10, 10)</td>
<td>30</td>
<td>7.29</td>
<td>116</td>
<td>11.33</td>
<td>49</td>
<td>7.28</td>
<td>6.6</td>
<td>37.5%</td>
</tr>
<tr>
<td>(60, 25, 25)</td>
<td>30</td>
<td>8.12</td>
<td>113</td>
<td>11.9</td>
<td>49</td>
<td>8.16</td>
<td>6.4</td>
<td>37.5%</td>
</tr>
<tr>
<td>(40, 30, 30)</td>
<td>29</td>
<td>7.28</td>
<td>117</td>
<td>11.63</td>
<td>51</td>
<td>7.28</td>
<td>6.7</td>
<td>36.25%</td>
</tr>
<tr>
<td>Random</td>
<td>29</td>
<td>6.76</td>
<td>118</td>
<td>10.54</td>
<td>50</td>
<td>6.75</td>
<td>6.2</td>
<td>36.25%</td>
</tr>
</tbody>
</table>

Table 3: Effectiveness of the MMPC algorithm according to the distribution of independent variables.
4.3 Evaluation of SIA Based Algorithm

We repeated the same experiences as those in Section 4.2 to evaluate our SIA based detection algorithm in order to achieve the most possible honest comparison. It should be noted though that this disadvantaged SIA. Indeed, the data were generated from the models, Bayesian networks, which are based on conditional probability measurement. The SIA approach uses an alternative measure that does not have the same properties. In particular, a very significant difference is that the Bayesian network model is not transitive while the SIA is. But a totally fair comparison is not possible and, taking into account these differences in our analysis, this comparison seemed to be the best way to proceed.
### 4.3.1 Finding the Dependencies

We use the same data as in section 4.2. Our algorithm uses several parameters: the percentage of best Sup, x for each table $T_i$ and the maximum number of variables to be added to all parent sets, maxEdge. For each of these parameters we used different values. Those we found most relevant and we presented here are 10% and 50% for x and 35, 50 and 150 edges for maxEdge parameter. We have evaluated three different configurations corresponding to a real situation in which we do not know the number of dependencies of the problem in advance. Actually we search slightly less, slightly more and much more dependencies that really exist by setting maxEdge to 35, 50 and 150 respectively. The results presented in Tables 6 and 7 indicate that our algorithm discovered few dependencies. The measure appears more sensitive to the distribution used to generate the independent variables. The results obtained with the value x = 10% is slightly better. The calculation time is also higher than the max-min algorithm, but our program has not yet been optimized for computational efficiency.

### 4.3.2 The Problem of Selection of Variables

We used the same data sets to test the ability of our algorithm to solve the problem of selection of variables involved in dependencies relation. The results are presented in Tables 8 and 9 and show a strong potential of our algorithm for this problem. The results are much better than those obtained with the max-min algorithm. Although the number of TP is slightly lower, the number of FP is considerably lower. What is most important is the fact that the level of prediction is much better than one would expect by chance. As the ratio of dependent variables to the number of independent variables is 1/3 in the model used to generate the data, a random prediction would give the same ratio of TP / FP (ie, in this case TP / (75-TN)). In Tables 8-9 in column TP / (0.33xFP), we present the gain compared to a random selection of variables. In the cases with distributions of independent variables (40, 30, 30) and (50, 25, 25) the gain is very significant, up to 16.1. For comparison, the results of the max-min algorithm show more stability, but a gain that never exceeds 1.18. Our algorithm seems to have more difficulty when the independent variables have extreme distributions, 'random' or (80, 10, 10). With x = 10% and when we search less dependencies that it really exists (35 Edges), the gain is always at least 1. Although this is a first version, our algorithm seems to have a very high potential to detect the dependent variables and thus to solve the problem of selection of variables. We also tested our algorithm on the data

### Table 6: Results based on distribution of I, x=10%

<table>
<thead>
<tr>
<th>Dist. of I &amp; maxEdge</th>
<th>Avg of TP</th>
<th>Avg of FP</th>
<th>Av of FN</th>
<th>P Run Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>(80, 10, 10), 35</td>
<td>0.9</td>
<td>34.1</td>
<td>39.1</td>
<td>2.25</td>
</tr>
<tr>
<td>(50, 25, 25), 35</td>
<td>7.8</td>
<td>27.2</td>
<td>32.2</td>
<td>19.5</td>
</tr>
<tr>
<td>Random, 35</td>
<td>0.6</td>
<td>34.4</td>
<td>39.4</td>
<td>1.5</td>
</tr>
<tr>
<td>(80, 10, 10), 50</td>
<td>0.2</td>
<td>49.8</td>
<td>39.8</td>
<td>0.5</td>
</tr>
<tr>
<td>(50, 25, 25), 50</td>
<td>5.1</td>
<td>49.9</td>
<td>34.9</td>
<td>12.7</td>
</tr>
<tr>
<td>Random, 50</td>
<td>0.4</td>
<td>34.6</td>
<td>39.6</td>
<td>1</td>
</tr>
<tr>
<td>(80, 10, 10), 150</td>
<td>0.2</td>
<td>49.8</td>
<td>39.8</td>
<td>0.9</td>
</tr>
<tr>
<td>(50, 25, 25), 150</td>
<td>0.2</td>
<td>43.8</td>
<td>33.8</td>
<td>15.5</td>
</tr>
<tr>
<td>Random, 150</td>
<td>0.5</td>
<td>49.5</td>
<td>39.5</td>
<td>1.25</td>
</tr>
</tbody>
</table>

### Table 7: Results based on distributions of I, x=50%

<table>
<thead>
<tr>
<th>Dist. of I &amp; maxEdge</th>
<th>Avg of TP</th>
<th>Avg of FP</th>
<th>Av of FN</th>
<th>P Run Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>(80, 10, 10), 35</td>
<td>0.2</td>
<td>34.8</td>
<td>39.8</td>
<td>0.5</td>
</tr>
<tr>
<td>(50, 25, 25), 35</td>
<td>7.7</td>
<td>27.5</td>
<td>32.5</td>
<td>19.25</td>
</tr>
<tr>
<td>Random, 35</td>
<td>0.4</td>
<td>34.6</td>
<td>39.6</td>
<td>1</td>
</tr>
<tr>
<td>(80, 10, 10), 50</td>
<td>0.2</td>
<td>49.8</td>
<td>39.8</td>
<td>0.9</td>
</tr>
<tr>
<td>(50, 25, 25), 50</td>
<td>5.1</td>
<td>49.9</td>
<td>34.9</td>
<td>12.7</td>
</tr>
<tr>
<td>Random, 50</td>
<td>0.4</td>
<td>34.6</td>
<td>39.6</td>
<td>1</td>
</tr>
<tr>
<td>(80, 10, 10), 150</td>
<td>0.2</td>
<td>49.8</td>
<td>39.8</td>
<td>0.9</td>
</tr>
<tr>
<td>(50, 25, 25), 150</td>
<td>0.2</td>
<td>43.8</td>
<td>33.8</td>
<td>15.5</td>
</tr>
<tr>
<td>Random, 150</td>
<td>0.5</td>
<td>49.5</td>
<td>39.5</td>
<td>1.25</td>
</tr>
</tbody>
</table>

32
Table 8: Results for selection of variables, x=10%.

<table>
<thead>
<tr>
<th>Dist. of I, maxEdge</th>
<th>Avg of TP</th>
<th>Avg of TN</th>
<th>TP/(0.33xFP)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(80, 10, 10), 35</td>
<td>6.7</td>
<td>55.5</td>
<td>1.03</td>
</tr>
<tr>
<td>(50, 25, 25), 35</td>
<td>15.5</td>
<td>71.8</td>
<td>14.7</td>
</tr>
<tr>
<td>(40, 30, 30), 35</td>
<td>13.4</td>
<td>72.4</td>
<td>9.6</td>
</tr>
<tr>
<td>random, 35</td>
<td>3.4</td>
<td>64.7</td>
<td>1</td>
</tr>
<tr>
<td>(80, 10, 10), 50</td>
<td>6.8</td>
<td>36.4</td>
<td>0.54</td>
</tr>
<tr>
<td>(50, 25, 25), 50</td>
<td>15.3</td>
<td>68.1</td>
<td>8.7</td>
</tr>
<tr>
<td>(40, 30, 30), 50</td>
<td>12.3</td>
<td>89.9</td>
<td>10.9</td>
</tr>
<tr>
<td>random, 50</td>
<td>6.4</td>
<td>61.3</td>
<td>1.36</td>
</tr>
<tr>
<td>(80, 10, 10), 150</td>
<td>9.2</td>
<td>13</td>
<td>0.45</td>
</tr>
<tr>
<td>(50, 25, 25), 150</td>
<td>22.8</td>
<td>31.8</td>
<td>1.9</td>
</tr>
<tr>
<td>(40, 30, 30), 150</td>
<td>21.8</td>
<td>46.7</td>
<td>3.33</td>
</tr>
<tr>
<td>random, 150</td>
<td>17.3</td>
<td>32.6</td>
<td>1.24</td>
</tr>
</tbody>
</table>

Table 9: Results for selection of variable, x= 50%.

<table>
<thead>
<tr>
<th>Dist. of I, maxEdge</th>
<th>Avg of TP</th>
<th>Avg of TN</th>
<th>TP/(0.33xFP)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(80, 10, 10), 35</td>
<td>7.6</td>
<td>60.4</td>
<td>1.58</td>
</tr>
<tr>
<td>(50, 25, 25), 35</td>
<td>16.8</td>
<td>66.2</td>
<td>7.79</td>
</tr>
<tr>
<td>(40, 30, 30), 35</td>
<td>14.8</td>
<td>67</td>
<td>1.85</td>
</tr>
<tr>
<td>random, 35</td>
<td>4</td>
<td>57</td>
<td>0.67</td>
</tr>
<tr>
<td>(80, 10, 10), 50</td>
<td>8</td>
<td>54.5</td>
<td>1.15</td>
</tr>
<tr>
<td>(50, 25, 25), 50</td>
<td>18.3</td>
<td>59.9</td>
<td>3.67</td>
</tr>
<tr>
<td>(40, 30, 30), 50</td>
<td>15.1</td>
<td>63.9</td>
<td>1.85</td>
</tr>
<tr>
<td>random, 50</td>
<td>5.6</td>
<td>50</td>
<td>0.67</td>
</tr>
<tr>
<td>(80, 10, 10), 150</td>
<td>11.8</td>
<td>6.7</td>
<td>0.52</td>
</tr>
<tr>
<td>(50, 25, 25), 150</td>
<td>22.1</td>
<td>23.5</td>
<td>1.28</td>
</tr>
<tr>
<td>(40, 30, 30), 150</td>
<td>22</td>
<td>41.2</td>
<td>1.97</td>
</tr>
<tr>
<td>random, 150</td>
<td>22</td>
<td>16.5</td>
<td>1.14</td>
</tr>
</tbody>
</table>

presented in section 2.3.2 in which D = 50, I = 50 and A = 80 (results not presented here).
The results show that with configuration x = 10%, the gains are between 1.28 and 1.82.

5. Conclusion

We conducted a study on the capabilities of two methods based on different measures for
discovering the dependencies of a problem: 1) the max-min algorithm, which is based on
the test of conditional dependency $G^2$; 2) an algorithm that we developed based on an
extension of the SIA measure. We applied these algorithms to several datasets by varying
the parameters of the problem such as the distribution of independent variables, the number
of dependent variables and the number of dependencies. We also considered two different
problems: to determine the dependencies relations and to identify the variables involved in
the dependency relationships. Of course finding a solution for the first problem can also
solve the second. However, it is generally not possible to directly and fully resolve this
problem. Being able to see at first just what is the subset of variables involved in the set
of dependencies reduces the complexity of the first problem and thus help to reach a better
solution.

Our results showed a good efficiency of the max-min algorithm for discovering the depen-
dendencies when all the variables of the problem are involved. The algorithm appears to
be little affected by the change in the complexity of the model and the distributions of the
independent variables. However, it has some significant limitations to detect dependencies
when part of the variables is independent. The algorithm max-min does not appear to
be effective for the second problem: the selection of variables. Our SIA based algorithm,
does not seem capable of directly detecting the dependencies whatever the configuration
was. But it seems very effective to determine the dependent variables. However, it is less
efficient in situations where the independent variables have extreme distributions like (80,
10, 10) or ‘random’. The two approaches seem complementary and promising. It would
be very interesting to develop a method combining these two approaches. In a first phase
our algorithm, using the extended version of the SIA, would select a subset of variables for
which there is a strong presumption of dependency. Then, in a second phase, the max-min
approach is applied to this sub-set to determine more precisely where these dependencies
are. All these information would then be used to build a Bayesian network. It would be
also interesting to compare the methods based on the importance of the dependencies using
some connection strength (Ebert-Uphoff, 2007) measure instead of just counting the num-
ber of discovered dependencies. It would be also interesting to compare the modified SIA
with multi-dimensional form of classical measures to detect correlation between variable distributions.

References


Attribute Selection Based on FRiS-Compactness

Nikolai G. Zagoruiko
Institute of Mathematics SD RAS, pr. Koptyg 4,
Novosibirsk 630090, Russia
ZAG@MATH.NSC.RU

Irina A. Borisova
Institute of Mathematics SD RAS, pr. Koptyg 4,
Novosibirsk 630090, Russia
BIAMIA@MAIL.RU

Vladimir V. Duybanov
Novosibirsk State University Pirogov st. 2,
Novosibirsk 630090, Russia
VLADIMIR.DYUBANOV@GMAIL.COM

Olga A. Kutnenko
Institute of Mathematics SD RAS, pr. Koptyg 4,
Novosibirsk 630090, Russia
OLGA@MATH.NSC.RU

Editor: Huan Liu, Hiroshi Motoda, Rudy Setiono, and Zheng Zhao

Abstract

Commonly to classify new object in Data Mining one should estimate its similarity with
given classes. Function of Rival Similarity (FRiS) is assigned to calculate quantitative mea-
sure of similarity considering a competitive situation. FRiS-function allows constructing
new effective algorithms for various Data Mining tasks solving. In particular, it enables
to obtain quantitative estimation of compactness of patterns which can be used as indi-
rect criterion for informative attributes selection. FRiS-compactness predicts reliability of
recognition of control sample more precisely, than such widespread methods as One-Leave-
Out and Cross-Validation. Presented in the paper results of real genetic task solving confirm
efficiency of FRiS-function using in attributes selection and decision rules construction.

Keywords: Pattern recognition, Function of Rival Similarity, Compactness, Informative-

1. Introduction

Two main parts “engine” and “criterion” can be partitioned in attribute selection algo-
rithms. Engine forms different variants of attributes subsystems and criterion estimates
quality of considered systems. We guess that the main element of the engine is algorithm
of directed search, the main characteristic calculated by criterion is compactness of the pat-
terns, and the basic element in compactness estimation is a measure of objects similarity.
These three items are considered in this work. In Section 2 relative measure of similarity
(FRiS-function) is introduced, measure of compactness of patterns based on FRiS-function
is defined in Section 3, in Section 4 algorithm for decision rule construction FRiS-Stolp is de-
scribed. Then, in Section 5 algorithm FRiS-GRAD for simultaneous attribute selection and
decision rule construction is proposed. Effectiveness of this algorithm on real recognition
task is illustrated in Section 6.

©2010 Zagoruiko, Borisova, Duybanov and Kutnenko.
2. How to estimate similarity?

Similarity of two objects $z$ and $a$ according to some attribute $X$ in metric space usually depends on difference $R(z, a)$ between values of $X$ for these objects. Many such type measures of similarity are described in the literature. Common peculiarity of these measures is that similarity is considered as absolute category. Similarity of objects $z$ and $a$ does not depend on the similarities of them with other objects.

But the measure of similarity, used in pattern recognition is not so primitive. Classifying object $z$ as a member of pattern $A$ in case of two patterns it is important to know not only similarity of $z$ with the $A$, but similarity of $z$ with the rival pattern $B$, and compare these values. Consequently, the similarity in pattern recognition is not absolute but relative category. To answer the question “How much is $z$ similar to $a$?” you need to know “In comparison with what?” Adequate measure of similarity should reflect relative nature of similarity, which depends on rival situation in the neighborhood of object $z$.

All statistical recognition algorithms take into account competition between classes. If probability density of class $A$ in point $z$ is equal to $P_A$, and probability density of class $B$ in this point is $P_B$, then $z$ is classified, for example, as member of class $A$ not because value $P_A$ exceeds certain threshold, but because $P_A > P_B$.

When distributions of classes are unknown or the number of attributes is higher than number of objects in training dataset, recognition methods based on probability densities are inapplicable. In these cases distances $R(z, a_i)$ from object $z$ to standards (representatives) of patterns $A_i$, $i = 1, \ldots, K$, ($K$ is the number of patterns) are commonly used. As a result object $z$ is classified as a member of pattern distance to which standard is less than the distances to the standards of other patterns. For example, in the method “$k$ nearest neighbours” (kNN) (Fix & Hodges, 1951) new object $z$ is recognized as object of pattern $A$ if the distance to this pattern, equal to average distance from $z$ to its $k$ nearest neighbours from pattern $A$, is smaller than the distance to the nearest rival pattern $B$. Similarity in this algorithm is considered in scale of order.

Measures of rival similarity in strong scales are used in pattern recognition as well. Quantitative estimation of rival similarity was proposed by Kira and Rendell (1992) in the algorithm RELIEF. To determine similarity of object $z$ with the object $a$ in competition with the object $b$ value $W(z, a|b)$ is calculated, which depends on the difference between distances $R(z, a)$ and $R(z, b)$ to competitors in an explicit form:

$$W(z, a|b) = \frac{R(z, b) - R(z, a)}{R_{\text{max}} - R_{\text{min}}}.$$

There $R_{\text{min}}$ and $R_{\text{max}}$ are the minimum and maximum distances between objects of the analyzed dataset. Normalization by the difference $(R_{\text{max}} - R_{\text{min}})$ has some weaknesses. If dataset consists of only two objects $a$ and $b$, then value of similarity of any new object $z$ with them can not be defined because the denominator is equal to 0. The same problem appears, if dataset consists of three points located in vertex of equilateral triangle. In addition, value $W(z, a|b)$ strongly depends on distant objects and can vary in considerable range when membership of training dataset is changed. Thus, this measure has absolute quality only within single task, its values in different tasks can not be compared to each over directly. We would like to obtain better measure of similarity. Specify properties that should have this measure:
1. **Locality.** Measure of similarity should depend on distribution of objects in the neighborhood of object \( z \), not entire dataset.

2. **Normalizing.** While measure of similarity of object \( z \) with the object \( a \) in competition with object \( b \) \((a \neq b)\) is estimated, if \( z \) coincides with the object \( a \), value \( F(z, a|b) \) should amount to its maximal value equal to 1, if \( z \) coincides with the \( b \) its similarity \( F(z, a|b) \) should be minimal and equal to \(-1\). In other cases rival similarity takes values in range between \(-1\) and 1.

3. **Antisymmetry.** In all cases \( F(z, a|b) = -F(z, b|a) \). If distances \( R(z, a) \) and \( R(z, b) \) from object \( z \) to objects \( a \) and \( b \) are equal then \( z \) is equally similar (and not similar) to both that objects and \( F(z, a|b) = F(z, b|a) = 0 \).

4. **Invariance.** Values \( F(z, a|b) \) and \( F(z, b|a) \) should be invariant under such attributes system transformation as moving, rotating and extending all coordinate axes with the same coefficient. Any sigmoid function obeys these conditions. We propose the following simple version of this function:

\[
F(z, a|b) = \frac{R(z, b) - R(z, a)}{R(z, b) + R(z, a)}.
\]

This kind of similarity measure we called FRiS-function (Function of Rival Similarity) (Zagoruiko et al., 2008a). Function \( F(z, a|b) \) is invariant under coordinate system moving, rotating and extending all coordinate axes with the same coefficient. But extending coordinate axes with different coefficients is changed effect of individual characteristics on similarity estimation. So, the similarity between objects depends on weights of their attributes. Changing these weights one can boost similarity or difference between the specified objects, or subsets. Such technique is ordinary used in pattern recognition. After weights fixing FRiS-function measures the similarity in absolute scale: its interpretation is changed by adding some coefficient over than 0 to value of \( F(z, a|b) \), or multiplying \( F(z, a|b) \) by any value other than 1.

It turned out that the additional information providing by absolute scale in comparing with the order scale allows to significantly improve methods of Data Mining.

3. **Measure of compactness of patterns**

Almost all recognition algorithms are based on compactness hypotheses (Braverman, 1962). Definitions of compactness presented in literature operate such no-formalized terms as “sufficiently extensive neighborhood”, “not too complex border”, and so on. We are interested in quantitative measure of compactness directly correspondent with expected reliability of recognition.

Main idea such kind measure proposed by Vorontsov and Koloskov (2006) is in the compactness profile calculation. Compactness profile is function \( V(j) \) equals to share of samples which \( j \)-th neighbor is the object of another class. Compactness profile is the formal expression of the basic idea of compactness hypotheses stating that similar objects more often lie in the same class than in different. The simpler task is, i.e. the more often close objects appears to be in the same class, the closer to the zero start part of profile \( V(j) \) is. In complex tasks or in spurious attribute subsystems all parts of profile \( V \) are close to 0.5 or another constant value depending on prior probabilities of patterns.
Only order between objects in dataset is important during compactness profile calculating. Consequently, if the patterns do not intersect, the profile reminds the same when variances of the patterns or the distance between them are changed.

We are interested in quantitative measure of compactness, which allows estimating as single pattern compactness, as compactness of whole system of patterns in dataset taking into account any changes in the variances and distances between patterns. Such a measure should answer following requirements:

1. **Universality.** Measure should allow correctly evaluating the compactness of patterns with any types of probability distributions, as for each pattern individually as for the entire set of recognized patterns.

2. **Sensitivity.** Value of compactness should increase with the area of pattern intersection decreasing. In case of disjoint patterns compactness should depend on both the variance of the patterns and the distance between them.

3. **Normalizing.** Measure of compactness should take values in range between \(-1\) (in case of full coincidence of patterns) and 1 (in case of infinite distance between patterns).

4. **Invariance.** Measure should be invariant under coordinate system moving, rotating and extending all coordinate axes with the same coefficient.

Using of rival similarity (FRiS-functions) allows to determining quantitative measure of compactness, meeting the specified requirements. This procedure is based on calculation of the similarity of objects from the same pattern with each over and distinctiveness (differences) of these objects with the objects of other patterns. The algorithm of FRiS-compactness calculation in case of two patterns is illustrated on Figure 1 and acts as follows:

1. Similarity of each object \(a_j, j = 1, \ldots, M_A\), of pattern \(A\) with some fixed object \(a_i\) of the same pattern in competition with nearest to \(a_j\) object \(b_j\) of rival pattern \(B\) is calculated on distances \(R(a_j, a_i)\) and \(R(a_j, b_j)\) from \(a_j\) to objects \(a_i\) and \(b_j\) by the next formula:

\[
F(a_j, a_i | b_j) = (R(b_j, a_i) - R(a_j, a_i))/(R(b_j, a_i) + R(a_j, a_i)).
\]

These values are added to the counter \(C_i\).

---

**Figure 1:** Rival similarities of objects of patterns \(A\) and \(B\) with fixed object \(a_i\).
2. Distinctiveness of each object \( b_q, q = 1, \ldots, M_B \), of pattern \( B \) with the object \( a_i \) is calculated, as rival similarity of \( b_q \) with its nearest neighbor from the same pattern in competition with the \( a_i \). So two distances are needed: distance \( R(b_q, b_s) \) from object \( b_q \) to its nearest neighbor \( b_s \) from pattern \( B \), and distance \( R(b_q, a_i) \) from object \( b_q \) to object \( a_i \). Calculated on these distances measures of rival similarity \( F(b_q, b_s|a_i) \) for all objects \( b_q \) of pattern \( B \) are added to the counter \( C_i \). To averaging value \( C_i \) is divided by number of objects in dataset \((M_A + M_B)\). Resulting value \( \overline{C_i} \) characterizes similarity of “own” objects and distinctiveness of “anothers” objects with fixed object \( a_i \).

3. After calculating values \( \overline{C_i} \) for all objects \( a_i, i = 1, \ldots, M_A \), of pattern \( A \) average value \( G_A \) of pattern \( A \) compactness is calculated:

\[
G_A = \frac{1}{M_A} \sum_{i=1}^{M_A} \overline{C_i}.
\]

4. The same way compactness of pattern \( B \) can be estimated. If number of patterns in task is larger than two, then to define compactness of \( k \)-th pattern, \( k = 1, \ldots, K \), this pattern is considered as pattern \( A \), and all other patterns associate in pattern \( B \).

5. General value of compactness \( G \) of all \( K \) patterns estimated on given dataset is calculated as geometrical mean of all \( G_k \):

\[
G = \left( \prod_{k=1}^{K} G_k \right)^{\frac{1}{K}}.
\]

The lower the variances of patterns and the higher distances between them are, the higher value of compactness \( G \) becomes. Measure of compactness proposed by Fisher to calculate informativeness of attribute subsystems has the same peculiarity. The difference is that the Fishers measure is designed for patterns with Gaussian distributions, and measure of FRiS-compactness is applied to arbitrary distributions.

Our experiments with using proposed measure of compactness as a criterion for informative attributes selection (Zagoruiko, 2009) demonstrate its higher efficiency in comparing with widely used criterion based on number of objects of training dataset unrecognized by KNN rule in the mode One-Leave-Out (OLO). These two criteria \( G \) and OLO are compared in following experiment.

The initial data included 200 objects belonging to two patterns (100 objects for each pattern) in the 100-dimensional space. Attributes were generated in such a way that they have different informativeness. As a result, about 30 attributes were more or less informative, whereas other attributes generated by random number generator were certainly spurious. In addition, the dataset was distorted by noises with different intensity. For every pattern, 35 randomly chosen objects were selected for training. Other 130 objects formed the test dataset. At every noise level (from 0.05 to 0.3), the most informative subsystems were selected. The recognition reliability in selected attributes subsystems averaging over 10 experiments for each noise level is presented on Figure 2. Thin lines correspond to results on training subsets, bold lines – to results on test subsets.

Attribute systems, selected with OLO, have high reliability estimations (upper dotted line) on training datasets independently of noise level. But their quality on test datasets
Attributes selected with FRiS-compactness criterion $G$ have more realistic reliability estimations (upper solid line) which are confirmed on test samples (lower solid bold line).

Higher stability of criterion $G$ can be explained as follows. Most of objects affected by noise are situated far from central part of pattern on the patterns bounds. Measure OLO depends on samples from the bound between patterns, while measure $G$ is based on all objects of the pattern.

![Figure 2: Results of training and test recognition with $G$ and OLO criteria.](image)

**4. Algorithm FRiS-Stolp for set of representatives forming**

To classify test object $z$ values of rival similarities of the $z$ with typical representatives (stolps) of patterns are used. Algorithm FRiS-Stolp (Zagoruiko et al., 2008b) selects objects with *defensive capability* (high similarity with other objects from the same pattern allows recognizing that objects) and *tolerance* (low similarity with the objects of other patterns prevents their unrecognizing as “own”) to use as stolps.

This procedure is realized as follows:

1. Some object $a_i$ of pattern $A$ is tested as a single stolp of this pattern. As in compactness estimation similarity of each object $a_j$, $j = 1, \ldots, M_A$, of pattern $A$, and distinctiveness of each object $b_q$, $q = 1, \ldots, M_B$, of pattern $B$ with $a_i$ are calculated and added to counter $C_i$. Averaging value $\overline{C_i}$ is considered as efficiency of object $a_i$ in a role of the stolp of pattern $A$.

2. Step 1 is repeated for all objects of pattern $A$. Object $a_i$ which provides maximum value $\overline{C_i}$ is selected as the first stolp of pattern $A$. All $m_1$ objects of pattern $A$, which similarity with this stolp is higher than $F^*$ (for example, $F^* = 0$), form first cluster and are eliminated from pattern $A$. Average value of similarity of objects from cluster $Q_1$ with the stolp of the cluster is used as compactness of this cluster estimation.
3. If \( m_1 < M_A \) steps 1–2 are repeated on remaining objects of pattern \( A \). As a result list of \( k_A \) stolps of pattern \( A \) with values of corresponded clusters compactnesses \( Q_j, j = 1, \ldots, k_A \), is obtained.

4. Average weighted value of compactnesses of all clusters of pattern \( A \):

\[
G'_A = \frac{1}{k_A} \sum_{j=1}^{k_A} Q_j m_j.
\]

can be used as compactness of pattern \( A \) estimation. In contrast to \( G_A \) this value characterizes quality of description of pattern \( A \) by the system of stolps. In our algorithms for inconclusive attributes elimination less labour-intensive criterion \( G_A \) is used. But for more precise attribute subsystem selection criterion \( G'_A \) appears to be more effective.

5. Steps 1–4 are repeated for pattern \( B \) to construct list of \( k_B \) stolps of this pattern. If number of patterns in task is larger than two, then technique described in previous section is used.

In case of Gaussian distributions, for example, the most typical objects of the patterns are selected by algorithm FRiS-Stolp at the points of statistical expectations. In case of multimodal distributions and linearly inseparable patterns stolps are placed at the centers of the modes (at the centers of areas of local concentrations of objects). With growing distribution complexity the number of stolps increases.

The decision rule consists of the list of objects-stolps and procedure of calculation of similarity of control object \( z \) with all stolps. Object \( z \) is classified as a member of pattern similarity with which stolp is maximal. Value of rival similarity can be used as estimation of reliability of object \( z \) recognition.

5. Attributes subsystems forming. Algorithm FRiS-GRAD

There are many variants of “engines” to select \( n \) most informative attributes among base set of \( N \) attributes. Main ideas of two basic greedy approaches (forward and backward searches) are used in our engine. Backward elimination (algorithm Deletion) offered by Merill and Green (1963) increases attribute subsystem quality as much as possible with each deletion of attributes. Forward selection (algorithm Addition) offered by Barabash et al. (1963) achieves this aim with each inclusion of attributes. In algorithm AdDel (Zagoruiko, 1999) next combination of these approaches is used: at first, \( n_1 \) informative attributes are selected by method Add. Then \( n_2 \) worst of them (\( n_2 < n_1 \)) are eliminated by method Del. Number of attributes in selected subset after that two steps is equal to \( (n_2 - n_1) \). Such consecution of actions (algorithms Add and Del) is repeated until quality of selected attributes is maximum. Analysis of subsystems with different number of attributes shows that on first steps while the number of attributes increases the quality increases too. But at some moment when all informative attributes are in selected subsystem already, the quality becomes decreasing after adding redundant or not relevant attributes. Inflection on the curve of quality allows specifying optimum number of attributes.

In algorithm GRAD (Zagoruiko, Borisova & Kutnenko, 2005) (“Granulated AdDel”) method AdDel works on set of most informative “granules”. Each granule consists of \( w \) attributes (\( w = 1, 2, 3 \)). In list of one-dimensional granules \( m_1 \) “best” according to their individual informativeness attributes (\( m_1 < N \)) are included.
Exhaustive search among all possible pairs and triplets of \( m_1 \) attributes is used for \( m_2 \) two-dimensional and \( m_3 \) three-dimensional granules forming. Among them \( m \) most informative granules are selected and used as an input of algorithm AdDel. Comparing effectiveness of algorithms AdDel and GRAD on different tasks showed, that algorithm GRAD was much better than algorithm AdDel.

To find the best subsystem of attributes and effective decision rule algorithm FRiS-GRAD (Zagoruiko et al., 2008b) uses procedure of directed search, offered in algorithm GRAD. On each step some variant of attribute subsystem is formed and then algorithm FRIS-Stolp is started to construct set of stolps and to calculate FRiS-quality \( G' \) of the subsystem. If \( t \)-dimension subsystem of attributes has been selected and \( q \) next steps of algorithm GRAD working quality of decision was decreasing then this subsystem is considered as most informative and set of stolps in this subspace determines the rule to classify new objects.

6. Recognition of two types of leukemia with algorithm FRiS-GRAD

Efficiency of offered algorithm was demonstrated on task for medical diagnosis (two types of leukemia recognition). This task was interesting for us because results of its solving by different researchers were published and we could compare effectiveness of our algorithm with competitors. In the work (Guyon et al., 2002) the best in the world at the moment of the publication results obtained by Support Vector Machines (SVM) were presented.

In this task analyzed data set consists of a matrix of gene expression vectors obtained from DNA micro-arrays for a number of patients with two different types of leukemia (ALL and AML) (Golub et al., 1999). Training set consists of 38 samples (27 ALL and 11 AML) from bone marrow specimens. The test set has 34 samples (20 ALL and 14 AML). It prepared under different experimental conditions and including 24 bone marrow and 10 blood sample specimens. Number of features in the task is 7129. Each attribute corresponds to some normalized gene expression extracted from the micro-array pattern.

The informative subset of attributes in (Guyon et al., 2002) got out by method RFE (a version of algorithm Deletion). In selected subspace of two best attributes 30 test objects were correctly recognized, in subspace of 4 attributes number of correctly recognized objects was 31, in subspace of 128 attributes it was 33. Our results for this task were the following (Zagoruiko et al., 2008b). From 7129 initial attributes algorithm FRiS-GRAD selected 39 most informative attributes and constructed 30 decision rules: 27 of them recognized all 34 test samples correctly. Ten most informative according their FRiS-compactness \( G' \) attribute subsystems are presented in Table 1. Indexes attached to attribute numbers show weights of these attributes in decision rules.

These weights are defined by algorithm GRAD. At calculation of distances between objects the values of the attribute with weight \( v \), are multiplied by \( v \).

Offered algorithm is linear. Its laboriousness has the order of complexity \( O([N + m_1^2/6]M^3) \). There \( M \) is number of objects in the mixed dataset, \( N \) is dimension of features space, \( m_1 \) — number of attributes used for forming two- and three- dimension granules. Here we used \( m_1 = m_2 = m_3 = 100 \).

Difference in presented results and results of SVM can be explained by peculiarities of as attributes selection method, as algorithm of decision rule construction. To compare
Attribute Selection Based on FRiS-Compactness

<table>
<thead>
<tr>
<th>Informative attributes</th>
<th>$G$</th>
<th>$P$</th>
</tr>
</thead>
<tbody>
<tr>
<td>356, 2266, 2358, 2641, 4049, 6280</td>
<td>0.73835</td>
<td>34</td>
</tr>
<tr>
<td>356, 2266, 2358, 2641, 2724, 4049</td>
<td>0.73405</td>
<td>34</td>
</tr>
<tr>
<td>356, 2266, 2641, 3772, 4049, 4261</td>
<td>0.73302</td>
<td>34</td>
</tr>
<tr>
<td>1383, 1833, 2641, 4049, 5441, 6800</td>
<td>0.73263</td>
<td>34</td>
</tr>
<tr>
<td>356, 435, 2641, 4049</td>
<td>0.73214</td>
<td>34</td>
</tr>
<tr>
<td>356, 435, 2641, 2724, 4049</td>
<td>0.73204</td>
<td>34</td>
</tr>
<tr>
<td>1833, 2641, 4049, 4367, 4873, 6800</td>
<td>0.73088</td>
<td>34</td>
</tr>
<tr>
<td>356, 435, 2641, 3560, 4049, 6800</td>
<td>0.72919</td>
<td>34</td>
</tr>
<tr>
<td>356, 2641, 2895, 3506, 4049, 5059</td>
<td>0.72814</td>
<td>34</td>
</tr>
<tr>
<td>356, 2266, 2641, 4049, 4229, 6280</td>
<td>0.72699</td>
<td>34</td>
</tr>
</tbody>
</table>

Table 1: Attributes, used in decision rules in leukemia task

their effectiveness SVM and FRiS-Stolp were run on subspace of two selected by method RFE attributes (genes 803 and 4846). Decision rule constructed by FRiS-Stolp correctly recognized 33 test objects, SVM — 30 objects. In subspace of one gene (4846) results of FRiS-Stolp and SVM were 30 and 27 correctly recognized objects correspondingly.

For comparison decision rule constructed in best two-dimension attribute subsystem (2641 and 4049) selected by FRiS-GRAD gave 33 of 34 correct predictions, in one-dimension subsystem (2461) — 32 of 34 correct predictions. This example demonstrates high competitiveness of attribute selection and decision rule construction based on FRiS-function.

7. Conclusion

Using of similarity measure which considers competitive situation, allows building effective algorithms for main Data Mining tasks solving. Function of rival similarity gives a way to estimate quantitative values of compactness of patterns, informativeness of attribute subspaces and to build easily interpreted decision rules. Presented method can be applied to tasks with any number of patterns, any character of their distributions and any ratio between number of objects and number of attributes in dataset. Laboriousness of a method allows to use it for the decision enough complex real tasks. Applied tasks decisions quality can compete with other methods.

Our future researches of FRiS deal with its application on other types of tasks of the Data Mining — filling of blanks, search of associations, censoring a training dataset and analysis of other types of function of rival similarity.

Acknowledgments


URL: http://www.genome.wi.mit.edu/MPR/data_set_ALL_AML.html.


Effective Wrapper-Filter hybridization through GRASP Schemata

M.A. Esseghir

Artois University, Faculty of Applied Sciences of Bethune, TechnoPark Futura, 62400, France.

Editor: Huan Liu, Hiroshi Motoda, Rudy Setiono, and Zheng Zhao

Abstract

Of all of the challenges which face the selection of relevant features for predictive data mining or pattern recognition modeling, the adaptation of computational intelligence techniques to feature selection problem requirements is one of the primary impediments. A new improved metaheuristic based on Greedy Randomized Adaptive Search Procedure (GRASP) is proposed for the problem of Feature Selection. Our devised optimization approach provides an effective scheme for wrapper-filter hybridization through the adaptation of GRASP components. The paper investigates the GRASP component design as well as its adaptation to the feature selection problem. Carried out experiments showed Empirical effectiveness of the devised approach.

Keywords: Feature selection, Combinatorial optimization, Hybrid modeling, GRASP, Local Search.

1. Introduction

Researchers in machine learning, data mining, pattern recognition and statistics have developed a number of methods for dimensionality reduction based on usefulness and classification accuracy estimates of both individual features and subsets. In fact, Feature Selection (FS) tries to select the most relevant attributes from raw data, and hence guide the construction of the final classification model or decision support system. From one hand, the majority, of learning scheme, are being relying on feature selection either as independent pre-processing technique or as an embedded stage within the leaning process (Guyon and Elisseeff, 2003). On the other hand, both feature selection and data mining techniques struggle to gain the attended reliability, especially, when they face high dimensional data (Liu and Motoda, 2008).

In this paper, we propose, a new hybrid search technique through the adaptation of the GRASP heuristic to the FS problem. The devised approach investigates the effective wrapper-filter combination by exploiting the intrinsic properties of the GRASP heuristic. The main motivations for this proposal are three folds: (i) filter-wrapper collaboration might enhance the relevance of the selected feature subsets (ii) local search approaches have shown their effectiveness in FS with both sequential deterministic procedures (i.e. SFFS (Somol et al., 1999), IFFS (Nakariyakul and Casasent, 2009), etc) and stochastic approaches (i.e. Hill Climbing (Kohavi and John, 1997), Simulated Annealing (Guyon et al., 2006) and Tabu search (Yus, 2009)). The GRASP is a multi-start heuristic based on local search (iii)
endowing basic sequential search procedures with both filter guidance and the stochastic ability to alleviate FS challenging problems like local minima and nesting effect (Guyon et al., 2006; Liu and Motoda, 2008).

The remainder of this paper is organized in five sections. Section 2 formalizes the feature selection problem and gives an overview of representative approaches. Section 3 briefly introduces the GRASP heuristic as well as its recent application to feature selection. Section 4 details the devised GRASP for FS. Section 5 compares and assesses GRASP alternatives behaviors empirically. Finally, Section 6 concludes the paper and provide some directions of future research.

2. Feature selection: basics and background

Let $D$ be a data set with $F$ as a set of features such that $|F| = n$, and let $X$ ($X \subseteq F$) be a subset of $F$. Let $J(X)$ the function that assesses the relevance of the features subset $X$. The problem of feature selection states the selection of a subset $Z$ such that:

$$J(Z) = \max_{X \subseteq F} J(X) \tag{1}$$

In other words, the retained feature subset should be compact and representative of the dataset objects or the underlying context. This might be done by both removing redundant noisy or/and irrelevant attributes by keeping the minimal information loss.

For a given dataset of $n$ features, the exploration would require the examination of $2^n$ possible subsets. Consequently, the search through the feasible solutions search space is a $np$-hard combinatorial problem (Liu and Motoda, 2008). An exhaustive exploration of the feature space seems to be impractical, especially, when $n$ became large or even moderate. Numerous reference approaches have been proposed for the identification of salient features with the highest predictive power (Guyon et al., 2006; Liu and Motoda, 2008). The representative approaches could be, broadly, categorized into two classes: filters and wrappers (Guyon and Elisseff, 2003; Guyon et al., 2006).

**Filters**

Considered as the earliest approaches to feature selection, filter methods discard irrelevant features, without any reference to a data mining technique, by applying independent search which is mainly based on the assessment of intrinsic attribute properties and their relationship with the data set class (i.e. Relief, Symmetrical uncertainty, Pearson correlation, etc) (Liu and Motoda, 2008). The main advantage of the filter methods is their reduced computational complexity which is due to the simple independent criterion used for feature evaluation. In most of the cases, filters provide a ranking based on scores reflecting attribute usefulness to the class.

**Wrappers**

When feature selection is based on a wrapper, attributes are simultaneously evaluated using a classification algorithm. The subset exploration requires a heuristic search strategy. Kohavi et al. (Kohavi and John, 1997) were the first to advocate the wrapper as a general framework for feature selection in machine learning. Numerous studies have used the above framework with different combinations of the evaluation and search components. Featured search technique are ranging from greedy sequential attribute selection methods...
Adaptive Feature Selection

(i.e. SFS, SBE, Floating search (Somol et al., 1999)) to randomized and stochastic methods (i.e. GRASP (Yus, 2009), TABU, BEAM, Genetic Algorithm (Guyon et al., 2006)). The wrapper methods often provide better results than filter ones because they consider a classifier within the evaluation process. We should note that feature selection methods based on wrappers are computationally expensive compared to filters, due to the cost of iterative running of the classification algorithm (Guyon and Elisseeff, 2003).

The motivation to hybrid approaches design is the exhibited multidisciplinary problem nature and the need to overcome the pitfalls of one approach by the advantage of the other one. The simplest form of recombination is to use both filters and wrappers. The common scheme of combination entails a couple of steps. The first one applies filter to reduce the search space, while the second step explores with a wrapper the subsets built from the yielded features returned by the first step (Liu and Motoda, 2008).

3. GRASP

This section covers paper materials by introducing GRASP heuristic principle, components and its recent application to feature selection modeling. The Greedy Randomized search procedure (GRASP) is meta-heuristic for combinatorial optimization problems (Resende, 1999; Resende et al., 2002). Usually known as multi-start procedure, GRASP is based on an iterative process which constructs a solution then fine tune it, through a local search. The multi-start property enlarges the search coverage by exploring different regions of the search space without being influenced by the previous solutions found.

GRASP heuristic was successfully applied to numerous combinatorial problem ranging from scheduling (Aiex et al., 2003) and quadratic assignment (Ahuja et al., 2000) to data mining (Ahmadi and Osman, 2005).

3.1 GRASP Components

The recent optimization scheme proposed by GRASP (Resende, 1999) applies an iterative local search scheme based on incremental solution construction and neighborhood exploration. The iterative process consists of two stages: the construction of a feasible solution and local search.

The construction stage builds a solution, incrementally, using a Restricted List of Candidates RCL. The RCL is, generally, formed by the best solution elements (i.e. elements which can improve the current solution). Solutions are built using a random selection from the RCL.

Once the solution is generated, it passes through the second stage. Within the second stage the solution is iteratively, refined by local search until it reaches a local optima. This procedure is mainly based on neighborhood generation and the exchange of the current solution by the best solution among neighbors. The procedure restarts until no improvement could be gained. The pseudo-codes of both GRASP local search (LS) procedures adapted to the problem of FS will be detailed, below, by Algorithm 1 and 2 in the following section.

The multi-start property of GRASP allows the search process to be not trapped in local minima and to explore different regions of the search space, without being constrained or influenced by the best solution found.
Algorithm 1: G.R.A.S.P.
Input:
F: Initial Feature set
C: Target class Attribute
β: Threshold
d: number of attributes to select
n_{max}: attempts number
Output:
S_{best}: Selected Features
1 Begin
2 S ← ∅
3 S_{best} ← S
4 While Stop Criterion not Satisfied do
5 // Construction stage
6 Foreach f_i ∈ F do
7 g_i ← IGV(f_i, C)
8 Sol_list ← ∅
9 repeat
10 S ← ∅
11 repeat
12 min ← \arg\min_i(g_i),
13 max ← \arg\max_i(g_i)
14 RCL_list = \{v_j, g_j ≤ α.g_{max} + (1 − α)g_{min}\}
15 Randomly select v_j ∈ \{v_j ∈ RCL_list, v_j ∉ S\}
16 S ← S ∪ \{v_j\}
17 RCL ← RCL \{v_j\}
18 until |S| == d;
19 S.fitness = Evaluate(S, Cla)
20 Sol_list ← \{S\} ∪ Sol_list
21 until |Sol_list| = n_{max};
22 S ← getBest(Sol_list);
23 // iterative local search
24 S ← LocalSearch(S)
25 If S.fitness > S_{best}.fitness then
26 S_{best} ← S
27 Return (S_{best})
28 End

Algorithm 2: Iterative Local Search
procedure
Input:
F: Initial Feature set
C: Target class Attribute
Cla: a classifier for solution evaluation
S: input Solution
Output:
S': result of local search
1 Begin
2 S_1 ← S, S_{best} ← S_1
3 Stop ← false
4 repeat
5 Sol_list ← NH(S_1, F)
6 ∀X ∈ Sol_list, Evaluate(X, Cla)
7 S_1 ← getBest(Sol_list)
8 If S_1.fitness > S_{best}.fitness then
9 S_{best} ← S_1
10 Else
11 Stop ← true
12 until (Stop = true);
13 S' ← S_{best}
14 Return (S')
15 End
3.2 GRASP for Feature Selection

The application of GRASP to the FS problem was, recently provided by Yusta in (Yus, 2009). The proposed GRASP was compared to effective FS search techniques like to genetic algorithms, tabu search and SFFS.

The GRASP proposed in (Yus, 2009), is illustrated by Algorithm 1. The algorithm is based on two main stages, namely solution construction (Lines 6-21) and local search procedure (see Algorithm 2). The first stage constructs a fixed number of solutions \( n_{\text{max}} \), and the best one will be selected as a candidate for the second stage. Solutions are constructed according to the attributes retained by the RCL list. The RCL is based on the In-Group Variability (IGV function) criterion (see eq. 2).

\[
\text{IGV}(f_j, C) = \sum_i (f^i_j - \mu_{C(i)})^2
\]

(2)

Where \( f^i_j \) and \( \mu_{C(i)} \) denote respectively the \( i \)-th value of the attribute \( f_j \) and the mean \( \mu_{C(i)} \) of \( f_j \) values for the instances belonging to the same class as the instance \( i \). Besides the attribute selection, is controlled by the parameter \( \alpha \) (Lines 11-17). In fact, it controls the degree of randomness of the procedure.

The second stage applies a hill climbing procedure to the solution provided by the first stage. The pseudo-code of the iterative LS is illustrated by algorithm 2. Each iteration generates neighborhood solutions and exchange current solution with best neighbors if it can improve classification accuracy. The neighborhood structure proposed, by Yusta in (Yus, 2009), is based on attribute replacement and is given by the equation 3:

\[
NH(S) = \{X, X = S \cup \{f_i\} \setminus \{f_j\}, \forall f_i \in F \setminus S, \forall f_j \in S\}
\]

(3)

Such a neighborhood structure \( NH(S) \) consider all combinations of attribute exchange. Consequently, LS is sensitive to the number of selected features. The neighborhood exploration becomes prohibitive even for moderate value of \( n \). The computational complexity is in the order of \( O(p \cdot m) \).  

4. Proposed GRASP-FS

In this section, we investigate the proposed a new GRASP schemata for FS. We focus on a set commonly used local search procedures and filters. Next, we try to adapt and deploy them as GRASP components.

Since the GRASP scheme is based on a restricted list of candidates, this list could be represented by features that seems to be relevant or those that might provide incremental usefulness to the selected feature subset. For the GRASP construction stage we opt for selection scheme capable of generating attribute ranking. Hence, the score associated to features will serve as selection criterion for the RCL generation. The second stage of GRASP tries to enhance solutions by an iterative neighborhood exploration. The subsets are assessed according to a classification criterion (i.e. generalization error rate). The quality of solution fine-tuning, mainly, depends on the nature of the involved neighborhood structure of LS.

---

1. \( p \) and \( m \) respectively denote the number of selected and non-selected features \( (p + m = n) \).
We devise a number of LS procedures based on different neighborhood structures inspired from well known sequential search procedures. The following two sections, detail different design alternatives for both RCL and local search GRASP component.

4.1 RCL generation

Comparatively to the GRASP scheme proposed by Yusta in (Yus, 2009), the same construction phase steps (see Algo. 1 Lines 6-21) are adopted, except the procedure which generates the RCL the (see Algo. 1 Line 6-7).

Any filter criterion could be, instead, used to build RCL. In this paper, we opt for three well known and different selection schema: ReliefF (Robnik-Sikonja and Kononenko, 1997), Symmetrical Uncertainty (SU) (Guyon et al., 2006), and FCBF (Yu and Liu, 2003).

Typically, filters return solutions based on the selection of features with the highest scores. Once the initial RCL is generated\(^2\), the variables are randomly selected to build GRASP first stage solutions. Such a selection schema have, at least, tow benefits: (i) reducing the risk of selection of, only, highly correlated relevant features (ii) the combination of features with moderate usefulness, which are not highly relevant to the target, might promote interaction among selected attributes.

4.2 Local Search Procedures

The local search (LS) is applied at the second stage of the GRASP. It aims at the improvement of the solution provided by the GRASP first stage process. An interesting aspect that could motivate the wrapper choice as component of the GRASP second stage, is the successful application of local search methods in FS modeling (i.e. Tabu search, Simulated annealing, Hill climbing) (Guyon et al., 2006).

In this paper, we devise effective LS procedures inspired from successfully search techniques adapted to the FS. The following paragraphs detail the neighborhood structures that will be deployed within the local search procedures. They will be, also, discussed in the context of FS search space exploration.

**Bit-Flip Local search (BF)** explores neighboring solutions by adding or removing one feature at a time. For solutions encoded with binary strings this operator inverts one bit value for each neighbor. In comparison to the sequential search procedures, the generated neighborhood covers both solutions explored in SFS (see eq. 5) and SBE (see eq. 6). The bit-Flip operator (BF) neighborhood is illustrated by the equation 4.

\[
NH_{BF}(S) = \{X | X = NH^+(S) \cup NH^-(S)\}
\]  

\[(4)\]

\[
NH^+(S) = \{X | X = S \cup \{f_i\}, \forall f_i \in F, f_i \notin S\}
\]  

\[(5)\]

\[
NH^-(S) = \{X | X = S \setminus \{f_i\}, \forall f_i \in S\}
\]  

\[(6)\]

The problem of nesting effect encountered with both sequential forward and backward procedures is alleviated by the merge of the neighborhoods explored by both procedures.

---

2. using filter criterion
Attribute-Flip (AF) local search procedure constructs neighborhood using a permutation between a selected and a non-selected features (see eq. 3). This neighborhood structure was used, by Yusta in (Yus, 2009), as a local search procedure. The two operators explore different region of the current solution neighborhood. There is no overlapping regions \( \mathbb{N}H_{BF}(S) \cap \mathbb{N}H_{AF}(S) = \emptyset \) and the second neighborhood structure is much larger than the first which would require more computational time.

Local search based floating search (SFFS1). Since, SFS and SBE approaches could be seen as local search procedures, floating searches (SFFS and SFBS) could be also considered as an improved version of both sequential procedures and their associated local search. In fact, solutions explored by an iteration of SFFS are those generated by the union of \( NH_1 = NH^+(S) \) and the conditional application of the backward search to the best improvement provided by \( NH_1 \). Note that the LS based on SFFS1 neighborhood is not comparable to that using AF local search. AF applies \( NH^+(.) \) and \( NH^-(.) \) to the same initial solution while, with SFFS, \( NH^-(.) \) is applied to the improved solution after the application of \( NH^+(.) \). Besides, there is no risk of cycling, because the Neighborhood procedures are only applied to improved solutions.

Local search based floating search 2 (SFF2) In the case of SFF2, the same floating search scheme as in SFF1 is adopted, however the backward procedure \( NH^-(.) \) is not applied once but the backtrack is applied iteratively repeated until no improvement can be reached. Comparatively to SFF1, SFF2 requires more computational time than the first floating alternative but might lead to more compact subset size.

5. Empirical study

In this section, we empirically assess the behavior of proposed GRASP schema as well as a selection of the devised components. They will be, also, compared to the baseline GRASP (IGV, AF) proposed by Yusta in (Yus, 2009), where reported results have confirmed the superiority of GRASP over Tabu search, Genetic and Memetic algorithms, and SFFS approach.

Five benchmark datasets were used to validate GRASP components: Sonar, Ionosphere, SpamBase, Audiology and Arrhythmia with respectively 60, 34, 57, 69 and 279 attributes. These datasets are provided by the UCI repository (Blake and Merz, 1998). Reported results, correspond to the average values of at least 50 trial runs. Means, Standard deviation and statistical test validation (t-Test with confidence level of 97.5%) are also provided.

Two types of results are proposed: (i) those corresponding to the best solution fitness (generalization error rate) yielded from the GRASP search. K-Nearest Neighbors (KNN) is used as wrapper classifier (\( K = 3 \)) (ii) the validation on independent data set instances of the resulting features subsets using Artificial neural network (ANN) and Naive Bayes(Guyon et al., 2006). The selection of different classification paradigms for both search and validation make the validation less biased and independent of wrapper classifier. Besides, the validation stage is based on 10-folds cross-validation technique.
<table>
<thead>
<tr>
<th>Data</th>
<th>Model</th>
<th>Fitness (%)</th>
<th>Validation Error (%)</th>
<th>CPU Time(s)</th>
<th>Gain % (Yus, 2009)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sonar</td>
<td>RCL LS</td>
<td>(Yus, 2009)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sonar</td>
<td>IGV AF</td>
<td>15.89(1.71)</td>
<td>32.83(3.40)</td>
<td>40.30(3.01)</td>
<td>14183.88(7196)</td>
</tr>
<tr>
<td>Sonar</td>
<td>IGV AF</td>
<td>14.29(1.88)</td>
<td>31.31(3.66)</td>
<td>39.85(2.71)</td>
<td>14915.13(7493)</td>
</tr>
<tr>
<td>Sonar</td>
<td>SU LS</td>
<td>12.79(1.13)</td>
<td>30.54(3.06)</td>
<td>39.53(2.51)</td>
<td>15867.80(524)</td>
</tr>
<tr>
<td>Sonar</td>
<td>FCBF LS</td>
<td>13.46(0.00)</td>
<td>31.32(3.10)</td>
<td>37.27(1.70)</td>
<td>14920(7450)</td>
</tr>
<tr>
<td>Audiology</td>
<td>IGV AF</td>
<td>49.12(1.96)</td>
<td>52.43(3.14)</td>
<td>54.05(0.15)</td>
<td>343915(280683)</td>
</tr>
<tr>
<td>Audiology</td>
<td>IGV AF</td>
<td>46.74(3.92)</td>
<td>51.54(4.47)</td>
<td>54.09(0.22)</td>
<td>337756(248912)</td>
</tr>
<tr>
<td>Audiology</td>
<td>SU LS</td>
<td>33.36(3.2)</td>
<td>40.52(4.55)</td>
<td>54.16(0.25)</td>
<td>350761(267582)</td>
</tr>
<tr>
<td>Audiology</td>
<td>FCBF LS</td>
<td>36.08(4.72)</td>
<td>40.53(6.91)</td>
<td>54.06(0.14)</td>
<td>338867(258913)</td>
</tr>
<tr>
<td>Arrhythmia</td>
<td>IGV AF</td>
<td>39.72(1.57)</td>
<td>41.98(2.04)</td>
<td>43.61(1.65)</td>
<td>183959(132000)</td>
</tr>
<tr>
<td>Arrhythmia</td>
<td>IGV AF</td>
<td>40.17(1.76)</td>
<td>42.73(2.05)</td>
<td>44.22(1.74)</td>
<td>170925(114978)</td>
</tr>
<tr>
<td>Arrhythmia</td>
<td>SU LS</td>
<td>36.15(1.89)</td>
<td>39.73(2.42)</td>
<td>44.56(1.79)</td>
<td>173505(15749)</td>
</tr>
<tr>
<td>Arrhythmia</td>
<td>FCBF LS</td>
<td>33.82(1.26)</td>
<td>39.33(2.11)</td>
<td>43.54(1.76)</td>
<td>176065(17012)</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>IGV AF</td>
<td>5.63(0.91)</td>
<td>16.34(1.98)</td>
<td>17.92(2.13)</td>
<td>22316(12626)</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>IGV AF</td>
<td>5.95(1.00)</td>
<td>15.51(2.26)</td>
<td>16.97(2.04)</td>
<td>21855(12081)</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>SU LS</td>
<td>5.76(0.98)</td>
<td>15.21(2.48)</td>
<td>17.31(2.6)</td>
<td>24031(14531)</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>FCBF LS</td>
<td>3.51(0.32)</td>
<td>16.33(0.92)</td>
<td>15.73(0.89)</td>
<td>21973(11873)</td>
</tr>
<tr>
<td>SpamBase</td>
<td>IGV AF</td>
<td>16.47(1.04)</td>
<td>19.91(1.50)</td>
<td>20.23(1.57)</td>
<td>347062(190196)</td>
</tr>
<tr>
<td>SpamBase</td>
<td>IGV AF</td>
<td>16.43(1.05)</td>
<td>19.92(1.9)</td>
<td>19.58(1.72)</td>
<td>338671(185750)</td>
</tr>
<tr>
<td>SpamBase</td>
<td>SU LS</td>
<td>14.18(1.12)</td>
<td>15.89(1.66)</td>
<td>17.13(2.7)</td>
<td>311037(159631)</td>
</tr>
<tr>
<td>SpamBase</td>
<td>FCBF LS</td>
<td>13.05(0.84)</td>
<td>15.96(2.18)</td>
<td>15.31(1.88)</td>
<td>331498(181414)</td>
</tr>
</tbody>
</table>

Table 1: GRASP with RCL based filters

<table>
<thead>
<tr>
<th>Data</th>
<th>Model</th>
<th>Fitness (%)</th>
<th>Validation Error (%)</th>
<th>CPU Time(s)</th>
<th>Gain % (Yus, 2009)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sonar</td>
<td>RCL LS</td>
<td>(Yus, 2009)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sonar</td>
<td>IGV AF</td>
<td>15.89(1.71)</td>
<td>32.83(3.40)</td>
<td>40.30(3.01)</td>
<td>14183.88(7196)</td>
</tr>
<tr>
<td>Sonar</td>
<td>IGV BF</td>
<td>28.68(1.58)</td>
<td>33.59(3.90)</td>
<td>41.22(4.19)</td>
<td>15215(7922)</td>
</tr>
<tr>
<td>Sonar</td>
<td>IGV SFF1</td>
<td>5.92(2.05)</td>
<td>31.14(3.47)</td>
<td>40.05(3.08)</td>
<td>14481(6652)</td>
</tr>
<tr>
<td>Sonar</td>
<td>IGV SFF2</td>
<td>6.12(1.9)</td>
<td>31.26(3.38)</td>
<td>38.75(3.62)</td>
<td>12208(5244)</td>
</tr>
<tr>
<td>Audiology</td>
<td>IGV AF</td>
<td>49.12(1.96)</td>
<td>52.43(3.14)</td>
<td>54.05(0.15)</td>
<td>343915(280683)</td>
</tr>
<tr>
<td>Audiology</td>
<td>IGV BF</td>
<td>68.78(1.34)</td>
<td>69.59(2.87)</td>
<td>72.64(2.4)</td>
<td>322789(248574)</td>
</tr>
<tr>
<td>Audiology</td>
<td>IGV SFF1</td>
<td>29.41(1.47)</td>
<td>41.78(3.96)</td>
<td>54.09(0.22)</td>
<td>234579(98248)</td>
</tr>
<tr>
<td>Audiology</td>
<td>IGV SFF2</td>
<td>30.99(1.21)</td>
<td>41.12(2.81)</td>
<td>54.08(0.17)</td>
<td>209670(111471)</td>
</tr>
<tr>
<td>Arrhythmia</td>
<td>IGV AF</td>
<td>39.72(1.57)</td>
<td>41.98(2.04)</td>
<td>43.61(1.65)</td>
<td>183959(132000)</td>
</tr>
<tr>
<td>Arrhythmia</td>
<td>IGV BF</td>
<td>49.47(1.02)</td>
<td>44.91(4.7)</td>
<td>46.34(1.99)</td>
<td>160659(99674)</td>
</tr>
<tr>
<td>Arrhythmia</td>
<td>IGV SFF1</td>
<td>25.38(2.57)</td>
<td>38.27(2.9)</td>
<td>43.41(1.81)</td>
<td>156343(73463)</td>
</tr>
<tr>
<td>Arrhythmia</td>
<td>IGV SFF2</td>
<td>24.42(2.67)</td>
<td>36.64(2.99)</td>
<td>42.64(1.71)</td>
<td>152791(89546)</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>IGV AF</td>
<td>5.63(0.91)</td>
<td>16.34(1.98)</td>
<td>17.92(2.13)</td>
<td>22316(12626)</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>IGV BF</td>
<td>12.63(0.67)</td>
<td>15.78(3.01)</td>
<td>17.38(2.03)</td>
<td>21374(11603)</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>IGV SFF1</td>
<td>2.27(0.52)</td>
<td>14.71(1.64)</td>
<td>17.35(1.2)</td>
<td>18561(8464)</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>IGV SFF2</td>
<td>2.48(0.56)</td>
<td>15.42(1.44)</td>
<td>17.66(1.03)</td>
<td>16035(6597)</td>
</tr>
<tr>
<td>SpamBase</td>
<td>IGV AF</td>
<td>16.47(1.04)</td>
<td>19.91(1.50)</td>
<td>20.23(1.57)</td>
<td>347062(190196)</td>
</tr>
<tr>
<td>SpamBase</td>
<td>IGV BF</td>
<td>23.72(1.1)</td>
<td>22.38(3.46)</td>
<td>21.59(2.43)</td>
<td>328341(17611)</td>
</tr>
<tr>
<td>SpamBase</td>
<td>IGV SFF1</td>
<td>6.85(0.73)</td>
<td>12.28(1.27)</td>
<td>14.90(2.66)</td>
<td>532610(281813)</td>
</tr>
<tr>
<td>SpamBase</td>
<td>IGV SFF2</td>
<td>6.87(0.84)</td>
<td>12.05(1.26)</td>
<td>15.11(2.75)</td>
<td>496728(206866)</td>
</tr>
</tbody>
</table>

Table 2: GRASP with different local search procedures
5.1 Construction Phase

In the first stage of the empirical study, we assess the behaviors of the baseline GRASP with the devised GRASP scheme which is based on Filters to both built RCL and construct solutions.

For each experiment we present, for each dataset, on columns, best solution fitness (lowest error rate %), test accuracy on independent dataset, average runtime CPU, cardinality of best solution (#features) and the gain Toward Baseline GRASP fitness. In Addition to the, average , standard deviation values of the different trials, $t$-test was used for the assessment of the statistical validity of the obtained results toward the baseline method. Table 1 provides results for each data set. Globally, according to the gain (last column) obtained with a GRASP scheme generating the RCL with filters, the baseline method is outperformed in most of the cases.

Fortunately, the improvement obtained with fitness values is confirmed with validation stage (independent data, and different classification techniques for validation). In most of the cases the mean values and $t$-tests showed decrease of the generalization errors. The overall improvement, points out the reliability of the approach, particularly the filters enlisted in the selection of suitable features. All filters enhance at least once, both fitness and validation accuracies. Surprisingly, Relief scores used in the RCL build, seems to be the less relevant filter used in the first stage of GRASP whereas GRASP alternatives based on FCBF confirm a slight superiority over those ones using SU.

5.2 Local search enhancement

The local search of the baseline method uses Attribute Flip neighborhood whereas the proposed GRASP uses local search procedures inspired from deterministic sequential searches.

The devised local search procedures are deployed within new GRASP instances using the IGV criterion on the First stage. Table 2 compare and evaluate the fours GRASP instances. Even though, the solutions provided by the first GRASP stage are based on IGV criterion, some of the devised local search procedures have succeed to outperform the baseline algorithm. Indeed, local search alternatives adopting floating selection, have empirically confirmed their superiority over Yusta GRASP. On the other hand, the neighborhood structure based on the selection or removal of one attribute ($NH_{BF}$), the less effective fine tuning scheme.

Besides, the overall improvement of the new devised GRASP local search procedures are most significant that the improvements afforded by the use of filters. In any case, the adapted new GRASP scheme is based have empirically shown that enhancements could be afforded by filters in first stage as well as wrappers in second stage.

6. Conclusion

We devise a new GRASP approach for feature selection capable of hybridizing filters and wrappers. The effectiveness of the different GRASP components combinations were assessed empirically. Carried out results, confirms the robustness of the hybridization schemata and motivates us to investigate in depth both algorithmic and behavioral aspects of further combinations issues, scalability study, and adaptation to high dimensional problems.
References


Feature Extraction for Machine Learning:
Logic–Probabilistic Approach

Vladimir Gorodetsky
GOR@IIAS.SPB.SU
Prof. of Computer Science,
Chief Scientist of Practical Reasoning, Inc. and The Intelligent Systems
Laboratory of The St. Petersburg Institute
for Informatics and Automation of the Russian
Academy of Science.
SPIIRAS, 39, 14-th Line V.O.,
St. Petersburg, 199178, Russia

Vladimir Samoylov
SAMOVL@IIAS.SPB.SU
research fellow of Practical Reasoning, Inc. and The Intelligent Systems
Laboratory of The St. Petersburg Institute
for Informatics and Automation of the Russian
Academy of Science.
SPIIRAS, 39, 14-th Line V.O.,
St. Petersburg, 199178, Russia

Editor: Huan Liu, Hiroshi Motoda, Rudy Setiono, and Zheng Zhao

Abstract

The paper analyzes peculiarities of preprocessing of learning data represented in object
data bases constituted by multiple relational tables with ontology on top of it. Exactly
such learning data structures are peculiar to many novel challenging applications. The
paper proposes a new technology supported by a number of novel algorithms intended
for ontology-centered transformation of heterogeneous possibly poor structured learning
data into homogeneous informative binary feature space based on (1) aggregation of the
ontology notion instances and their attribute domains and subsequent probabilistic cause-
consequence analysis aimed at extraction more informative features. The proposed tech-
nology is fully implemented and validated on several case studies.

Keywords: ontology, object data base, feature aggregation, cause-consequence depen-
dency, non-classical probabilistic space

1. Introduction

The paper proposes automatic feature extraction algorithm in machine learning for classifi-
cation or recognition. Specificity of the problem statement is that it assumes that learning
data (LD) are of large scale and represented in object form, i.e. by multiple tables of rela-
tional database with ontology on top of it. Existing techniques for feature extraction and
machine learning are mostly oriented to LD represented in the form of a flat table. In case
of data stored in object database, a lot of new problems emerge. Indeed, to extract par-
ticular instance (object), it is necessary to use specific query language (Jean et al., 2006).
But what is actually challenging here is that various objects can be of various formats and
structures. Every object instance structure is composed of formidable number of concept instances, and each concept can be specified with a lot of heterogeneous attributes, e.g. categorical, Boolean, real valued, and even with a text thus making feature selection and detection of most informative ones a challenging problem.

On the other hand, object data, in its nature, is much more informative in comparison with LD represented in relational data base or in flat table. The main advantage of object-based LD representation is that object data base instances contain rich context embedded in it via object structure and object attributes. In fact, each object instance is a piece of knowledge compatible with ontology formalizing meta-knowledge. This is a reason why learning of classification using LD in object form is very perspective and productive although complex research direction.

The paper proposes an original technology for preprocessing of ontology-based LD intended for its transformation into a compact binary-valued flat table representing LD object instances in terms of highly informative features. This technology is demonstrated by a case study, electrical machine diagnosis based of vibro-acoustic data measurements. In the rest of the paper, Section 2 describes briefly the aforementioned case study and its ontology specifying meta-knowledge. Section 3 outlines the proposed technology of ontology-based LD aggregation for feature extraction and filtering. It is worth to note here that the main peculiarity of this filtering algorithm is that the resulting sets of features are class-specific. Section 4 outlines the final step of the feature sets and LD transformation to more informative and compressed form via extraction cause-consequence rules. Section 5 concludes the paper and outlines technology perspectives.

2. Cases Study and Domain Ontology

The case study is taken from UCI repository (4, 1990). The task objective is classification of states of electrical pumps using measurements of vibro-acoustic data (VAD) in different measurement points (key points) of pumps in different lines (directions). These data are very multidimensional and have complex structure that is represented by the developed ontology (Fig. 1). Let us describe this ontology while explaining, in parallel, the ontology-based structure of learning data.

For any Electric Pump (EP) Electric pump having own Shaft speed and state Machine state, measurement data Measuring data assigned a time stamp is done. This data is presented in the form of VAD Vibroacoustic measuring. VADs are measured in several key points Measure keypoint along several orthogonal lines (directions). The VADs, in turn, are represented as spectral data Spectral data obtained by several filters. As a rule, no more than three filters are used in every key point along every direction. Spectral data are presented by amplitudes Amplitude mapped to several values of frequency Frequency for current value of time stamp and Preceding amplitude in the same key point and along the same line corresponding to the immediately previous value of time stamp. The total number of combinations of used filters and measurement directions is fixed; it is equal to 9. These combinations are introduced as the values of a specific feature “direction-filter” Direction-Filter. Example of measurement data instances at a time instant is presented in Fig. 2.
In addition to the above described ontology, so-called ontology of secondary features is introduced by the expert. They can be of two categories, auxiliary features and secondary features involved in learning. Auxiliary features are the ones corresponding to the initial real-valued measurements of spectral data transformed into categorical measurement scale. This transformation was made using overlapping spectral data domain quantization with the total number of intervals equal to 20 with overlapping ratio 10%. For the secondary features involved in learning, a new feature type is introduced, Pair-wise of any measurements. Such feature type contains two positions mapped correspondingly two connected concepts. In general case, components of any pair-wise measurement can be categorical, ordered, or real valued. The secondary feature ontology is given in Fig. 3. In the case study, the following features of the standard or Pair-wise of any measurements involved in learning are used:

Secondary features of standard type:
- Nominal amplitude; | Nominal difference of amplitudes.

Secondary features of pair-wise type:
- Frequency–Nominal amplitude     Key point–Frequency
- Frequency–Nominal difference of amplitudes     Shaft speed–Frequency
- Nominal amplitudes–Nominal difference of amplitudes     Key point–“Direction-filter”
- “Direction-filter” – Nominal difference of amplitudes     “Direction-filter” – Nominal amplitude

Let us note that in the case study the components of all secondary features are categorical.

It can be seen that structure of LD, in the case study in question, is rather complex and multidimensional. Due to introduced preliminary expert-based transformation of spectral data it is reduced to a structure of categorical data. Later on, it is used for demonstration of the developed feature extraction procedure.
Figure 2: Instance of learning data represented in ontology-based form (in object data base).

3. Technology for Ontology-Based Feature Extraction

3.1 Ontology-Based Learning Data Transformation and Feature Aggregation

The proposed technology is designed for learning of classification with LD stored in object data base. In the case study, such data are structured according to the domain ontology. In general case, LD can also include poorly-structured data in the form of texts on a natural language.

The technology itself illustrated by Fig. 4 is composed of several phases while assuming that ontology can be either given as input information or developed by expert (the last
The **first phase** is expert-based selection of preliminary feature space and transformation of the initial structure of LD to this space. The expert is permitted to select any number of potentially relevant features without any care about types of them or dimensionality, up to thousands. The mandatory requirement here is that the selected features have to be concepts or attributes of the ontology. This is important because such features are semantically interpretable and their structure determined by the ontology constitutes particular context of any LD instance. When such preliminary set of features is selected, any instance of LD can be extracted using an object data base query language (Jean et al., 2006). Through such queries all LD instances are transformed into the space of the preliminary selected (potentially redundant) feature space. According to the technology, the resulting LD are represented as “star”-structured set of tables, in which columns of fact tables corresponds to elements of the designated preliminary feature set with one row in kernel table per every LD instance assigned a class label. This representation is context-dependent where different LD instances can be of different formats since some features introduced by expert can be irrelevant to particular instances. Therefore any table of star structure can contain “missing-like” values to be interpreted as “irrelevant” to the corresponding object instance.

The **second phase** is aggregation and filtering of the features selected at the first phase, as well as representation of filtered set of aggregated features in unary predicate form. The final procedure of the second phase is transformation of LD obtained at phase 1 to new **class-centered feature space**. Let us briefly explain the mathematical idea of feature
aggregation procedure while using following denotations: $\Omega = \{\omega_1, \ldots, \omega_m\}$ stands for the set of classes labels that can be assigned to an LD instance, e.g. $\Omega = \{1, \ldots, 6\}$, in the case study; $\mathbf{X} = \{X_1, \ldots, X_n\}$ — the set of feature identifiers (ID), where $X_i$ stands for particular feature ID; $x^i_s$ — particular value of the feature with ID $X_i$, and $\mathbb{N}_i$ — domain of the feature $X_i$, i.e. $x^i_s \in \mathbb{N}_i$. Let us note that cardinality of any feature $X_i$ domain may be huge (if either categorical, or numeric, or real valued). For example, categorical feature “Key male role in a movie” in the NetFliX task (5) can possesses thousands of values corresponding to particular actors’ names. Let also symbol $\sum$ stands for the set of LD instances in the target filtered feature space.

Feature aggregation and filtering is realized by single procedure. For a value $x^i_s$ of feature $X_i$, $x^i_s \in \mathbb{N}_i$ and a class $\omega_k$, an aggregate $\mathbb{N}_i(\omega_k) \in \mathbb{N}_i$ is defined as follows:

$$x^i_s \in \mathbb{N}_i(\omega_k) \text{ if and only if for } \forall \omega_s \in \Omega, \nu \neq k : p(\omega_k/x^i_s) > p(\omega_s/x^i_s) + \Delta, \quad (1)$$

where $\Delta$ is a positive real value defining a dominance threshold. The inequality (1) states that conditional probability of the class $\omega_k$, $p(\omega_k/x^i_s)$, if the feature $X_i$ is instantiated by the value $x^i_s$ is larger than the same conditional probability for any other class. Thus, to compute an aggregate $\mathbb{N}_i(\omega_k)$, it is necessary to check the inequality (1) for $\forall x^i_s \in \mathbb{N}_i$ and $\forall \omega_s \in \Omega$ for all $\nu \neq k$. Each such aggregate can be computed using sample $\sum$.

Finally, at the second phase, let us introduce unary predicates $B_i(\omega_k)$ that are instantiated by the truth value “true” if and only if $x^i_s \in \mathbb{N}_i(\omega_k)$, and “false”, otherwise. The truth domains of these predicates are determined uniquely by aggregates with the same subscripts and argument $\omega_k$ values. Thus, the results of the second phase are the aggregates $\mathbb{N}_i(\omega_k)$ and corresponding unary predicates $B_i(\omega_k)$, $i \in I(\omega_k)$, where $I(\omega_k)$ is the subset of indexes of features $X_i$ successfully passed the test (1) for fixed $\omega_k$. 

Figure 4: Ontology-based classification system technology.
Using inequality (1) and definition of the predicates $B_i(\omega_k)$, the LD sample $\sum$ is transformed to the set of samples $\sum(\omega_1), \ldots, \sum(\omega_m)$, representing LD in the space of binary features that are predicates $B_i(\omega_k)$.

The authors’ experience based on prototyping of several applications where the developed technology was used showed that, as a rule, the procedure (1) filters many features of the set $\{X_1, \ldots, X_n\}$ that are not satisfied with (1) for any $\omega_k \in \Omega$. Let us also note that the value $\Delta$ of the dominance threshold can be used as a means to restrict the total number of the finally extracted features (either aggregates $\mathcal{N}_i(\omega_k)$, or unary predicates $B_i(\omega_k)$) to a predefined limit.

Thus, in result of the phases 1 and 2 the source high-dimensional heterogeneous LD of a complex structure is transformed to a homogeneous binary feature space of desirable dimension.

### 3.2 Cause-Consequence Rule Extraction

Phase 3 starts when aggregates $\mathcal{N}_i(\omega_k)$, unary predicates $B_i(\omega_k)$, $i \in I(\omega_k)$, $\omega_k \in \Omega$ and LD samples $\sum(\omega_1), \ldots, \sum(\omega_m)$ are formed. In Fig. 4 this phase is denoted as phase 3. Its objective is to find cause-consequence dependencies (rules) between conjunction of predicates $B_i(\omega_j)$, $i \in I(\omega_j)$. $\Omega = \{\omega_1, \ldots, \omega_m\}$ and $\omega_j \in \Omega$. For this purpose, a probabilistic approach is used. Let us describe it for particular $\omega_k \in \Omega$.

For $\omega_k \in \Omega$ probabilistic space is introduced as follows. The set of aggregates $\mathcal{N}_i(\omega_j)$ is considered as a family set $\{\mathcal{N}_i(\omega_j)\}_{i \in I(\omega_k), j = 1, \ldots, m}$, where each set $\mathcal{N}_i(\omega_j)$ is mapped a probabilistic measure

$$p(\mathcal{N}_i(\omega_j)) = |\mathcal{N}_i(\omega_j)|/|\mathcal{N}_i|,$$

where $| |$ denotes cardinality of the corresponding set. It is clear that

$$p(B_i(\omega_j)) = p(\mathcal{N}_i(\omega_j)) \quad (2')$$

Since the aggregates $\mathcal{N}_i(\omega_j)$ can overlap with $\mathcal{N}_r(\omega_j)$, $j \neq r$ these aggregates and corresponding random events can be dependent. Each set $\mathcal{N}_i(\omega_j)$ of the family set $\{\mathcal{N}_i(\omega_j)\}_{i \in I(\omega_k), j = 1, \ldots, m}$, can also be correlated with any $\omega_k \in \Omega$, which are also considered in the model as random events with predefined a priori probabilities. Therefore the sets of family $\{\{\omega_k\}_{k = 1}^m, \{\mathcal{N}_i(\omega_j)\}_{i \in I(\omega_k), j = 1, \ldots, m}\}$, cannot be used as elementary random events and thus probabilistic space cannot be defined here in the classical manner. In this work, “non-classical” definition of the probabilistic space and corresponding non-classical probability space axiomatics (Halpern, 2003).

While omitting some algebraic details, this probabilistic space projected to the subspace taking into account only $\omega_k$ can be modeled as an upper $\gamma_k^\vee = \langle \{\omega_k, \{\mathcal{N}_i(\omega_j)\}_{i \in I(\omega_k), j = 1, \ldots, m}\}, \gg \rangle$ or lower $\gamma_k^\wedge = \langle \{\omega_k, \{\mathcal{N}_i(\omega_j)\}_{i \in I(\omega_k), j = 1, \ldots, m}\}, \leq \rangle$ semi-lattice, where order relation is defined in usual theoretic-set sense. In this semi-lattices, any node is mapped a probability of the corresponding random event. Further on, the lower semi-lattice is used. In this semi-lattice, $\omega_k$ is the class label node called below “target node”. The model described below is identical for any $\omega_k \in \Omega$.

**Definition.** Hasse diagram of the lower semi-lattice $\gamma_k^\wedge = \langle \{\omega_k, \{\mathcal{N}_i(\omega_j)\}_{i \in I(\omega_k), j = 1, \ldots, m}\}, \leq \rangle$ is below called Associative Bayesian Network (ABN).
Let us note that this notion was introduced in the paper (Gorodetsky, 1992). Fig. 5 gives an example of a fragment of ABN built for the case study described in Section 2. Semantics of the aggregates is also described in that section.

Let us consider the set \( \{ \omega_k, \{ B_i(\omega_j) \}_{i \in I(\omega_k), j=1,...,m} \} \) as the basic set (nodes) of the lower semi-lattice \( \gamma_k^l \) that is isomorphic with the semi-lattice \( \gamma_k^l = < \{ \omega_k, \{ N_i(\omega_j) \}_{i \in I(\omega_k), j=1,...,m} \}, \leq > \). General idea of the developed algorithm of cause-consequence (CC) — rule extraction consists in iterative construction of ABN which nodes represent premises of the cause-consequence rules (CC-rules) in the form \( <\text{conjunction of a subset of the basic predicates of ABN with negation or without it}> \Rightarrow \omega_k \) only. This algorithm is iterative and the number of particular iterations coincides with the length of conjunctive premises generated at corresponding iteration. Below very short and slightly simplified outline of CC-rule extraction algorithm is done (due to limit of the paper space). Below the denotation \( \vec{B}_i(\omega_j) \) is used predicate identifier (literal) that can take two values: \( B_i(\omega_j) \) if it is considered without negation and \( \overline{B}_i(\omega_j) \) with negation.

1. **Generation of the rule set containing 1-literal premises.** Let \( \{ \omega_k, \vec{B}_i(\omega_j) \}_{i \in I(\omega_k), j=1,...,m} \) be all the pairs composed of a literal \( \vec{B}_i(\omega_j) \) and the target node \( \omega_k \). The first to be done is to assess joint probabilities \( p(\vec{B}_i(\omega_j) \omega_k) \) for every assignment of the literal \( \vec{B}_i(\omega_j) \). These filters applied to \( p(\vec{B}_i(\omega_j) \omega_k) \) described below are sequentially used to filter the above pairs that can be the sources of rules in the form \( \text{"If } \vec{B}_i(\omega_j) \text{ then } \omega_k \text{" assigned confidence measure } p(\omega_k/\vec{B}_i(\omega_j)) \) where \( \vec{B}_i(\omega_j) \in \{ B_j(\omega_j), \overline{B}_i(\omega_j) \} \) (positive and negative literals respectively).

Filter 1 (filters the rules containing independent premises and consequents)

\[
I(B_i(\omega_j), \omega_k) = \frac{|p(B_i(\omega_j) \omega_k) - p(B_i(\omega_j))p(\omega_k)|}{|p(B_i(\omega_j))p(\omega_k)|} \geq \delta_{\text{min}} > 0 - \alpha \tag{3}
\]
selection threshold. Otherwise, the corresponding 1-literal rule is non-interesting.

Filter 2 (filters the rules that are dependent but do not correspond to the CC-dependencies)

\[
R(\hat{B}_i(\omega_j), \omega_k) = |p(\omega_k/\hat{B}_i(\omega_j)) - p(\omega_k/B_i(\omega_j))|/|p(\hat{B}_i(\omega_j))[1 - p(\hat{B}_i(\omega_j))]| =
\]

\[
= |p(\hat{B}_i(\omega_j)\omega_k) - p((\hat{B}_i(\omega_j))\omega_k)|/|p(\hat{B}_i(\omega_j))[1 - p(\hat{B}_i(\omega_j))]| \geq \delta_{\min}, \delta_{\min} > 0 - a \quad (4)
\]

Filter 3 (filters CC-rules with low confidence)

\[
p(\omega_k/\hat{B}_i(\omega_j)) = p(\hat{B}_i(\omega_j)\omega_k)/p(\hat{B}_i(\omega_j)) \geq \gamma_{\min}
\]

at least, for one of assignments of the random events \(\hat{B}_i(\omega_j)\) and \(\omega_k\), \(\gamma_{\min} > 0 - a\) selection threshold value. Otherwise, the corresponding 1-literal rule is non-interesting.

Notice: In fact, this filter is more complex. The filtration has to be done not only for any possible assignments of random event \(\hat{B}_i(\omega_j)\) \(\in \{B_i(\omega_j), \overline{B_i(\omega_j)}\}\), but also for two assignments of random event \(\omega_k \in \{\omega_k, \overline{\omega_k}\}\) in order not to lose the rules in the form \(\hat{B}_i(\omega_j) \Rightarrow \overline{\omega_k}\). If, at least, for one of variant of assignment of above mentioned random events the filtration is successful then corresponding 1-literal rule remains to be a candidate, otherwise it is deleted from the candidate set. Here and at the subsequent steps of CC-rule extraction such additional checks are assumed on default and are not described due to limitation of the paper space.

Let us note that measure \(R(\hat{B}_i(\omega_j), \omega_k)\) is well known in probability theory and mathematical statistics as regression coefficient of the random events \(\hat{B}_i(\omega_j)\) and \(\omega_k\).

First difference is that, at this step, all the conjunctive pairs \(\hat{B}_i(\omega_j) \land \hat{B}_j(\omega_r), \hat{B}_i(\omega_j),\hat{B}_j(\omega_r)\in C_1\) are considered as the 2-literals CC-rule premises candidates. They are subjected to the analogous three step filtration used for 1-literal rules, and then, like \(C_1\), the set \(C_2\) of 2-literals premises containing the chosen conjunctive pairs \(\hat{B}_i(\omega_j) \land \hat{B}_j(\omega_r)\in C_2\), \(i, j \in I_2(\omega_k)\) is formed (\(\omega_k\) is target node).

Second difference is that additionally, at this (and, in analogy, at the subsequent steps too), the set of non-chosen predicate literals \(\overline{B}_i(\omega_j)\) united in the set \(A_1(\omega_k)\) that is the set of 1-literal premises of the rules \(R_1(\omega_k)\) in the form \(\overline{B}_i(\omega_j) \Rightarrow \omega_k, i \in I_1(\omega_k)\), containing \(\omega_k\) in the consequences.

The process stops when either the set \(C_k\) of \(k\)-literals candidates became empty, or a predefined number of rules is found. The latter often is a good choice in order to prevent an over-fitting. Control attribute \(\Delta\) in the equation (1) plays the same role. The resulting set \(A(\omega_k) = \bigcup_{r=1}^N A_r(\omega_k)\), is the target set of features forming new feature space.
4. Some Experimental Results

An extended experiment was performed for the case study described in Section 2. Let us first note that in UCI repository (4, 1990) only results obtained by the benchmark authors are given. In fact, this task is too complex for existing approaches due to very complex data structure. Unfortunately the benchmark contains very limited number of instances (objects instances). They were divided into training and testing sets and the latter were not involved in learning procedure. The results of testing of the produced classifier on training data are presented in Tab. 1, whereas the results of its testing on the data that was not used in training are done in Tab. 2. Let us comment shortly these results. It is important to note that training data set has much less training instances as compared with testing one. One of our ideas of such decision was to check performance of the developed feature selection technology on relatively small training sample. It can be seen from the Tab. 1 that classification quality with regard to training sample is rather good. What concerns testing sample, it is important to note that the resulting algorithm has practically no misclassification, but in a large number of cases it refused to decide in favor of particular class. But classification algorithm was not carefully designed due to the fact that the paper objective is other than design good classification algorithm. More important, for this paper, is that the features designed according to the proposed technology found out informative and even for not carefully designed classifier provides the decision quality that is not worth in comparison of the results provided in UCI repository.

| Table 1. Contingency matrix for testing of classifier on training data |
|---|---|---|---|---|---|---|
| 1 | 15 | 16 | 7 | 10 | 15 | 1 |
| 2 | 1 | 46 |
| 3 | 10 |
| 4 | 1 |
| 5 | 7 |
| 6 | 13 |

| Table 2. Contingency matrix for testing of classifier on new data |
|---|---|---|---|---|---|---|
| 1 | 17 | 16 | 7 | 10 | 15 | 1 |
| 2 | 46 |
| 3 | 53 |
| 4 | 7 |
| 5 | 7 |
| 6 | 13 |

5. Conclusion

The authors’ practical experience proved that the proposed feature space synthesis approach works well in very “heavy” high dimensional learning tasks using heterogeneous relational data with ontology on top of it. One of the important advantages of the developed approach is that the resulting feature space is homogeneous (binary) and most of the existing classification mechanisms can be used at decision making step. The proposed feature extraction approach was fully implemented and validated using several applications. It was also used in design and implementation of an ontology-based profiling and recommending system. In particular, intelligent e-mail assistant for incoming e-mail sorting was prototyped.
References

Netflix. http://www.netflix.com. Different pruning measures can be used but this aspect is out of the paper scope.


Feature Extraction for Outlier Detection in High-Dimensional Spaces

Hoang Vu Nguyen
ng0001vu@ntu.edu.sg

Vivekanand Gopalkrishnan
asvivek@ntu.edu.sg

School of Computer Engineering
Nanyang Technological University
50 Nanyang Avenue, Singapore

Editor: Huan Liu, Hiroshi Motoda, Rudy Setiono, and Zheng Zhao

Abstract
This work addresses the problem of feature extraction for boosting the performance of outlier detectors in high-dimensional spaces. Recent years have observed the prominence of multidimensional data on which traditional detection techniques usually fail to work as expected due to the curse of dimensionality. This paper introduces an efficient feature extraction method which brings nontrivial improvements in detection accuracy when applied on two popular detection techniques. Experiments carried out on real datasets demonstrate the feasibility of feature extraction in outlier detection.

Keywords: Feature Extraction, Dimensionality Reduction, Outlier Detection

1. Introduction
Outlier detection is an important data mining task and has been widely studied in recent years (Knorr and Ng, 1998). As opposed to data clustering, where patterns representing the majority are studied, anomaly or outlier detection aims at uncovering abnormal, rare, yet interesting knowledge which may stand for important events. Popular techniques for outlier detection, especially distance-based ones (Knorr and Ng, 1998), usually compute distances of every data sample to its neighborhood to determine whether it is an outlier or not. However, as these approaches compute distances in the full feature space, they suffer the curse of dimensionality (Aggarwal and Yu, 2005).

Reducing data dimensions for better learning process, especially in sparsely filled high-dimensional spaces has been studied for a long time. Various solutions, ranging from principle component analysis (PCA) (Kirby and Sirovich, 1990), linear discriminant analysis (LDA) (Swets and Weng, 1996), null space LDA (NLDA) (Liu et al., 2004), etc., have been successfully proposed to address this issue for the classification task. Outlier detection itself can be regarded as a binary asymmetric/unbalanced pattern classification problem, where one class has much higher cardinality than the other, provided that some training data are available (Lazarevic and Kumar, 2005). Recently, Chawla et al. (2003) pointed out that the high imbalance in class cardinalities of asymmetric classification causes normal classification techniques to yield unsatisfactory accuracy (e.g., too complex learning rules which cause overfitting). This necessitates the development of new techniques to specifically deal with...
the issue. Analogously, existing feature extraction techniques for normal classification also do not work well when applied to our problem of interest. Particularly, Chen et al. (2008) highlights that traditional techniques typically seek accurate performance over a full range of instances, and hence, tend to classify all data into the majority class. This causes the minority class, which is usually more important, to be missed out. More suitable methods, like those in (Lee and Stolfo, 2000; Wu and Banzhaf, 2010), have been proposed to address the problem. In other words, it is impractical to apply standard discriminant feature extraction approaches for outlier detection.

In general, being able to reduce the number of data dimensions helps to overcome the lack of data and avoid the over-fitting issue. Recognizing this need, we present Dimensionality Reduction/Feature Extraction for OUTlier Detection (DROUT), an efficient method for feature extraction in outlier detection. In brief, DROUT first applies eigenspace regularization on a training set randomly sampled from the considered dataset. It then extracts a relevant set of features, and finally transforms the testing set where detection algorithms are applied using the features obtained. By performing eigenspace regularization, we are able to mitigate the loss of discriminant information during the feature extraction process. Furthermore, different from other techniques on feature extraction, in DROUT, eigenvalue regularization and feature extraction are performed on weight-adjusted scatter matrices (explained in Section 3) instead of normal ones. Those matrices specifically target at outlier detection where class cardinalities (normal class v/s. outlier class) are highly unbalanced. This helps DROUT to work better than existing techniques in mining anomalies.

The rest of this paper is organized as follows. Related works are presented in the next section. We present the DROUT approach in Section 3. In Section 4, we apply DROUT on two existing outlier detection techniques and empirically evaluate its performance on real datasets. Finally, we conclude the paper in Section 5 with directions for future work.

2. Literature Review

Linear subspace analysis for feature extraction and dimensionality reduction has been studied in depth for a long time and many methods have been proposed in the literature, including principle component analysis (PCA) (Kirby and Sirovich, 1990), linear discriminant analysis (LDA) (Swets and Weng, 1996), null space LDA (NLDA) (Liu et al., 2004) etc. Though applied very successfully for pattern classification, these methods usually miss out some discriminant information while extracting relevant features for the classification task.

In particular, the eigenspace spanned by eigenvectors of the data within-class scatter matrix can be divided into three subspaces: the principal, the noise, and the null subspaces (Jiang et al., 2008). In words, the principal subspace, corresponding to eigenvectors of large eigenvalues contains the most reliable discriminant information extracted from the training data. The noise subspace, on the other hand is spanned by eigenvectors with nonzero small eigenvalues. These eigenvalues are unreliable and cause over-fitting to each specific training set. Finally, the null subspace consists of eigenvectors of zero eigenvalues. The impact of the null subspace is similar to that of the noise one. Feature extraction methods typically: (a) solve the eigenvalue problem to obtain a set of eigenvectors and corresponding eigenvalues, and (b) discard the unreliable dimensions with small eigenvalues and keep the rest
for performing the classification task. The noise and null subspaces are caused by noise
and mainly by the insufficient training data. As new data are added to the training set
or as the training set is replaced by a different one, small or zero eigenvalues can be easily
changed, i.e., zero eigenvalues become nonzero and small ones become larger. Therefore,
simply getting rid of them in the early stage may cause loss of discriminant information.
This is especially true in the case of outlier detection because: (a) outliers are rare and
hard to collect, (b) selecting subspaces for outlier detection is a complex problem (Aggar-
wal and Yu, 2005). The second factor implies that outliers may be present in only some
lower-dimensional projections of data. Thus, accidentally rejecting some dimensions just for
the sake of easy computation may lead to some loss of knowledge. Motivated by the issue,
Jiang et al. (2008) proposes ERE, a dimensionality reduction method that first regularizes
all three subspaces of the data within-class scatter matrix, and then extracts discriminant
features on the transformed total scatter matrix. The merit of ERE lies in the fact that no
dimensionality reduction is done during the regularization phase, i.e. discriminant informa-
tion is likely preserved.

Current solutions for feature extraction (including ERE) are unsuitable for binary asym-
metric classification, because they rely on the usual within-class scatter matrix that after
being processed by solving the eigenvalue problem, usually leads to rejection of reliable
features. To overcome this issue, (Jiang, 2009) suggest to adjust the weights of class condi-
tional covariance matrices. However, their proposed approach (APCDA) extracts discrim-
inant features after applying PCA on the adjusted total scatter matrix. According to our
aforementioned discussion, this will cause loss of discriminant features.

3. The DROUT Approach

Our approach aims to overcome the weaknesses of the ERE and APCDA approaches. Simi-
lar to APCDA, our DROUT approach performs eigenspace decomposition as well as feature
extraction on the weight-adjusted scatter matrices. But in order to preserve the discrim-
inant information till the feature extraction phase, DROUT applies the strategy of ERE,
and does not discard any feature during the eigenspace regularization process. Thus, our
approach can take advantage of both ERE and APCDA to overcome the curse of dimen-
sionality in outlier detection.

In DROUT, the selected set of features is not a subset of the initial set of attributes, but
is extracted from a transformation of the original data space (vector space). In order for
DROUT to work, we make the following assumptions. First, a training set containing both
normal data and a small amount of outliers is available. We further assume that training
and testing sets have similar structures, allowing features extracted from the training set
to be applicable on the testing set. While the latter assumption is widely used in almost
all works on dimensionality reduction (Liu et al., 2004), the former appears frequently in
works on anomaly detection (Lazarevic and Kumar, 2005).

The training set in our assumption consists of two classes: the normal class $\omega_m$ and the
anomaly class $\omega_a$. The class $\omega_m$ contains $N_m$ points with class-conditional mean vector $\mu_m$
and covariance matrix $\Sigma_m$. Analogously, $N_a$, $\mu_a$, and $\Sigma_a$ are the support, class-conditional
mean vector, and covariance matrix of $\omega_a$. Each data point $p$ is expressed as a column vector
of $d$ dimensions, i.e., $p \in \mathbb{R}^d$. Let $N_t = N_m + N_a$ be the training set’s total cardinality and
Feature Extraction for Outlier Detection

Algorithm 1: ExtractFeatures

**Input:** $DS_t$: the training set, $\xi_m$ and $\xi_a$: the adjusted weights, $b$: the number of extracted features

**Output:** $A_T$: the transformation matrix

1. Compute $\Sigma_m$, $\Sigma_a$, and $\Sigma_b$ from $DS_t$.
2. Set $\Sigma_w = \xi_m \Sigma_m + \xi_a \Sigma_a$.
3. Compute $\tilde{\Phi}_w^{\xi}$ based on $\Sigma_w$ and (2).
4. Compute $\Sigma_t$ in (3) by solving the eigenvalue problem on $\tilde{\Sigma}_t$.
5. Compute $\Phi_a^{\xi}$ and $\Phi_b^{\xi}$ in (4).
6. Set $A_T = \Phi_w^{\xi} \Phi_a^{\xi} \Phi_b^{\xi}$.

$\mu_t$ be the mean vector of all training data. The within-class, between-class, and total scatter matrices of the training set are defined as: (a) $\Sigma_w = \frac{N_m}{N_t} \Sigma_m + \frac{N_a}{N_t} \Sigma_a$, (b) $\Sigma_b = \frac{N_m}{N_t} (\mu_m - \mu_t)(\mu_m - \mu_t)^T + \frac{N_a}{N_t} (\mu_a - \mu_t)(\mu_a - \mu_t)^T$, (c) $\Sigma_t = \Sigma_w + \Sigma_b = \frac{N_m}{N_t} \Sigma_m + \frac{N_a}{N_t} \Sigma_a + \Sigma_b$, respectively.

In order to overcome the limitations of existing techniques as well as to better align DROUT towards outlier detection, we propose to apply eigenspace decomposition and regularization as in ERE on weight-adjusted scatter matrices instead of the usual ones. The details are summarized in Algorithm 1 and explained in the remaining of this section.

### 3.1 Weight-Adjusted Within-Class Scatter Matrix

While computing the within-class scatter matrix, $\Sigma_m$ and $\Sigma_a$ are weighted by $N_m$ and $N_a$, respectively, which are required by PCA for minimizing the least-mean-square reconstruction error (Müller et al., 2001). Since normal data abound while outliers are very rare and hard to collect, the ratio $N_m/N_a$ is typically very large and $\Sigma_a$ is far less reliable than $\Sigma_m$. APCDA demonstrates that this weight imbalance causes some of the small eigenvalues of $\Sigma_m$ to be unexpectedly less than some unreliable small values of $\Sigma_a$ though their corresponding eigenvectors are more reliable. To overcome this issue, the less reliable covariance matrix, i.e., $\Sigma_a$, must be given higher weight (Jiang, 2009). The within-class scatter matrix is subsequently rewritten as $\Sigma_w^{\xi} = \xi_m \Sigma_m + \xi_a \Sigma_a$, where $\xi_m$ and $\xi_a$ are the adjusted weights of $\Sigma_m$ and $\Sigma_a$, respectively. They are uncorrelated to class prior probabilities of the training set and $\xi_m + \xi_a = 1$, $\xi_m < \xi_a$. The total scatter matrix now becomes:

$$\Sigma_t^{\xi} = \xi_m \Sigma_m + \xi_a \Sigma_a + \Sigma_b$$

By using $\Sigma_w^{\xi}$ and $\Sigma_t^{\xi}$ for feature extraction, APCDA is able to achieve a better result for the asymmetric classification task. This motivates us to apply the same heuristic in DROUT since it also targets at the same issue.

### 3.2 Subspace Decomposition and Feature Extraction

One would expect to extract features that minimize the within-class and maximize the between-class variances. Since the within-class variances are estimated from limited training data, the small variances estimated tend to be unstable and cause over-fitting. Hence,
similar to ERE, we first proceed with regularizing the three subspaces spanned by eigenvectors of the adjusted within-class scatter matrix $\Sigma_\nu^\xi$.

**Subspace identification:** By solving the eigenvalue problem for $\Sigma_\nu^\xi$, we obtain its $d$ eigenvectors $\{\phi_1, \phi_2, \ldots, \phi_d\}$ with corresponding eigenvalues $\lambda_1^w \geq \lambda_2^w \geq \ldots \geq \lambda_d^w$.

The set of eigenvectors is then divided into three subsets (corresponding to three subspaces): $\{\phi_1, \phi_2, \ldots, \phi_m\}$, $\{\phi_{m+1}, \phi_{m+2}, \ldots, \phi_r\}$, and $\{\phi_{r+1}, \phi_{r+2}, \ldots, \phi_d\}$ where $r = \max \{i | 1 \leq i \leq d \land \lambda_i^w \neq 0\}$ and $m$ is the index of the least eigenvalue in the principal subspace.

While the identification of $r$ is straightforward, finding the value of $m$ is slightly complex. More specifically, to determine the starting point of the noise dominant region $m + 1$, the point near the center of the noise region is identified by: $\lambda_{med}^w = \text{median}_{1 \leq r} \{\lambda_i^w\}$. The distance between $\lambda_{med}^w$ and the smallest nonzero eigenvalue is $d_{m,r} = \lambda_{med}^w - \lambda_m^w$. The upper bound of the unreliable eigenvalues is estimated by $\lambda_{med}^w + d_{m,r}$. The value of $m$ is subsequently defined as: $\lambda_{m+1}^w = \max_{1 \leq r} \{\lambda_i^w | \lambda_i^w < 2 \lambda_{med}^w - \lambda_r^w\}$.

**Subspace Regularization:** Based on ERE, the three subspaces spanned by eigenvectors of $\Sigma_\nu^\xi$ are regularized as follows (Jiang et al., 2008): (a) if $m < i \leq r$: $\tilde{\lambda}_i^w = \frac{\alpha}{\lambda_i^w + \beta}$, (b) if $m < i < r$: $\tilde{\lambda}_i^w = \frac{\alpha}{\lambda_i^w}$, and (c) if $r < i \leq d$: $\tilde{\lambda}_i^w = \frac{\alpha}{\lambda_i^w - \beta}$, where $\alpha = \frac{\lambda_{med}^w - \lambda_m^w}{\lambda_m^w - \lambda_m^w}$, and $\beta = \frac{\lambda_m^w - \lambda_m^w}{\lambda_m^w - \lambda_m^w - \lambda_{med}^w}$.

Let us denote:

$$\hat{\Phi}_w^d = [\hat{\phi}_1 \phi_i \ldots \hat{\phi}_d]$$

where $\hat{\phi}_i^w = 1/\sqrt{\tilde{\lambda}_i^w}$, and $\hat{\phi}_i^w$ is the resulting regularized eigenvalue. In words, $\hat{\phi}_w^d$ is the full-dimensional intermediate transformation matrix, meaning it is used to transform the original data space to another feature space without doing any dimensionality reduction. Specifically, an arbitrary data point $p$ of the original training data vector is transformed to $\hat{p} = (\hat{\Phi}_w^d)^T p$.

The weight-adjusted regularized total scatter matrix formed by the transformed training set is denoted as $\tilde{\Sigma}_\nu^\xi$ (computed based on (1)). By solving the eigenvalue problem for $\tilde{\Sigma}_\nu^\xi$, we obtain its $d$ eigenvectors $\{\tilde{\phi}_1^w, \tilde{\phi}_2^w, \ldots, \tilde{\phi}_d^w\}$ with corresponding eigenvalues $\lambda_1^w \geq \lambda_2^w \geq \ldots \geq \lambda_d^w$. Dimensionality reduction is carried out here by extracting the first $b$ eigenvectors with largest eigenvalues:

$$\hat{\Phi}_l^b = [\hat{\phi}_1 \phi_i \ldots \hat{\phi}_d]$$

Note that discriminant feature extraction is only done after eigenvectors of the adjusted within-class scatter matrix are regularized and no eigenvector is discarded before that. Hence, according to ERE, the discriminant capability of data is likely preserved. The final transformation matrix $A_T = \hat{\Phi}_w^d \hat{\Phi}_l^b$ is used for converting the $d$-dimensional testing set to the $b$-dimensional feature space (with $b < d$), and hence, reduces the data dimensionality.

### 3.3 Discussions

Though ERE is shown to be effective in extracting discriminative features for general classification task, it works directly on usual scatter matrices. According to APCDA, this
is irrelevant for asymmetric classification. In particular, the less reliable covariance matrix corresponding to the rare class (i.e., the class of anomalies) must be assigned higher weight. Though APCDA adjusts the scatter matrices for the feature extraction task, it starts extracting features at the eigenvalue regularization stage, which causes a loss in discriminant power (Jiang et al., 2008). The topic of dimensionality reduction for asymmetric classification is also explored in (Lindgren and Spångéus, 2004). Their technique, ACP, measures the spread of class $\omega_a$ with respect to $\omega_m$’s mean rather than to that of $\omega_a$ itself. It then solves the following generalized eigenvalue problem to extract discriminant features:

$$\tilde{\Sigma}_a D = \Sigma_m D \Lambda,$$

where (a) $\tilde{\Sigma}_a$ is the modified version of $\Sigma_a$, (b) $\Lambda$, $D^T \Sigma_m D$, and $D^T \Sigma_a D$ are diagonal. However, ACP neither considers the issue of imbalanced class cardinalities, nor the importance of the noise and null subspaces. As a consequence, ACP does not perform well for outlier detection (c.f., empirical evaluation in Section 4).

Combining the findings in both ERE and APCDA, we perform eigenspace decomposition and feature extraction on the weight-adjusted scatter matrices. The weights assigned here are unrelated to the class prior probabilities. This allows us to benefit from both feature extraction techniques for overcoming the curse of dimensionality in outlier detection. APCDA suggests to set $\xi_m = 0.2$ and $\xi_a = 0.8$. However, in outlier detection, the number of normal data points are expected to be much larger than anomalies. Hence we propose to use $\xi_m = 0.1$ and $\xi_a = 0.9$ with asymmetric ratio $\xi_a/\xi_m = 9$. In other words, we expect outliers to occupy only up to 10% of the total dataset’s size. This agrees with many previous studies (Angiulli and Fassetti, 2009; Lazarevic and Kumar, 2005). In this paper as well as in other these works, the number of outliers in an arbitrary dataset is assumed to be much less than 10% of the dataset’s cardinality (e.g., 1% or 5%). Nonetheless, we find that setting the asymmetric ratio to 9 is good enough for practical applications, though a good performance is achieved even with larger values.

The runtime cost of DROUT is $O(N_t d \cdot \min(N_t, d))$, which is comparable to other feature extraction techniques (Swets and Weng, 1996). However, since DROUT is only performed once on a small training set, this time complexity is not that important. Instead, the runtime overhead of the testing phase which is executed on a much larger dataset is of our great interest. Note that running detection methods on a transformed testing set will cost less time than on a full-dimensional one since their runtime overheads are proportional to the number of dimensions (Angiulli and Fassetti, 2009).

4. Outlier Detection with DROUT

In this section, we demonstrate the benefit of applying DROUT on two popular outlier detection techniques, through experiments on real datasets.

4.1 Detection Techniques

**ORCA:** In the field of distance-based outlier detection, ORCA (Bay and Schwabacher, 2003) is one of the most popular methods due to its high efficiency in terms of time complexity and accuracy. In ORCA, we aim to detect top $n$ outliers whose total distances to their respective $k$ nearest neighbors are largest. Since its outlier definition is based on the notion of nearest neighbors, and we know that nearest and farthest neighbors are roughly the same in such spaces (Aggarwal and Yu, 2005), ORCA suffers the curse of dimensionality
(i.e. its accuracy is reduced in high-dimensional spaces).

**BSOUT:** Kollios et al. (2003) introduces *Biased Sampling OUTlier Detection* (BSOUT) which aims to flag outliers whose total numbers of neighbors within radius $R$ (called $R$-neighborhood) are less than a threshold $P$. In BSOUT, each data point $p$'s local density is first estimated using a nonparametric kernel density estimator. Its $R$-neighborhood's cardinality is then approximated based on the calculated density. If this amount falls below $P$, it is placed in the candidate set which will be refined later to obtain true outliers. Similar to ORCA, the performance of BSOUT also degrades in high-dimensional data because of two reasons. While the first reason is analogous to ORCA’s, the second one stems from the fact that BSOUT utilizes nonparametric density estimation, and it is known that estimation accuracy downgrades greatly in such data (Müller et al., 2001).

### 4.2 Experiment Setup

**Testing Procedure:** We evaluate the performance of DROUT, ERE, APCDA, and ACP (Lindgren and Spångéus, 2004) when applying to ORCA and BSOUT. The detection accuracy here is measured as the area under the ROC curve, called AUC, which is widely used to assess outlier detectors. We compare the performance of ORCA and BSOUT on the original set of attributes, against ORCA on the new feature set obtained by applying each of the dimensionality reduction techniques. In order to use AUC as the evaluation metric, we employ real datasets that can be converted to the binary classification problem. This setup procedure has been successfully used for studying outlier detection (Lazarevic and Kumar, 2005).

**Benchmark Datasets:** The first dataset is extracted from the KDD Cup 1999 one following the method introduced in (Lazarevic and Kumar, 2005). Particularly, the smallest intrusion class, U2R, consisting of 246 data points is selected as the outlier class $\omega_a$. This class contains a variety of attacks like ftp_write, imap, multihop, nmap, phf, pod, and teardrop. The total dataset hence includes the normal class $\omega_m$ of 60593 data records and 246 outliers in $d$-dimensional space with $d = 34$ (we have excluded the 7 categorical attributes from the total of 41 attributes). The second dataset, Ann-Thyroid, is taken from the UCI Machine Learning Repository. It contains 3428 records in 21-dimensional space. The largest class (class 3) is selected as the normal class $\omega_m$, and we generate two test sets: Ann-Thyroid 1 ($\omega_a$ is class 1), and Ann-Thyroid 2 ($\omega_a$ is class 2). The maximum dataset dimensionality in our experiment is 34 which is similar to that of (Aggarwal and Yu, 2005).

### 4.3 Results

With the KDD dataset, we randomly sample 50 records from $\omega_a$ and 1000 records from $\omega_m$ for training, and keep the remaining 59789 records for testing. For the Ann-Thyroid dataset, we randomly select 50 records from $\omega_a$ and 450 records from $\omega_m$ for training, and keep the remaining (2751 records for Ann-Thyroid 1, and 2855 for Ann-Thyroid 2) for testing. Notice that the asymmetric ratio is 20 and 9 for the KDD and Ann-Thyroid datasets, respectively. This means our approach’s performance is also assessed in the case the asymmetric ratio is not exactly 9.
For ORCA, the number of nearest neighbors $k$ is varied in the range $0.02\%N \leq k \leq 0.1\%N$ with $N$ being the underlying dataset’s size, while $n$ is chosen to be $0.05\%N$. With BSOUT, $P$ is also varied from $0.02\%N$ to $0.1\%N$ and $R$ is chosen such that the number of outliers flagged using the algorithm in (Knorr and Ng, 1998) is exactly $0.05\%N$. These parameter settings follow the proposal in previous work (Angiulli and Fassetti, 2009). For each value of $b$ (number of extracted features) tested ($b \leq \lfloor d/2 \rfloor$), we construct the training set using random split described above for five times. The resultant average AUCs and their respective standard deviations are computed. Since the values of standard deviations are negligible, we do not present them in our results.

**Dimensionality Reduction on ORCA:** Figure 1 describes how the AUC values of ORCA using different dimensionality reduction methods change as $b$ increases. Notice that the performance of the original ORCA is unrelated to $b$. With small values of $b$, ORCA with feature extraction performs worse than the original ORCA. This is because by using insufficient number of features, discriminant information is likely lost even though the extraction process has been carefully designed to preserve it. However, for higher values of $b$, feature extraction starts producing better accuracy. The performance of APCDA is slightly better than ERE in general while ACP loses out in all test cases. On the other hand, DROUT achieves the best accuracy and highest gain in detection quality attributed to the fact that it performs dimensionality reduction on the adjusted scatter matrices and no feature rejection is carried out during the eigenspace regularization process. Overall, increasing $b$ does not ensure a better detection accuracy for methods utilizing feature extraction. This is reflected by a slight reduction and then relative stabilization of the AUC curves. The outcomes suggest that $b$ should not be too large (e.g., $b \leq \lfloor d/2 \rfloor$), otherwise the curse of dimensionality will happen again on the new feature space.

**Dimensionality Reduction on BSOUT:** From Figure 2, it can be seen that with BSOUT, the performance gain by applying feature extraction is even more pronounced. This is because, as compared to ORCA, BSOUT has one more factor causing its accuracy to downgrade in high-dimensional spaces: the nonparametric kernel density estimation. Therefore, reducing dimensions in BSOUT brings two benefits, it: (a) makes the notion of nearest neighbors more meaningful, and (b) improves the accuracy of estimating data local densities. Among the feature extraction techniques utilized, DROUT once again yields the...
Figure 2: Effect of feature extraction techniques on accuracy of BSOUT.

best accuracy. In addition, since ACP fails to preserve discriminant information, it suffers the worst performance. APCDA on the other hand slightly outperforms ERE. As \( b \) keeps increasing and exceeds a threshold, the accuracies of techniques based on feature extraction tend to first decrease and then become stable. These findings agree with the results obtained from the experiment on ORCA.

5. Conclusions

This paper explored the application of feature extraction on outlier detection research and proposed a novel method (DROUT) to accomplish the task. In brief, DROUT operates in two phases: eigenspace regularization and discriminant feature extraction. During the first phase, DROUT decomposes the data eigenspace into three components (the principal, the noise, and the null subspaces) where different regularization policies are applied and no subspace is discarded. This helps DROUT to preserve the discriminant information in the data before entering the actual feature extraction process. In the second phase, discriminant features are obtained from the regularized eigenspace by solving the traditional eigenvalue problem on the regularized total scatter matrix. One additional advantage of our method is that both of its phases are carried out on the weight-adjusted scatter matrices which makes DROUT better tuned to outlier detection than other existing techniques. Though the idea of doing feature extraction to improve the performance of outlier detectors in high-dimensional spaces is rejected by the subspace mining community (Aggarwal and Yu, 2005), empirical studies of DROUT applied to ORCA and BSOUT (two outstanding anomaly detectors) verify that DROUT (and hence, feature extraction methods) is able to bring nontrivial accuracy gain for detection methods. As future work, we are considering to extend our analysis on more large and high-dimensional datasets to better study the full benefits of DROUT. We are also carefully examining other possibilities of dimensionality reduction for outlier detection apart from our proposed technique. This will help us to better choose suitable ways for dealing with the curse of dimensionality.

References

Feature Extraction for Outlier Detection


Feature Selection for Text Classification Based on Gini Coefficient of Inequality

Sanasam Ranbir Singh
Department of Computer Science and Engineering
Indian Institute of Technology Guwahati
Guwahati 781039, Assam, India
ranbir@iitg.ernet.in

Hema A. Murthy
Department of Computer Science and Engineering
Indian Institute of Technology Madras
Chennai 600036, Tamil Nadu, India
hema@lantana.tenet.res.in

Timothy A. Gonsalves
Department of Computer Science and Engineering
Indian Institute of Technology Madras
Chennai 600036, Tamil Nadu, India
tag@lantana.tenet.res.in

Abstract

A number of feature selection mechanisms have been explored in text categorization, among which mutual information, information gain and chi-square are considered most effective. In this paper, we study another method known as within class popularity to deal with feature selection based on the concept Gini coefficient of inequality (a commonly used measure of inequality of income). The proposed measure explores the relative distribution of a feature among different classes. From extensive experiments with four text classifiers over three datasets of different levels of heterogeneity, we observe that the proposed measure outperforms the mutual information, information gain and chi-square static with an average improvement of approximately 28.5%, 19% and 9.2% respectively.

Keywords: Text categorization, feature selection, gini coefficient, within class popularity

1. Introduction

Text categorization (TC) is a supervised learning problem where the task is to assign a given text document to one or more predefined categories. It is a well-studied problem and still continues to be topical area in information retrieval (IR), because of the ever increasing amount of easily accessible digital documents on the Web, and, the necessity for organised and effective retrieval. High dimensionality of feature space is a major problem in TC. The number of terms (i.e., features) present in a collection of documents, in general, is large and few are informative. Feature selection for TC is the task of reducing dimensionality of feature space by identifying informative features and its primary goals are improving classification effectiveness, computational efficiency, or both. The performance of a classifier is affected by the employed feature selection mechanism.

©2010 Singh, Murthy and Gonsalves.
This paper proposes a feature selection mechanism called within class popularity (WCP) which addresses two important issues of feature selection for text classification i.e., uneven distribution of prior class probability and global goodness of a feature. The performance of WCP is then compared with the performance of the most commonly used measures – mutual information (MI), information gain (IG), chi-square (CHI).

The rest of the paper is organised as follows. Section 2 reviews few related studies. Section 3 presents the proposed feature selection. Section 4 presents experimental evaluations. The paper concludes in Section 5.

2. Review of Few Related Studies

At present feature selection methods for TC are based on statistical theory and machine learning. Some well-known methods are information gain, term frequency, mutual information, chi-square statics, Gini index (Yang and Pedersen, 1997; Shankar and Karypis, 2000). We briefly review these measures in this section.

2.1 Mutual Information:

Mutual information (MI) between a term $t$ and a class $c$ is defined by $MI(t, c) = \log \frac{Pr(t,c)}{Pr(t)Pr(c)}$. To measure the global goodness of a term in feature selection, we combine the category specific scores as $MI_{\text{max}}(t) = \max_i MI(t, c_i)$. Alternatively, in some studies (Yang and Pedersen, 1997), it is also define as $MI_{\text{max}}(t) = \sum_i Pr(c_i)MI(t, c_i)$.

2.2 Information Gain:

It is defined by following expression (Yang and Pedersen, 1997).

$$IG(t) = -\sum_i Pr(c_i) \log Pr(c_i) + Pr(t) \sum_i Pr(c_i|t) \log Pr(c_i|t) + Pr(\bar{t}) \sum_i Pr(c_i|\bar{t}) \log Pr(c_i|\bar{t})$$

It is frequently used as a term goodness criterion in machine learning. It measures the number of bits required for category prediction by knowing the presence or the absence of a term in the document.

2.3 $\chi^2$ static:

The $\chi^2$ static (CHI) is defined by the following expression (Yang and Pedersen, 1997).

$$\chi^2(t, c) = \frac{N \times (AD - CB)^2}{(A + C) \times (B + D) \times (A + B) \times (C + D)}$$

where $N$ is the number of documents, $A$ is the number of documents of class $c$ containing the term $t$, $B$ is the number of documents of other class (not $c$) containing $t$, $C$ is the number of documents of class $c$ not containing the term $t$ and $D$ is the number of documents of other class not containing $t$. It measures the lack of independence between $t$ and $c$ and comparable to $\chi^2$ distribution with one degree of freedom. The commonly used global goodness estimation functions are maximum and mean functions i.e., $\chi^2(f) = \arg \max_{c_i} \chi^2(f, c_i)$ or $\chi^2(f) = \sum_i Pr(c_i)\chi^2(f, c_i)$. 

77
3. Proposed Feature Selection

This section discusses the design and implementation details of the proposed within class popularity (WCP). The implementation details of text classifiers (seed-based, naive Bayes, kNN and SVM) are discussed in the Appendices A.

The proposed framework addresses the issues of uneven distribution of prior class probability and global goodness of a feature in two stages. First, it transforms the samples space into a feature specific normalized samples space without compromising the intra-class feature distribution. In the second stage of the framework, it identifies the features that discriminates the classes most by applying gini coefficient of inequality (Lorenz, 1905).

3.1 Transforming Samples Space

In the first stage of the proposed framework, we create a normalized samples space for each feature. Given a feature, the goal is to transform the original samples space into a normalized samples space of equal class size without altering the intra-class feature distribution.

To transform the samples space, we first define popularity of a feature $f$ within a class by a conditional probability of $f$ given a class label $c_i$ i.e. $Pr(f|c_i)$ using Laplacian smoothing as follows:

$$Pr(f|c_i) = \frac{1 + N(f, c_i)}{|V| + \sum_{f \in V} N(f, c_i)}$$ (1)

where $N(f, c_i)$ is the number of occurrences of the term $f$ in all the documents in $c_i$ and $V$ is the vocabulary set. Such a smoothing is important for classifiers such as naive Bayes where a sequence of the products of conditional probabilities is involved. Other smoothing techniques are also studied in (Wen and Li, 2007). Now, $Pr(f|c_i)$ defines intra-class distribution of a feature in a unit space. Thus, for a given feature $f$, each class can be normalized to the samples size of unit space without compromising feature distribution. Figure 1 shows the transformation pictorially. Dark area represents the portion of the samples containing the feature $f$ in a class.

In the normalized samples space, classes are evenly distributed. In an uniform space, the probability $Pr(c_i|f)$ (i.e., given a term $f$, what is the probability that a document belongs to the class $c_i$) is often effectively used to estimate the confidence weight of an association rule in data mining. We therefore apply the same concept to estimate the association between a
Feature Selection for Text Classification Based on Gini Coefficient of Inequality

class and a feature. We now normalize the above popularity weight (i.e. Equation 1) across all classes and define *within class popularity* as follows.

\[ w_{cp}(f, c_i) = \frac{Pr(f|c_i)}{\sum_{k=1}^{|C|} Pr(f|c_k)} \]  

(2)

where \( C \) is the set of the class labels. It has the following characteristics:

- \( \sum_i w_{cp}(f, c_i) = 1 \).
- \( w_{cp}(f, c_i) \) ranges between (0, 1) i.e., \( 0 < w_{cp}(f, c_i) < 1 \). It is because \( Pr(f|c_i) > 0 \).
- if a term \( f \) is evenly distributed across all classes then \( w_{cp}(f, c_i) = 1/|C| \forall c_i \in C \).
- if \( w_{cp}(f, c_i) > w_{cp}(f, c_j) \), then feature \( f \) is present more densely in the class \( c_i \) than class \( c_j \).
- if \( w_{cp}(f, c_i) \approx 1 \), then the feature \( f \) is likely to be present only in the class \( c_i \).

**Remark 1** The \( w_{cp}(f, c_i) \) is equivalent to \( Pr(c_i|f) \) in the normalized samples space. Since the classes are evenly distributed in the normalized samples space, \( w_{cp}(f, c_i) \) is un-biased to prior class probability.

As effectively used in association rules mining, with a reasonably high support weight (i.e., \( Pr(f|c_i) \)), a high value of \( w_{cp}(f, c_i) \) can represent high association between a class and a feature.

A conceptually similar feature selector has been used in (Aggarwal et al., 2004). However the estimator does not use smoothing while calculating \( Pr(f|c_i) \). Another difference is that it uses the square root of the sum of squares to estimate the distribution of a feature across different classes, whereas we use gini coefficient of inequality.

### 3.2 Global Goodness of a term

Commonly used global goodness estimators are *maximum* and *average* functions. Our goal is to identify the features that discriminates the classes most. A good discriminant term will
have skewed distribution across the classes. However, these two functions do not capture how a feature is distributed over different classes.

We use *gini coefficient of inequality*, a popular mechanism to estimate the distribution of income over a population, to analyse distribution of a feature across the classes. Pictorially, it can be shown as the plots in Figure 2. In the figure, gini of a population is defined by the area marked “A” divided by the areas marked “A” and “B” i.e., $gini = A/(A+B)$. If $gini = 0$, every person in the population receives equal percentage of income and if $gini = 1$, single person receives 100% of the income. A commonly used approach to represent the inequality and estimate the area under the curve is *Lorenz Curve* (Lorenz, 1905). In Lorenz curve, individuals are sorted by size in increasing order and the cumulative proportion of individuals (x-axis) is plotted against the corresponding cumulative proportion of their total size on y-axis. If we have a sample of $n$ classes, then the sample Lorenz curve of a term $t$ is the polygon joining the points $(h/n, L_h/L_n)$, where $h = 0, 1, 2, 3, ..., n$, $L_0 = 0$ and $L_h = \sum_i^h f(t, c_i)$ (Kotz et al., 1983). As shown in (Dixon et al., 1987), if the data is ordered increasing size, the Gini coefficient is estimated as follows.

$$G(t) = \frac{\sum_{i=1}^n (2i - n - 1) wcp(t, c_i)}{n^2 \mu}$$

(3)

where $\mu$ is sample mean. It has been shown that sample Gini coefficient calculated by Equation (3) is biased and is to be multiplied by $n/(n-1)$ to become unbiased.

### 3.3 Performance Metric

We use *F1 measure* (VanRijsbergen, 1979) to present performance of a classifier. F-measure is computed by calculating the harmonic mean of precision and recall as follows:

$$F = \frac{\alpha \cdot Precision \cdot Recall}{Precision + Recall}$$

F1-measure is commonly used F-measure where $\alpha = 2$. *Precision* is the ratio of correctly classified documents to the number of classified documents and *recall* is the ratio of correctly classified documents to the number of test documents. The F-measure is a binary class performance metric. In order to estimate F1-value for multi-class problem, we have used *micro-average estimation* (Yang, 1999).

### 4. Experimental Results

The performance of feature selection mechanisms are evaluated using four classifiers – seed-based, naive Bayes, kNN and SVM (Appendix A) over three datasets – Reuters-21578, 7Sectors-WebKB and a scan of the Open Directory Project. The classification performances over these datasets are evaluated using *5-fold cross validation*: four fold for training and one fold for testing and average over the 5-folds.

#### 4.1 Datasets and Characteristics

Table 1 summerizes the characteristics of the datasets. We briefly discuss the three datasets as follows:
Table 1: Characteristics of the Datasets. ‡ average over 5-fold after performing Porter stemming (Porter, 1980) and ignoring stopwords

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Reuters</th>
<th>ODP</th>
<th>7-Sectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Documents</td>
<td>21578</td>
<td>4,19,725</td>
<td>4582</td>
</tr>
<tr>
<td>Number of Terms‡</td>
<td>13,918.6</td>
<td>16,49,152</td>
<td>24,569.2</td>
</tr>
<tr>
<td>Number of Terms selected‡</td>
<td>3,845.4</td>
<td>28,721.6</td>
<td>6,288</td>
</tr>
<tr>
<td>Average Document size</td>
<td>63.8</td>
<td>346.4</td>
<td>194.8</td>
</tr>
<tr>
<td>Average #unique words/doc</td>
<td>38.1</td>
<td>131.9</td>
<td>96.5</td>
</tr>
<tr>
<td>Number of Categories</td>
<td>10</td>
<td>17</td>
<td>7</td>
</tr>
<tr>
<td>Evaluation Methodology</td>
<td>5-fold</td>
<td>5-fold</td>
<td>5-fold</td>
</tr>
<tr>
<td>Number of examples per fold</td>
<td>1,593</td>
<td>83,945</td>
<td>916</td>
</tr>
</tbody>
</table>

- **Reuters-21578** It is a highly skewed dataset containing 21578 news articles. For our experiments, we consider documents which are marked with **TOPICS** label. To ensure that each category contains a good number of training documents, as done in (Wang et al., 2007), we have considered the top 10 largest categories. We have considered the terms whose document frequency is at least 5.

- **7-Sectors WebKB** It is slightly skewed. We have considered the terms whose document frequency is at least 5. These documents are collected from different Web sources, which are developed and maintained by different groups of people.

- **Open Directory Project** We use Open Directory Project taxonomy from the March 2008 archive. This taxonomy consists of 4,592,207 number of urls and 17 classes in its top label. We have arbitrarily selected 419725 number of urls and locally crawled. We have considered the terms whose document frequency is at least 100.

4.2 Performance of Feature Selection Mechanisms

The experiments are executed with different feature space. Initially, features are ordered by their global goodness weights and then define the feature dimension by 10%, 20% and so on up to 100% of the selected features (refer Table 1). Table 2 shows a comparison of microaverage F1 measures among feature selectors using four text classifiers (seed-based, naive Bayes, kNN and SVM). It shows the minimum, average and maximum micro-average F1 values of different classifiers over different feature dimensions with different feature selectors over different datasets.

All four classifiers perform relatively well using WCP. Except on two instances i.e., Naive Bayes with CHI over Reuters-21578 and 7Sectors-WebKB datasets, WCP outperforms all other feature selectors in all instances. It is also observed that all four text classifiers perform relatively well on news dataset (Reuters-21578). The performance of the classifiers over 7Section-WebKb is moderate and performance over Open Directory Project is poor.
Table 2: Show minimum, average and maximum value of micro average F1 measure across different classifiers using different feature selectors

<table>
<thead>
<tr>
<th></th>
<th>Reuters-21578 Collections</th>
<th>Open Directory Project Collections</th>
<th>7-Sectors-WebKB Collections</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FS</td>
<td>Seed</td>
<td>kNN</td>
</tr>
<tr>
<td></td>
<td>Min Avg Max</td>
<td>Min Avg Max</td>
<td>Min Avg Max</td>
</tr>
<tr>
<td>WCP</td>
<td>0.91 0.94 0.95</td>
<td>0.91 0.92 0.94</td>
<td>0.89 0.91 0.93</td>
</tr>
<tr>
<td>CHI</td>
<td>0.81 0.81 0.81</td>
<td>0.90 0.91 0.92</td>
<td>0.83 0.89 0.90</td>
</tr>
<tr>
<td>IG</td>
<td>- - -</td>
<td>0.83 0.85 0.86</td>
<td>0.84 0.88 0.89</td>
</tr>
<tr>
<td>MI</td>
<td>0.26 0.58 0.86</td>
<td>0.49 0.78 0.91</td>
<td>0.56 0.77 0.89</td>
</tr>
</tbody>
</table>

It verifies the claim that traditional text classifiers with traditional feature selectors are not suitable for extremely heterogeneous dataset. Table 3 shows the average performance of each feature selector across all classifiers and datasets. Table 4 shows the performance improvement of different classifiers when they use WCP feature selector over the performance obtained by the same classifier using MI, CHI and IG feature selectors. There is an overall improvement of 25.4% over MI, 6.8% over CHI, and 16.2% over IG. In brief, we have observed the following – (i) Overall WCP is suitable for all datasets, (ii) Overall WCP is suitable for all classifiers, (iii) MI performs the least among the feature selectors, (iv) all four classifiers perform equally well on Reuters-21578 dataset, (v) With carefully selected examples (7Sectors-WebKB), traditional text classifiers can also provide high performance on Web document collections.
Table 3: Average performance over all classifiers using different datasets and over all datasets using different classifier

<table>
<thead>
<tr>
<th>Feature Selector</th>
<th>Over all Classifiers</th>
<th>Over all datasets</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reuters</td>
<td>Sectors</td>
</tr>
<tr>
<td>Within-Class-Popularity</td>
<td>0.93</td>
<td>0.58</td>
</tr>
<tr>
<td>CHI-square</td>
<td>0.89</td>
<td>0.52</td>
</tr>
<tr>
<td>Information Gain</td>
<td>0.876</td>
<td>0.46</td>
</tr>
<tr>
<td>Mutual Information</td>
<td>0.68</td>
<td>0.51</td>
</tr>
</tbody>
</table>

Table 4: Improvement in performance of different classifiers using WCP over other feature selectors

<table>
<thead>
<tr>
<th></th>
<th>Seed</th>
<th>kNN</th>
<th>NB</th>
<th>SVM</th>
<th>Reuters</th>
<th>7Sectors</th>
<th>ODP</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Over CHI</td>
<td>13.2%</td>
<td>8.6%</td>
<td>1.5%</td>
<td>10.3%</td>
<td>4.5%</td>
<td>11.5%</td>
<td>15%</td>
<td>9.2%</td>
</tr>
<tr>
<td>Over IG</td>
<td>-</td>
<td>8.6%</td>
<td>18.5%</td>
<td>19.6%</td>
<td>5.7%</td>
<td>26.1%</td>
<td>35.3%</td>
<td>19%</td>
</tr>
<tr>
<td>Over MI</td>
<td>30.5%</td>
<td>23.5%</td>
<td>28%</td>
<td>31.4%</td>
<td>36.8%</td>
<td>13.7%</td>
<td>35.3%</td>
<td>28.5%</td>
</tr>
</tbody>
</table>

5. Conclusion

In this paper we study a feature selection mechanism called within-class-popularity, which measures normalized popularity of a term within a class. It uses Gini coefficient of inequality to estimate global goodness of a term. The performance of WCP is then compared with that of mutual information, chi-square and information gain using four text classifiers (seed-based, kNN, naive Bayes and SVM) over three datasets (Reuters-21578, 7Sectors-WebKB and Open Directory Project). From extensive experiments, it is found that on an average WCP outperforms MI, CHI and IG feature selectors.

References


Appendix A. Experimental Text Classifiers

We use vector space model to represent documents (Salton et al., 1975) for the TC. In vector space model, a document \( d \) is represented by a term vector of the form \( d = \{ w_1, w_2, ..., w_n \} \), where \( w_i \) is a weight associated with the term \( f_i \). We use TF-IDF and cosine normalisation (Aggarwal et al., 2004) to define the weight of a feature \( f_i \) in a document vector \( d \) as follows:

\[
w_i = \frac{tfidf(f_i, d)}{\sqrt{\sum_k tfidf(f_k, d)^2}} \quad \text{and} \quad tfidf(f_i, d) = tf(f_i, d) \cdot \log \frac{|D|}{df_D(f_i)},
\]

where \( tf(f_i, d) \) is the term frequency of \( f_i \) in \( d \), \( D \) is the document set and \( df_D(f_i) \) is document frequency of the term \( f_i \).

A.1 Seed-based Classifier

In our study, we design a Seed-based classifier (also known as centroid based classifier) especially for WCP. Each class is represented by a term vector known as seed. We define a pseudo-seed \( c_i \) for each class \( c_i \) as follows:

\[
c_i = \{ w_f | w_f = wcp(f, c_i), \forall f \in F \}
\]

where \( F \) is a set of selected features. Given a test example \( d \) defined over \( F \), we classify \( d \) by the following function:

\[
\text{classify}(d) = \arg \max c_i \{ \text{cosine}(d, c_i) \}
\]

where \( \text{cosine}(d, c_i) \) is the cosine similarity between \( d \) and \( c_i \). IG does not provide class specific weight. Therefore, it is omitted from exploring the seed-based classifier.
A.2 naive Bayes

Assuming naive’s condition i.e., features are conditionally independent, we defined naive Bayes classifier by

\[ Pr(c_k|d_i) = \frac{Pr(c_k) \prod_j Pr(d_{ij}|c_k)}{\prod_j Pr(d_{ij})} \]

As denominator is independent of class, effectively, we have estimated \( Pr(c_k|d_i) \) as \( \prod_j Pr(d_{ij}|c_k) \), where \( Pr(d_{ij}|c_k) \) is defined by Equation 1.

A.3 kNN

Cosine similarity is used to estimate distance between test examples and training examples. For each test sample, at most 30 nearest neighbours are considered to count for the winner class. In the case of Open Directory Project dataset having very large number of documents, we have randomly selected only 100 test examples from each class and 400 examples from each training class and estimated similarity between 1700 test examples and 6800 training examples.

A.4 SVM

We use the SVMTorch software \(^1\) for our reported estimations, which is publicly available for download. Again training SVM with large dataset is very expensive. Therefore, like kNN, we have randomly selected only 100 test examples and 400 training examples from each class. We run svm tool using linear kernel. From various experiments, we find that linear kernel perform better compared to radial and Gaussian kernel.

---

Increasing Feature Selection Accuracy for $L_1$ Regularized Linear Models in Large Datasets

Abhishek Jaiantilal
Department of Computer Science
University of Colorado
Boulder, CO, 80309, USA

Gregory Grudic
Flashback Technologies, LLC
Longmont, CO, 80503, USA

Editor: Huan Liu, Hiroshi Motoda, Rudy Setiono, and Zheng Zhao

Abstract

$L_1$ (also referred to as the 1-norm or Lasso penalty) penalty based formulations have been shown to be effective in problem domains when noisy features are present. However, the $L_1$ penalty does not give favorable asymptotic properties with respect to feature selection, and has been shown to be inconsistent as a feature selection estimator; e.g. when noisy features are correlated with the relevant features. This can affect the estimation of the correct feature set, in certain domains like robotics, when both the number of examples and the number of features are large. The weighted lasso penalty by (Zou, 2006) has been proposed to rectify this problem of correct estimation of the feature set. This paper proposes a novel method for identifying problem specific $L_1$ feature weights by utilizing the results from (Zou, 2006) and (Rocha et al., 2009) and is applicable to regression and classification algorithms. Our method increases the accuracy of $L_1$ penalized algorithms through randomized experiments on subsets of the training data as a fast pre-processing step. We show experimental and theoretical results supporting the efficacy of the proposed method on two $L_1$ penalized classification algorithms.

Keywords: Feature selection, $L_1$ penalized algorithms

1. Introduction

Feature selection using the $L_1$ penalty (also referred as 1-norm or Lasso penalty) has been shown to perform well when there are spurious features mixed with relevant features and this property has been extensively discussed in (Efron et al., 2004), (Tibshirani, 1996) and (Zhu et al., 2003). In this paper, we focus on feature selection via the $L_1$ penalty for classification, addressing open problems related to feature selection accuracy and large datasets. This paper is organized as follows, Section-2 presents motivation and background, primarily focusing on the fact that asymptotically $L_1$ penalty based method might include spurious features. Based on the work in (Zou, 2006), we show that random sampling can find a set of weights that improves accuracy over the unweighted (which is normally used) $L_1$ penalty methods and we detail this in Section-3. In Section-4, we show results on two different classification algorithms and compare the weighted method proposed in (Zou, 2007) with the random sampling method described in our paper. Our method differs from Zou’s
Increasing Feature Selection Accuracy for \( L_1 \) Regularized Linear Models

method as it hinges on random sampling to find the weight vector instead of using the \( L_2 \) penalty. The proposed method is shown to give significant improvement in accuracy over a number of data sets. Section 5 summarizes the results and concludes with future work.

The contribution of our work is as follows: we show that a fast pre-processing step can be used to increase the accuracy of \( L_1 \) regularized models and is a good fit when the number of examples are large; we connect the theoretical results from (Rocha et al., 2009) showing the viability of our method on various \( L_1 \) penalized algorithms and also show empirical results supporting our claim.

2. Background Information and Motivation

Consider the following setup in which information about \( n \) examples, each with \( p \) dimensions, is represented in a \( n \times p \) design matrix denoted by \( X \), with \( y \in \mathbb{R}^n \) representing target values/labels, and \( \beta \in \mathbb{R}^p \) representing a set of model parameters to be estimated. For our paper, we consider classification based linear models with a convex loss function and a penalty term (a regularizer). In (1), we show a regularized formulation that can be used to generally describe many machine learning algorithms. The metric or loss, \( L(X,y,\beta) \), may represent various loss functions including ‘hinge loss’ for classification based Support Vector Machines (SVMs) and ‘squared error loss’ for regression.

\[
\beta = \arg \min_{\beta} L(X,y,\beta) + \lambda J(\beta)
\]

where \( L(X,y,\beta) = \) loss function, \( J(\beta) = \) penalty function and \( \lambda \geq 0 \)

Popular forms of the penalty functions (\( J(\beta) \)) are by using the \( L_2 \) and \( L_1 \) norm on \( \beta \) and are termed Ridge and Lasso penalty respectively in literature (refer to (Tibshirani, 1996)).

2.1 Asymptotic properties of \( L_1 \) penalty

Many papers including (Tibshirani, 1996), (Efron et al., 2004) and (Zhu et al., 2003) discuss the merits of the \( L_1 \) penalty. The \( L_1 \) penalty has been shown to be efficient in producing sparse models (models with many of the \( \beta \)'s set to 0) and this feature selecting ability makes it robust against noisy features. In addition, the \( L_1 \) penalty is a convex penalty and when used in conjunction with convex loss functions, the resultant formulation has a global minimum.

As the \( L_1 \) penalty is used for simultaneous feature selection and correct estimation, a topic of interest is to understand whether sparsity holds when \( n \to \infty \), \( n \) = number of examples. Intuitively, given enough samples, the estimated parameters \( \beta_n \) should approach the true parameters \( \beta_0 \).

\[
y = X\beta_0 + \epsilon
\]

Assume that the data is generated as shown in (2), with \( \epsilon \) being gaussian noise of zero mean and \( \beta_0 \) being the true generating model parameters. Also, \( \beta_{0j} \) represents the \( j^{th} \) feature for \( \beta_0 \). If \( A_0 = \{ j \mid \beta_{0j} \neq 0 \} \) is the true model and \( A_n \) is the model found for \( n \to \infty \). For consistency in feature selection, we need \( A_n = \{ j \mid \beta_{nj} \neq 0 \} \) and \( \lim_{n \to \infty} P(A_n = A_0) = 1 \), that is we find the correct set of features \( A_0 \) asymptotically. (Zou, 2006) showed that lasso estimator is consistent (in terms of \( \beta_N \to \beta_0 \)) but can be inconsistent as a feature selecting estimator in presence of correlated noisy features.
2.1.1 Hybrid SVM

(Zou, 2006) showed that weighted lasso penalty as shown in (3) and which is termed as the weighted lasso regression, can be used for simultaneous feature selection and creating accurate models. In (Zou, 2007), the same properties are applied in case of classification and referred to as ‘Improved 1-norm SVM’ or ‘Hybrid SVM’. The weighted lasso formulations for regression and classification are shown in (3) and (4) respectively. In (3), \( \beta(\text{OLS}) \) denotes the weights found via least squares regression. For the weighted lasso penalty, the formulations in (3) and (4) are still convex and will require almost no modification to the (unweighted) lasso penalty based algorithms. Refer to (Zou, 2006) for the modifications that are needed. Intuitively, the weights found via the \( L_2 \) penalty are inversely proportional to the true model parameter \( \beta_0 \). If those weights are lower (i.e. the true model magnitude is higher) then in the weighted lasso penalty we are penalizing those corresponding features lesser and thereby encouraging those features to have higher magnitude in the weighted \( L_1 \) models and vice-versa for noisy features.

Weighted Lasso Regression: \[
\min_{\beta} \left\| y - X\beta \right\|^2 + \lambda \sum_j W_j |\beta_j| \quad \text{s.t. } W_j = |\beta(\text{OLS})_j|^{-\gamma}, \gamma > 0 \tag{3}
\]

Improved 1-norm SVM: \[
\min_{\beta,\beta_0} \sum_i [1 - y_i(x_i,\beta + \beta_0)]_+ + \lambda \sum_j W_j |\beta_j|, \tag{4}
\]

where \( W_j = |\beta(l_2)_j|^{-\gamma}, \gamma > 0 \), \( \beta(l_2) = \arg \min_{\beta,\beta_0} \sum_i [1 - y_i(x_i,\beta + \beta_0)]_+ + \lambda \sum_j |\beta_j|^2 \)

Improved SVM2: \[
\min_{\beta,\beta_0} \sum_i [1 - y_i(x_i,\beta + \beta_0)]_+^2 + \lambda \sum_j W_j |\beta_j|, \tag{5}
\]

where \( W_j = |\beta(l_2)_j|^{-\gamma}, \gamma > 0 \), \( \beta(l_2) = \arg \min_{\beta,\beta_0} \sum_i [1 - y_i(x_i,\beta + \beta_0)]_+^2 + \lambda \sum_j |\beta_j|^2 \)

\( \{x_i, y_i\} \) represents an example, \( \lambda, \lambda_2 \) are regularizing parameters. \( v_+ = \max(v, 0) \) in the above equations.

2.2 Motivation for our Method

The weighted lasso penalty is dependent on obtaining suitable weights ‘\( W \)’. (Zou, 2006, 2007) shows that the ordinary least squares estimates and the estimates from SVM with the \( L_2 \) norm penalty can be used to find the weights as shown in (3) and (4). For our paper, we obtain these weights via feature selection on randomized subsets of the training data. If the accuracy is higher than the unweighted case, it means that the features are appropriately (and correctly) weighted.

One of our goals was to see the translation of results from (Zou, 2006) to other linear formulations and thus we also experimented on the weighted SVM2 formulation shown in (5) (unweighted formulation is shown in (7)). The SVM2 formulation is referred to in literature as Quadratic loss SVM (but with \( L_2 \) penalty) or 2-norm SVM (refer to (Shawe-Taylor and Cristianini, 2004)). It is squared hinge loss coupled with the \( L_1 \) penalty.

2.2.1 Efficient Algorithms to solve formulations with \( L_1 \) norm penalty

(Efron et al., 2004) showed an efficient algorithm for lasso regression called Least Angle Regression (LARS), that can solve for all values of \( \lambda \), that is \( 0 \leq \lambda \leq \infty \). In (Rosset and
Increasing Feature Selection Accuracy for $L_1$ Regularized Linear Models

Zhu, 2007), a generic algorithm, for which LARS is a special case, is documented that can be used for all double differentiable losses with the $L_1$ penalty. For our experiments, we resort to specific linear SVM based formulations for which entire regularization paths can be constructed. (6) is the penalized formulation for ‘1-norm SVM’. (Zhu et al., 2003) showed a simple piecewise algorithm to solve for $0 \leq \lambda \leq \infty$ in the 1-norm SVM. As the loss and the penalty function are both singly differentiable, a piecewise path cannot be constructed as efficiently as in LARS but linear programming can be employed to calculate the step size. (7) is an equivalent to (6) and similar to the formulation seen in literature except with the $L_2$ loss function instead of the $L_1$ loss function. (7) is the penalized formulation for squared hinge loss (or Quadratic loss SVM) with the $L_1$ penalty. As the loss function is doubly differentiable, via the method described by (Rosset and Zhu, 2007), an efficient piecewise algorithm that be constructed to solve for $0 \leq \lambda \leq \infty$. Our vested interest in using such piecewise algorithms, is to help understand whether better (entire) regularization paths are created or not for the weighted $L_1$ penalty.

$$1\text{-norm SVM: } \min_{\beta, \beta_0} \sum_i [1 - y_i(x_{:,i}\beta + \beta_0)]_+ + \lambda \sum_j |\beta_j|,$$

Equivalent to (6) : $\min_{\beta, \beta_0} ||\beta||_1 + C \sum_i \xi_i, \text{s.t. } y_i(x_{:,i}\beta + \beta_0) \geq 1 - \xi_i, \xi_i \geq 0$

$$\text{SVM2: } \min_{\beta, \beta_0} \sum_i [1 - y_i(x_{:,i}\beta + \beta_0)]_+^2 + \lambda \sum_j |\beta_j|,$$

$$v_+ = \max(v, 0) \text{ in the above equations.}$$

3. Randomized Sampling (RS) Method to Create Weight Vector

Our randomized sampling method depends on a small random subset of training data. We assume that the subset of the training data is small, i.e. it is computationally cheap to act on such a set in a reasonable time. Also, such randomized sampling is done multiple times.

3.1 Randomized Sampling (RS) Method

Our randomized sampling algorithm is described below in Algorithm-1: Randomized Sampling Method. The algorithm can be explained as follows: We choose a subset of $m$ examples out of the presented $n$ examples such that $m << n$. We train a $L_1$ penalty based algorithm (e.g. 1-norm SVM (Zhu et al., 2003), SVM2, etc.) so that we can find a set of relevant features. We keep a note of the features that we found in that particular experiment. After many such randomized experiments, the counts of the number of times a feature was found in these randomized experiments is summed up and normalized and denoted by $V$.

This count vector, denoted by $V$, is then inverted and used as weights for the weighted version of the algorithm; i.e. weights used in the weighted formulations are $W = 1/V$. Intuitively, if a feature is important and is found multiple times via the RS method, then the corresponding weight for the feature is less and thus it is penalized lesser, encouraging higher magnitude for the feature.
**Algorithm-1: Randomized Sampling (RS) Method:**

**Input:** \( n \) examples each with \( p \) features, \( K \) randomized experiments, \( B \) (Block) number of examples used to train model in each randomized experiment.

**Output:** Count vector \( W \) (1xD vector) representing number of times features were selected in \( K \) randomized experiment.

Divide \( N \) examples into \( K \) randomized sets each of size \( B \) and denote them as \( N_{trn_i}, i = 1 \ldots K \) and let \( V \leftarrow 0 \)

for \( i = 1 \ldots K \) do
  Get \( N_{trn_i} \), construct \( N_{tst_i} \) and \( N_{val_i} \) set.
  Train Model
    \( _i \) = \( L_1 \) Algorithm(\( N_{trn_i}, N_{tst_i}, N_{val_i} \))
  \( S_i \) = selected features in Model
  \( V \leftarrow V + \{ x, x \in \mathbb{R}^D | x_j = 1 \text{ if } j \in S_i \text{ else } x_j = 0 \} \)
end

### 3.2 Consistency of choosing a set of Features from Randomized Sampling (RS) Experiments

Our method is dependent on finding some set of relevant features and their counts for a given dataset via the RS method. Our experimental results are restricted to the weighted and unweighted formulation for SVM2 and 1-norm SVM, but our theoretical results are applicable to all linear models with twice differentiable loss function with the \( L_1 \) penalty.

We next mention results, regarding the asymptotic consistency and normality properties in \( n \) (number of examples) for \( L_1 \) penalized algorithms, which can help understand the consistency of our method.

**Lemma 1:** This result is from Theorem-5 in (Rocha et al., 2009). If the loss function \( L(X,y,\beta) \) shown in (1) is bounded, unique and a convex function, with \( E|L(X,y,\beta)| < \infty \) and furthermore \( L(X,y,\beta) \) is twice differentiable with a positive hessian \( H \) matrix, then the following consistency condition defined for the \( L_1 \) penalty when using the formulation in (1) and true model in (2):

\[
||H_{A_c,A}[H_{A,A} - HH_{A_0,A}^{-1}H_{A_0,A}]^{-1}sign(\beta_A)||_\infty \leq 1, \text{ where } H_{x,y} = \frac{d^2L(X,y,\beta)}{dxdy} \quad (8)
\]

Where \( A_c = \{ j \in 1...p | \beta_j = 0 \} \), \( A = \{ j \in 1...p | \beta_j \neq 0 \} \) and \( \beta_0 \) is an intercept.

- if \( \lambda_n \) is a sequence of non-negative real numbers such that \( \lambda_n n^{-1} \rightarrow 0 \) and \( \lambda_n n^{-(1+c)/2} \rightarrow \lambda > 0 \) for some \( 0 < c < 1/2 \) as \( n \rightarrow \infty \) and the condition (8) is satisfied then \( P[sign(\beta_n(\lambda_n)) = sign(\beta)] \geq 1 - exp[-n^c] \). \( \beta_n \) is parameter found for number of examples=\( n \).

- If the condition in (8) is not satisfied then for any sequence of non-negative numbers \( \lambda_n \)
  \[
  \lim_{n \rightarrow \infty} P[sign(\beta_n(\lambda_n)) = sign(\beta)] < 1. \text{ The probability of choosing incorrect variables is bounded to } exp(-Dn^c), \text{ where } D \text{ is a positive constant (shown in the proof of Theorem 5 of (Rocha et al., 2009))}. \]

If the condition in (8) is fulfilled, it means that the interactions between relevant and noisy features are distinguishable and the \( L_1 \) penalty can correctly identify the signs in \( \beta \). If
the condition in (8) is not fulfilled, then noisy features will be added to the model with a probability away from 1. Also, note that the above conditions are applicable for 1-norm SVM, as shown in (Rocha et al., 2009).

**Lemma 2:** We use b to specify the size of the subset and assuming $b \to \infty$, then from Lemma-1, when condition of consistency (8) is satisfied then $P[\text{sign}(\beta_b(\lambda_b) = \text{sign} (\beta))] \ge 1 - \exp(-b^c) \approx 1$, where $\beta_b$ and $\lambda_b$ represent the parameters for the subset of size $b$. For $k$ such subsets $V$, as depicted in the algorithm in Section 3.1, is bounded to $k(1 - \exp(-b^c)) \approx k$, $b \to \infty$. When the condition in (8) is not satisfied then the probability of choosing noisy variables in a subset is upperbounded to $\exp(-Db^c)$ and for $k$ subsets, $\sum(V_j) \le k \cdot \exp(-Db^c)$ and $V_j \approx 0$, $b \to \infty$, (where $V_j$ are indices of noisy variables). Thus, the noisy variables have a probability of having a low count in $V$ and a large weight in $W$, thus penalizing the noisy features heavily.

**Table 1: Mean ± Std. Deviation of Error Rates in % on Models 1 & 4 by SVM2**

<table>
<thead>
<tr>
<th>$q$</th>
<th>$p$</th>
<th>2-norm SVM2</th>
<th>1-norm SVM2</th>
<th>Hybrid (Zou)</th>
<th>RS(20%)</th>
<th>RS(30%)</th>
<th>RS(40%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>14</td>
<td>9.64±2.30</td>
<td>7.92±1.89</td>
<td>7.88±2.09</td>
<td>7.69±1.71</td>
<td>7.67±1.66</td>
<td>7.68±1.69</td>
</tr>
<tr>
<td>4</td>
<td>27</td>
<td>10.90±2.41</td>
<td>8.01±1.84</td>
<td>7.88±2.09</td>
<td>7.73±1.59</td>
<td>7.73±1.59</td>
<td>7.71±1.60</td>
</tr>
<tr>
<td>6</td>
<td>44</td>
<td>12.17±2.64</td>
<td>7.93±1.79</td>
<td>7.79±1.69</td>
<td>7.64±1.60</td>
<td>7.64±1.59</td>
<td>7.64±1.52</td>
</tr>
<tr>
<td>8</td>
<td>65</td>
<td>13.45±2.96</td>
<td>8.13±2.10</td>
<td>7.87±1.84</td>
<td>7.82±1.85</td>
<td>7.83±1.85</td>
<td>7.81±1.83</td>
</tr>
<tr>
<td>12</td>
<td>119</td>
<td>16.91±3.24</td>
<td>8.11±1.95</td>
<td>8.05±1.94</td>
<td>7.78±1.71</td>
<td>7.78±1.70</td>
<td>7.76±1.66</td>
</tr>
<tr>
<td>16</td>
<td>189</td>
<td>17.93±3.32</td>
<td>7.87±1.78</td>
<td>8.29±2.41</td>
<td>7.66±1.57</td>
<td>7.66±1.63</td>
<td>7.66±1.63</td>
</tr>
<tr>
<td>20</td>
<td>275</td>
<td>19.31±3.32</td>
<td>8.06±2.14</td>
<td>8.04±2.01</td>
<td>7.69±1.81</td>
<td>7.74±1.89</td>
<td>7.77±1.87</td>
</tr>
</tbody>
</table>

**Random Sampling is Subsampling:** To better quantify our random sampling method, we explain it in terms of subsampling (refer to (Politos et al., 1999)). Subsampling is a method of sampling $m$ examples from $n$ total examples with $m < n$, unlike bootstrap that samples $n$ times with replacement from $n$ samples. Let estimator $\theta$ be a general function of i.i.d data generated from some probability distribution $P$. In our case of feature selection, this estimator is the feature set. We are interested in finding an estimator and it’s confidence region based on the probability $P$ of the data and we define it as $\theta(P)$. When $P$ is large then we can construct an empirical estimator $\hat{\theta}_n$ of $\theta(P)$ such that $\hat{\theta}_n = \theta(P_n)$, where $P_n$ is the empirical distribution; that is estimate the true feature set empirically. We define a root of form $\tau_n(\theta - \theta)$, where $\tau_n$ is some sequence (like $\sqrt{n}$ or $n$) increasing with $n$ (number of examples), and we are looking at the difference between the empirical estimator $\hat{\theta}_n$ and the true estimator $\theta$. We define $J_n(P)$ to be the sampling distribution of $\tau_n(\theta - \theta(P))$ based on a sample size of $n$ from $P$ and define the CDF as

$$J_n(x, P) = \text{Probability}_P \{\tau_n(\hat{\theta}_n - \theta) \le x\}, \quad x \in R$$ (9)

**Lemma 3:** From (Politos et al., 1999), for data generated via i.i.d., there is a limiting law $J(P)$ such that $J_n(P)$ converges weakly (in probability) to $J(P)$ and $\tau_n(\theta_n - \theta) \to 0$ as $n \to \infty$ with the conditions that $n_b/\tau_n \to 0$, $b \to \infty$ and $b/n \to 0$, where $b$ is the number of examples in the subsample experiment and $n$ is the total number of available examples.

Lemma 3, has remarkably weak conditions for subsampling and it requires that the root has some limiting distribution and the sample size $b$ is not too large (but still going to infinity) compared to $n$. In our case, the subsets are of size $b \to \infty$, $b << n$ and for the rate of estimation at $\tau_n \propto n^c$, $\tau_b \propto b^c$, $0 < c \le 1$, then $n_b/\tau_n \to 0$. For the RS method, we create weight vector whose index for a feature is non-zero if that feature was found in...
Table 2: Mean ± Std. Deviation of Error Rates in % on Models 1 & 4 by 1-norm SVM

<table>
<thead>
<tr>
<th>q</th>
<th>p</th>
<th>2-norm SVM</th>
<th>1-norm SVM</th>
<th>Hybrid (Zou)</th>
<th>RS(20%)</th>
<th>RS(30%)</th>
<th>RS(40%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>14</td>
<td>8.74±1.30</td>
<td>7.64±0.09</td>
<td>7.64±1.02</td>
<td>7.63±1.02</td>
<td>7.64±1.01</td>
<td>7.53±0.09</td>
</tr>
<tr>
<td>4</td>
<td>27</td>
<td>9.76±1.75</td>
<td>8.75±1.14</td>
<td>7.95±1.34</td>
<td>7.83±1.28</td>
<td>7.79±1.24</td>
<td>7.69±1.19</td>
</tr>
<tr>
<td>6</td>
<td>44</td>
<td>10.57±1.95</td>
<td>10.57±1.95</td>
<td>9.72±1.12</td>
<td>7.97±1.12</td>
<td>7.77±1.18</td>
<td>7.69±1.23</td>
</tr>
<tr>
<td>8</td>
<td>65</td>
<td>11.47±2.31</td>
<td>9.81±0.99</td>
<td>8.99±1.36</td>
<td>7.75±1.13</td>
<td>7.74±1.15</td>
<td>7.63±1.09</td>
</tr>
<tr>
<td>12</td>
<td>119</td>
<td>13.27±2.48</td>
<td>13.27±2.48</td>
<td>11.36±1.35</td>
<td>7.77±1.16</td>
<td>7.82±1.21</td>
<td>7.63±1.00</td>
</tr>
<tr>
<td>16</td>
<td>189</td>
<td>15.58±2.94</td>
<td>15.58±2.94</td>
<td>13.21±1.31</td>
<td>7.74±1.31</td>
<td>7.75±1.23</td>
<td>7.64±1.14</td>
</tr>
<tr>
<td>20</td>
<td>275</td>
<td>17.14±2.96</td>
<td>17.14±2.96</td>
<td>15.11±1.11</td>
<td>7.77±1.20</td>
<td>7.80±1.27</td>
<td>7.69±1.19</td>
</tr>
</tbody>
</table>

a particular experiment. \( \hat{\theta}_n \) is the sample mean of \( n \) such RS experiments weights, having mean converging to \( \theta(P) \) (due to Lemma-2). Thus estimating the true feature set on basis of random sampling of subsets of data is weakly convergent. (Zou, 2006) used a root-n-consistent estimator’s weight (from the \( L_2 \) penalty) but mentions that the conditions can be further weakened and if there is an \( a_n \) such that \( a_n \to \infty \) and \( a_n(\hat{\theta} - \theta) = O(1) \) then such an estimator can also be used. By Lemma-3, our RS estimator is one such consistent estimator and thus can be used as a valid estimator for usage with the weighted lasso penalty.

4. Algorithms and Experiments

In this paper, we limit ourselves to an empirical study of data block sizes for the RS estimator. We replicate the experiments from ‘An Improved 1-norm SVM for Simultaneous Classification and Variable Selection’ by (Zou, 2007) and report on 1-norm SVM and SVM2.

Method for choosing Weights (for Hybrid and RS) and Validation data (for RS): For the Hybrid SVM, in order to find the optimal weights via \( L_2 \) penalty, we use the method described by (Zou, 2007). We first find the best SVM (or SVM2) algorithm model weights \( (\beta(l_2)) \) with the \( L_2 \) penalty via a parameteric search over costs \( C = \{0.1, 0.5, 1, 2, 5, 10\} \). We then create entire piecewise paths for various weight values \( |\beta(l_2)|^{-\gamma}, \gamma = \{1, 2, 4\} \); choose the best performing model on validation data and then report on the test dataset. Description on how we chose training set for the RS method is given in individual experiments. Our RS experiments need validation data to help choose the relevant features for each of the RS training set. We do the following: if \( n \) is the size of the training set and we choose \( m \) of those examples for the current RS training set, we just use the left out \( n - m \) examples (as validation) for choosing the best features from the piecewise paths generated by the \( L_1 \) algorithm on the \( m \) examples. In case, if a held out validation set was present, we use that instead.

4.1 Synthetic Datasets

We simulate two synthetic datasets, one akin to “orange data” described in (Zhu et al., 2003) and another a bernoulli distribution based dataset. The following notation is used for some of the tables: We use “C” and “IC” to denote the mean number of correctly and incorrectly selected features, respectively. Also, we resort to reporting to mean and std. deviation as the median of the incorrectly selected features was 0 for many experiments. “PPS” stands for the probability of perfect selection, i.e the probability of only choosing the correct feature set.
Models 1 and 4 from (Zou, 2007): The “orange data” has two classes, one inside the other like the core inside the skin of the orange. The first class has two independent standard normals $x_1$ and $x_2$. The second class also has two independent standard normals $x_1$ and $x_2$ but is conditioned on $4.5 \leq x_1^2 + x_2^2 \leq 8$. To simulate the effects of noise, there are ‘$q$’ independent standard normals. The Bayes rule is $1-2I(4.5 \leq x_1^2 + x_2^2 \leq 8)$, where $I()$ is an indicator function and the Bayes error is about 4%. We resort to an enlarged dictionary $D = \{ \sqrt{2}x_j, \sqrt{2}x_jx_k, x_j^2, j, k = 1, 2, \ldots, 2 + q \}$ as the original space is not linear. We have independent sets of 100 validation examples and 20000 test examples and. ‘$q$’ is set to 2, 4, 6, 8, 12, 16, 20 and we report on 500 experiments.

For the RS method, block sizes were set to 20%, 30% and 40% of the total training size and performed $10/(%$size of each block/100) total experiments; i.e. for 20% we generated $10/0.2=20$ total randomized training sets each of size 0.2*(total training data). The weighted vector was created via the RS method described earlier and then used to train the weighted 1-norm and SVM2 algorithms.

Table 3: Variable Selection Results on Models 1 & 4 using SVM2

<table>
<thead>
<tr>
<th>q</th>
<th>p</th>
<th>6</th>
<th>8</th>
<th>12</th>
<th>16</th>
<th>20</th>
<th>275</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-norm SVM2</td>
<td>IC</td>
<td>1.5±2.59</td>
<td>1.42±2.44</td>
<td>1.67±3.4</td>
<td>1.58±2.95</td>
<td>1.71±3.52</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PPS</td>
<td>0.564</td>
<td>0.544</td>
<td>0.536</td>
<td>0.564</td>
<td>0.592</td>
<td></td>
</tr>
<tr>
<td>Hybrid SVM2</td>
<td>IC</td>
<td>1.05±1.87</td>
<td>1.03±1.79</td>
<td>1.19±2.05</td>
<td>1.35±2.51</td>
<td>1.13±2.21</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PPS</td>
<td>0.596</td>
<td>0.598</td>
<td>0.554</td>
<td>0.576</td>
<td>0.596</td>
<td></td>
</tr>
<tr>
<td>RS(20%)</td>
<td>IC</td>
<td>0.65±1.15</td>
<td>0.62±1.15</td>
<td>0.8±1.48</td>
<td>0.61±1.17</td>
<td>0.54±1.04</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PPS</td>
<td>0.636</td>
<td>0.646</td>
<td>0.600</td>
<td>0.686</td>
<td>0.666</td>
<td></td>
</tr>
<tr>
<td>RS(30%)</td>
<td>IC</td>
<td>0.69±1.18</td>
<td>0.73±1.15</td>
<td>0.70±1.27</td>
<td>0.63±1.25</td>
<td>0.56±1.06</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PPS</td>
<td>0.626</td>
<td>0.604</td>
<td>0.626</td>
<td>0.666</td>
<td>0.662</td>
<td></td>
</tr>
<tr>
<td>RS(40%)</td>
<td>IC</td>
<td>0.62±1.05</td>
<td>0.61±1.01</td>
<td>0.68±1.29</td>
<td>0.66±1.25</td>
<td>0.55±1.02</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PPS</td>
<td>0.644</td>
<td>0.636</td>
<td>0.650</td>
<td>0.668</td>
<td>0.672</td>
<td></td>
</tr>
<tr>
<td>RS(50%)</td>
<td>IC</td>
<td>0.67±1.11</td>
<td>0.65±1.19</td>
<td>0.69±1.31</td>
<td>0.62±1.36</td>
<td>0.59±1.14</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PPS</td>
<td>0.628</td>
<td>0.628</td>
<td>0.630</td>
<td>0.670</td>
<td>0.670</td>
<td></td>
</tr>
</tbody>
</table>

*C (mean of Correct features)=2 for all above experiments

We depict error rates in Table 1 & 2 for SVM2 and 1-norm SVM respectively. q depicts the number of noise features in original space and p represents the number of features in the new space via the dictionary $D$. The $L_2$ algorithm version, in the 3rd column, show increasing error rates as the number of noisy features increase. The $L_1$ algorithm version, in the 4th column is much more robust to noise and the error rates do not degrade at all. Hybrid SVM perform usually better than the unweighted 1-norm SVM (except for couple of times for in Table 2). For all different block sizes, the RS method performs best. The feature selecting ability of individual algorithm is depicted in Table 3 (Note: 1-norm SVM results were omitted for space constraints and the results were similar to those of SVM2). We can see that the probability of finding the best model is high for all the algorithms. Hybrid is better at that compared to the 1-norm and the RS method performs best.

Models 2, 3 and 5 from (Zou, 2007): Models 2, 3 and 5 are simulated from the model $y \sim \text{Bernoulli}\{p(u)\}$ where $p(u) = \exp(x^T\beta + \beta_0 + \epsilon)/(1 + \exp(x^T\beta + \beta_0) + \epsilon)$ with $\epsilon$ being...
a standard normal representing error. We create 100 training examples, 100 validation examples, 20,000 test examples and report on 500 randomized experiments.

Model 2 (Sparse Model): We set $\beta_0 = 0$ and $\beta = \{3, 0, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0\}$. The features $x_1, ..., x_{12}$ are standard normals and experiments are done with correlation between $x_i$ and $x_j$ set to $\rho = \{0, 0.5\}$. The Bayes rule is to assign classes to $2I(x_i + x_6 + x_{12}) - 1$.

Model 3 (Sparse Model): We use $\beta_0 = 1$ and $\beta = \{3, 2, 0, 0, 0, 0, 0, 0, 0\}$. The features $x_1, ..., x_{12}$ are standard normals and experiments are done with correlation between $x_i$ and $x_j$ set to $\rho = \{0, 0.5\}$. The Bayes rule is to assign classes to $2I(3x_1 + 2x_2 + 1) - 1$.

Model 5 (Noisy features): We use $\beta_0 = 1$ and $\beta = \{3, 2, 0, 0, 0, 0, 0, 0, 0\}$. The features $x_1, ..., x_{12}$ are standard normals and experiments are done with correlation set to $\rho = 0.5^{|i-j|}$. We added 300 independent normal variables as noise features to get a total of 309 features.

### Table 4: Mean Error rates in % for Models 2, 3 & 5 using SVM2

<table>
<thead>
<tr>
<th>Exp. Name</th>
<th>Correlation</th>
<th>Bayes 2-norm</th>
<th>1-norm</th>
<th>Hybrid</th>
<th>RS(20%)</th>
<th>RS(30%)</th>
<th>RS(40%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model 2</td>
<td>$\rho = 0$</td>
<td>6.04</td>
<td>9.77</td>
<td>8.14</td>
<td>7.46</td>
<td>7.51</td>
<td>7.55</td>
</tr>
<tr>
<td></td>
<td>$\rho = 0.5$</td>
<td>4.35</td>
<td>7.74</td>
<td>6.43</td>
<td>5.96</td>
<td>5.97</td>
<td>5.86</td>
</tr>
<tr>
<td>Model 3</td>
<td>$\rho = 0$</td>
<td>8.48</td>
<td>11.04</td>
<td>9.79</td>
<td>9.54</td>
<td>9.46</td>
<td>9.45</td>
</tr>
<tr>
<td></td>
<td>$\rho = 0.5$</td>
<td>7.03</td>
<td>8.49</td>
<td>9.51</td>
<td>8.45</td>
<td>8.17</td>
<td>8.17</td>
</tr>
<tr>
<td>Model 5</td>
<td>$\rho = 0.5^{</td>
<td>i-j</td>
<td>}$</td>
<td>6.88</td>
<td>31.31</td>
<td>9.32</td>
<td>8.5</td>
</tr>
</tbody>
</table>

*range of std. deviation in accuracy for above table was between 1.02 to 1.96.*

For Models 2, 3 and 5: error rates are reported in Table-4 for SVM2 (results for 1-norm SVM were similar and hence skipped). Note, weighted models are consistently better than both of their 1-norm and 2-norm unweighted counterparts. The RS method has equal or greater accuracy than the Hybrid version.

### 4.2 Real World Datasets

**UCI datasets:** In Table 5, results on the Spam, WDBC and Ionosphere datasets from UCI repository, by (Asuncion and Newman, 2007), are reported. For WDBC and Ionosphere dataset, we split the data into 3 parts with 2 parts used for training (and validation) and the 3rd remaining part for testing. For the Spam dataset, indicators for test (1536 examples) set and training set can be obtained from [http://www-stat.stanford.edu/~tibs/ElemStatLearn/](http://www-stat.stanford.edu/~tibs/ElemStatLearn/). For our RS method we generated smaller datasets from the training set as follows: If the training set size is $N$ and size for individual RS set is $K$, then the number of datasets generated are $10 \times N/K$. We also show the size of the RS training set as Block in the table. For Hybrid SVM, the best parameter for $\gamma$ and $C$ are chosen as described earlier in Section 4. We report on 50 randomized experiments. In Table 5, error rates for both SVM2 and 1-norm SVM are shown. The use of weights via Hybrid and RS method, always increases the accuracy from the unweighted case. Also, as seen on both synthetic and real world datasets, RS blocksize does not create that much variability in the results.

**Robotic Dataset:** We now discuss a novel use of our subsampling method on robotic datasets (Procopio, 2007). These datasets are created by hand labeling 100 images obtained from running the DARPA LAGR robot in varied outdoor environments. The classes labeled are robot traverseable path and obstacles. The authors provide pre-extracted color histogram features for the dataset at (Procopio et al., 2009). We used a subset (12,000
Increasing Feature Selection Accuracy for $L_1$ Regularized Linear Models

Table 5: Mean ± Std. Deviation of Error Rates on Real world Datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Algorithm (Name/Block)</th>
<th>Without Weighting</th>
<th>Randomized Sampling Weighting</th>
<th>Hybrid SVM</th>
<th>2-norm SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>WDBC</td>
<td>1-norm (100)</td>
<td>3.66 ± 1.17</td>
<td>2.79 ± 0.93</td>
<td>2.89 ± 0.79</td>
<td>4.05 ± 1.36</td>
</tr>
<tr>
<td></td>
<td>1-norm (150)</td>
<td></td>
<td>2.79 ± 0.90</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SVM2 (100)</td>
<td>3.55 ± 1.81</td>
<td>2.78 ± 1.03</td>
<td>2.73 ± 1.01</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SVM2 (150)</td>
<td></td>
<td>2.90 ± 1.15</td>
<td>2.71 ± 1.13</td>
<td></td>
</tr>
<tr>
<td>SPAM</td>
<td>1-norm (200)</td>
<td>9.09 ± 0.878</td>
<td>8.18 ± 0.49</td>
<td>8.31 ± 0.61</td>
<td>7.06 ± 0.04</td>
</tr>
<tr>
<td></td>
<td>1-norm (1000)</td>
<td></td>
<td>7.53 ± 0.17</td>
<td>8.19 ± 0.73</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SVM2 (200)</td>
<td>8.45 ± 3.43</td>
<td>7.38 ± 0.52</td>
<td>7.39 ± 0.30</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SVM2 (1000)</td>
<td></td>
<td>7.70 ± 2.73</td>
<td>7.48 ± 0.52</td>
<td></td>
</tr>
<tr>
<td>Ionosphere</td>
<td>1-norm (50)</td>
<td>12.38 ± 2.04</td>
<td>11.52 ± 1.39</td>
<td>11.84 ± 1.38</td>
<td>13.03 ± 2.86</td>
</tr>
<tr>
<td></td>
<td>1-norm (75)</td>
<td></td>
<td>11.25 ± 1.98</td>
<td>11.56 ± 1.73</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1-norm (100)</td>
<td></td>
<td>11.29 ± 1.65</td>
<td>11.72 ± 1.23</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SVM2 (50)</td>
<td>12.69 ± 2.82</td>
<td>11.43 ± 2.52</td>
<td>11.21 ± 2.66</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SVM2 (75)</td>
<td></td>
<td>11.61 ± 2.50</td>
<td>11.22 ± 2.58</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SVM2 (100)</td>
<td></td>
<td>11.37 ± 2.67</td>
<td>11.26 ± 2.68</td>
<td></td>
</tr>
</tbody>
</table>

Table 6: Avg. Error rate on Robotic Datasets from (Procopio, 2007)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>DS1A</th>
<th>DS2A</th>
<th>DS3A</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unweighted SVM2</td>
<td>8.92</td>
<td>4.36</td>
<td>1.24</td>
</tr>
<tr>
<td>Weighted SVM2</td>
<td>6.41</td>
<td>4.13</td>
<td>1.15</td>
</tr>
</tbody>
</table>

examples) of the available data for each of the 100 frames. Each example is 15 dimensional. We set our experimentation as follows: for each frame $F_i$, $i$ is the index of the frame, we divide the obtained examples (12K examples) into 8 folds (9.6K examples) for training, 1 fold (1.2K examples) for validation and 1 fold (1.2K examples) for testing. We train/validate/test on the unweighted SVM2 algorithm. For the weighted experiment, we train via our RS method, by dividing the training into 10 subsets (each 960 examples) and finding the weight vector. This weight vector is then used to create the weighted SVM2 models and we report on the test set. Now, instead of discarding weights when a new frame arrives, we use the weights found in frame $F_i$ again in $F_{i+1}$, i.e. if weights in frame $F_i$ are noted as $W_i$ then:

$$W_{i+1} \leftarrow \{W_i + \text{weight results of RS for frame } F_{i+1}\}$$

This is one experimental environment, where creating $L_2$ models for the entire data is not feasible and the RS estimator is a potential approach. Also, propagating feature importance between frames is an advantage for the RS estimator. In Table-6, we show overall results for 100 frames for 3 datasets done 10 times. We propagate the weights for the weighted SVM2 between frames. As shown, there is a drop in error rates (between 5-28%) for the weighted SVM2 compared to the unweighted SVM2. The overhead of computing the weights via RS was < 10% that of computing a model for the entire training set.

5. Conclusions and Future work

A Random Sampling framework is presented and is empirically shown to give effective feature weights to the lasso penalty, resulting in both increased model accuracy and feature selection accuracy. The proposed framework is at least as effective (and at times more
effective than) the Hybrid SVM, with the added benefit of significantly lower computational cost. In addition, unlike the Hybrid SVM which must see all the data at once, Random Sampling is shown to be effective in an on-line setting where predictions must be made based on only partially available data (as in data taken from the robotics domain). In this paper the framework is demonstrated on two types of linear classification algorithms, and theoretical support is presented showing its applicability, in general, to sparse algorithms.

References


Learning Dissimilarities for Categorical Symbols

Jierui Xie  
Boleslaw Szymanski  
Mohammed J. Zaki  

Department of Computer Science  
Rensselaer Polytechnic Institute  
Troy, NY 12180, USA  

Editor: Huan Liu, Hiroshi Motoda, Rudy Setiono, and Zheng Zhao  

Abstract  

In this paper we learn a dissimilarity measure for categorical data, for effective classification of the data points. Each categorical feature (with values taken from a finite set of symbols) is mapped onto a continuous feature whose values are real numbers. Guided by the classification error based on a nearest neighbor based technique, we repeatedly update the assignment of categorical symbols to real numbers to minimize this error. Intuitively, the algorithm pushes together points with the same class label, while enlarging the distances to points labeled differently. Our experiments show that 1) the learned dissimilarities improve classification accuracy by using the affinities of categorical symbols; 2) they outperform dissimilarities produced by previous data-driven methods; 3) our enhanced nearest neighbor classifier (called LD) based on the new space is competitive compared with classifiers such as decision trees, RBF neural networks, Naïve Bayes and support vector machines, on a range of categorical datasets.  

Keywords: Dissimilarity, Categorical Data, Learning Algorithm, Classification, Feature Selection  

1. Introduction  

The notion of distance plays an important role in many data mining tasks, such as classification, clustering, and outlier detection. However, the notion of distance for categorical data is rarely defined precisely, if at all. By categorical or symbolic data, we refer to values that are nominal (e.g. colors) or ordinal (e.g. rating, typically imposed subjectively by a human). In many cases, the dissimilarities between symbols are fuzzy and often arbitrary. An example could be the rating of a movie, chosen from the list “very bad, bad, fair, good, very good”. It is hard to determine how much one symbol differs from another. In this paper, we introduce a new method to derive dissimilarities between categorical symbols in such a way that the power of distance-based data mining methods can be applied.  

The notations used throughout the paper are as follows. There is a dataset $X = \{x_1, x_2, \ldots, x_t\}$ of $t$ data points, where each point $x_i$ is a tuple of $m$ attributes values, $x_i = (x_{i1}, \ldots, x_{im})$. Each of the $m$ attributes $A_i$ is categorical, i.e., the attribute values for $A_i$ are drawn from a set of $n_i$ discrete values given as $\{a_i^1, a_i^2, \ldots, a_i^{n_i}\}$, which also constitute the domain of $A_i$. We assume that all symbols across all attributes are unique. For simplicity, we use the notation $A_i$ to refer to the $i$-th attribute, as well as the domain of...
that attribute. Each $a^j_i$ is also called a symbol. Each data point $x_i$ (in the training set) also has associated with the “true” class label, given as $L(x_i)$. In this paper we only consider the case where there are two classes, i.e., $L(x_i) \in \{1, 2\}$, where 1 and 2 are the two class labels.

The similarity between symbols $a^k_i$ and $a^l_i$ of an attribute $A^i$ is denoted as $S(a^k_i, a^l_i)$, whereas the dissimilarity or distance between two symbols is denoted as $D(a^k_i, a^l_i)$. Typically $S(a^k_i, a^l_i) : A^i \times A^i \rightarrow (0, 1)$, in which case $D(a^k_i, a^l_i) = 1 - S(a^k_i, a^l_i)$. In other cases $S(a^k_i, a^l_i) : A^i \times A^i \rightarrow \mathbb{R}^+$, in which case $D(a^k_i, a^l_i) = \frac{1}{S(a^k_i, a^l_i)}$. The distance between two data points $x_i$ and $x_j$ is defined in terms of the distance between symbols, as follows:

$$D(x_i, x_j) = \sqrt{\sum_{k=1}^{m} D(x^k_i, x^k_j)^2}$$

(1)

Given a point $x_i$, the error of a classifier on that point is defined as:

$$e_{x_i} = \frac{(L(x_i) - O(x_i))^2}{2}$$

(2)

where $O(x_i)$ is the output class of the classifier on point $x_i$. Since $O(x_i) \in \{1, 2\}$, $e_{x_i} \in \{0, \frac{1}{2}\}$. The total error rate of the classifier on a set of $t$ points is simply $E = \sum_{i=1}^{t} e_{x_i}$.

In this paper, our goal is to learn a mapping function from each categorical attribute $A^i$ onto the real number interval, given by the function $r : A^i \rightarrow \mathbb{R}$, with the aim of minimizing the total error rate $E$. Once $r$ has been learned, each categorical data point $x_i$ can be treated as a $m$-dimensional point or vector in $\mathbb{R}^m$, given as $r(x_i) = (r(x_{i,1}), \ldots, r(x_{i,m}))^T$. This enables one to apply any of the distance-based classification methods directly on the transformed dataset $r(X) = \{r(x_i)\}_{i=1}^{t}$.

![Figure 1](image-url) (a) Mapping symbols to real values. Dataset consists of three points: $x_1 = (A, a)$, $x_2 = (B, b)$ and $x_3 = (C, c)$. The mapping function $r$ is given as: $r(A) = 0.1$, $r(B) = 0.2$, $r(C) = 0.6$, $r(a) = 0.5$, $r(b) = 0.4$ and $r(c) = 0.1$. (b) Random mapping. (c) Linearly separable mapping.

As an example, consider Figure 1 (a), which shows three points over two categorical attributes “color” (with symbols A, B and C) and “shape” (with symbols a, b and c). In
the new continuous space, a value assignment to a symbol naturally defines a hyperplane that contains all the points in the dataset having that particular symbol. In this example, each point is defined by exactly two straight lines. Figure 1 (b) shows an extended example with 8 points, with a random initial mapping, which does not discriminate too well between the two classes. Our goal is to improve the initial mapping into a classification-aware mapping like that in Figure 1 (c), which achieves a low classification error rate (on the training set, and hopefully on the test set too).

2. Related Work

The most widely used measure on categorical data is simple matching (or overlap), which is defined as $S(a_i, a_j) = 1$ if $a_i = a_j$ and $S(a_i, a_j) = 0$, otherwise. This measure simply checks that two symbols are the same, which forms the basis for various distance functions, such as Hamming and Jaccard distance (Liang, 2004).

The simple matching ignores information from the dataset and the desired classification. Therefore, many more data-driven measures have been developed to capture preferences for matching or mismatching based on symbols' statistics. Here, we divide related methods into two categories: unsupervised and supervised methods. The unsupervised methods are typically based on frequency or entropy. Let $f(a_i)$ be the frequency of symbol $a_i$ of attribute $A$ in the dataset, then $p(a_i) = f(a_i)/t$.

Let $a_i$ and $a_j$ be two symbols in the domain of attribute $A$. Lin (1998) defines $S(a_i, a_j) = 2 \log p(a_i)$ if $a_i = a_j$, and $2 \log (p(a_i) + p(a_j))$, otherwise, which gives more weight to matches on frequent values and lower weight to mismatches on infrequent values. Burnaby (1970) defines $S(a_i, a_j) = 1$ if $a_i = a_j$. However, if $a_i \neq a_j$, then

$$S(a_i, a_j) = \frac{\sum_{a_k \in A} 2 \log (1 - p(a_k))}{\log \left( \frac{p(a_i)p(a_j)}{(1 - p(a_i))(1 - p(a_j))} \right) + \sum_{a_k \in A} 2 \log (1 - p(a_k))}$$

Smirnov (1968) not only considers the frequency, but also takes the distribution of the other attributes values into account, defining

$$S(a_i, a_j) = 2 + \frac{t - f(a_i)}{f(a_i)} + \sum_{a_k \in A \setminus \{a_i\}} \frac{f(a_k)}{t - f(a_k)}$$

if $a_i = a_j$, and

$$S(a_i, a_j) = \sum_{a_k \in A \setminus \{a_i, a_j\}} \frac{f(a_k)}{n - f(a_k)}$$

otherwise. Goodall (1966) proposed another statistical approach, in which less frequent attribute values make greater contribution to the overall similarity than frequent attribute values. A modified version called Goodall1 is proposed in (Boriah et al., 2008), defining

$$S(a_i, a_j) = 1 - \sum_{a_k \in A \setminus A_i, p(a_k) < p(a_i)} p^2(a_k)$$

if $a_i = a_j$, and 0 otherwise. Gambaryan (1964) proposed a measure related to information entropy, which gives more weight to matches where the number of matches is between
frequent and rare. If \( a_i = a_j \), the similarity is given as
\[
S(a_i, a_j) = -[p(a_i) \log_2 p(a_i) + (1 - p(a_i)) \log_2 (1 - p(a_i))] 
\]
and 0 otherwise. Eskin et al. (2002) consider the number of symbols of each attribute. In its modified version (Boriah et al., 2008), this measure gives more weight to mismatches that occur on an attribute with more symbols using the weight \( \frac{n^2}{n^2 + 2} \), where \( n \) is the number of symbols of attribute \( A \). Occurrence Frequency (OF) (Jones, 1988) gives lower similarity to mismatches on less frequent symbols and higher similarity on mismatches on more frequent symbols. Conversely, Inverse Occurrence Frequency (IOF) assigns higher similarity to mismatches on less frequent symbols. That is, if \( a_i \neq a_j \), then
\[
S(a_i, a_j) = \frac{1}{1 + \log(f(a_i)) \log(f(a_j))} 
\]
for OF, and
\[
S(a_i, a_j) = \frac{1}{1 + \log(f(a_i)) \log(f(a_j))} 
\]
for IOF. When \( a_i = a_j \), both define \( S(a_i, a_j) = 1 \). More discussion on these kinds of measures is given by Boriah et al. (2008).

The supervised methods take advantage of the class information. An example is Value Difference Metric (VDM) proposed in (Stanfill and Waltz, 1986). The main idea is that symbols are similar if they occur with a similar relative frequency for all the classes. The dissimilarity between \( a_i \) and \( a_j \) is defined as a sum over \( n \) classes:
\[
D(a_i, a_j) = \sum_{c=1}^{n} \left| \frac{C_{a_i,c}}{C_{a_i}} - \frac{C_{a_j,c}}{C_{a_j}} \right|^h 
\]
where \( C_{a_i,c} \) is the number of times symbol \( a_i \) occurs in class \( c \). \( C_{a_i} \) is the total number of times \( a_i \) occurs in the whole dataset. Constant \( h \) is usually set to 1. Cheng et al. (2004) proposed an approach based on Hadamard product and RBF classifier. They attempt to evaluate all the pair-wise distances between symbols, and they optimize the error function using gradient descent method. In our algorithm the number of values to be estimated is equal to the number of symbols across all attributes, i.e. linear in the symbol set size, which may enable faster and more robust learning. However, we were unable to compare the methods directly since we did not have access to the code from Cheng et al. (2004). Furthermore, in our approach, after learning, all pair-wise distances can be easily derived if needed.

3. Learning Algorithm

Our learning algorithm is based on the gradient descent method. Starting from an initial assignment of real values to the symbols, guided by the error rate based on a nearest neighbor classifier, our method iteratively updates the assignments. Intuitively, in each iteration, the method moves the symbols (hence the lines or, more generally, the hyperplanes, as seen in Figure 1) to new locations according to the net force imposed on them. Let \( x \) be the
closest point to \( p \) from class 1, and \( y \) the closest point from class 2. Let \( d_1 = D(p, x) \), and \( d_2 = D(p, y) \) be the corresponding distances, and \( \Delta d = d_1 - d_2 \) be the difference of the distances. Our simple nearest neighbor classifier assigns the class as follows:

\[
O(p) = S(\Delta d) + 1 = S(d_1 - d_2) + 1
\]  

(3)

where \( S(x) = \frac{1}{1+e^{-x}} \) is the sigmoid function. It is easy to verify that if \( d_1 \ll d_2 \), then \( O(p) \approx 1 \) and if \( d_1 \gg d_2 \) then \( O(p) \approx 2 \). The classification error for \( p \) is \( e_p \) as given in equation (2). We update the assignment of values to symbols depending on the error \( e_p \), as discussed below. Our method in fact cycles through all points, considering each as the target, in each iteration. In batch training, the total assignment is accumulated over all the points, but in online training, the assignment is updated immediately after each point. The pseudo code of the algorithm (with batch training) is given below.

**LD: Learning Algorithm (Dataset X):**

\[
t = \text{Number of instances in the dataset};
\]

\[
r = \text{Random initial assignment};
\]

\[
\text{while} (\text{stop criteria not satisfied}) \{
\]

\[
\text{sum} \Delta r = 0;
\]

\[
\text{for} \ k = 1:t \{
\]

\[
p = x_k \text{ or any point taken at random from } X;
\]

\[
d_1 = \min_{x_k \in X, L(x_k)=1, x_k \neq p} \{D(p, x_k)\};
\]

\[
d_2 = \min_{x_k \in X, L(x_k)=2, x_k \neq p} \{D(p, x_k)\};
\]

\[
\Delta d = d_1 - d_2;
\]

\[
\text{Compute } \Delta r \text{ using equation (8)};
\]

\[
\text{sum} \Delta r = \text{sum} \Delta r + \Delta r;
\]

\[
\}
\]

\[
\text{update } r = r + \text{sum} \Delta r;
\]

\[
\}
\]

### 3.1 Objective Function and Update Equation

The general update equation for a target point \( p \) is given as \( r = r + \Delta r \). Each element in \( r \), \( r_j \), represents the real value assignment for the \( j \)-th symbol of the \( i \)-th attribute. Thus, for each \( r_j \), the update equation is \( r_j = r_j + \Delta r_j \). \( r_j \) moves in the direction of the negative gradient of \( e_p \) to decrease the error. That is,

\[
\Delta r_j = -\eta \cdot \frac{\partial e_p}{\partial a_j} = -\eta \cdot \frac{\partial (L(p) - O(p))^2/2}{\partial a_j} = \eta \cdot (L(p) - O(p)) \cdot \frac{\partial O(p)}{\partial a_j}
\]  

(4)

where \( \eta \) is the learning rate, and the differential is taken with respect to the \( j \)-th symbol for attribute \( A^i, a^i_j \).

Note that, by equation (3),

\[
\frac{\partial O(p)}{\partial a_j} = \frac{\partial [S(d_1 - d_2) + 1]}{\partial a_j} = \frac{\partial S(\Delta d)}{\partial \Delta d} \cdot \frac{\partial \Delta d}{\partial a_j} = S(\Delta d) \cdot (1 - S(\Delta d)) \cdot \frac{\partial \Delta d}{\partial a_j}
\]  

(5)

The last step follows from the fact that the partial derivative of the sigmoid function \( S(x) \) is given as: \( \frac{\partial S(x)}{\partial x} = S(x)(1 - S(x)) \).
3.2 Computing the Derivative of $\Delta d$

Note that $\frac{\partial \Delta d}{\partial a_{ij}} = \frac{\partial d_1}{\partial a_{ij}} - \frac{\partial d_2}{\partial a_{ij}}$, where $d_1 = \sqrt{\sum_{i=1}^{m} D(p_i, x_i)^2}$ and $d_2 = \sqrt{\sum_{i=1}^{m} D(p_i, y_i)^2}$ are the distances from $p$ to the closest points $x$ in class 1, and $y$ in class 2, respectively. Since the derivative is with respect to the $j$-th attribute in attribute $i$, even in the distance terms $d_1$ and $d_2$, only the $i$-th attribute has to be considered. Let us consider $d_1$, we have:

$$\frac{\partial d_1}{\partial a_{ij}} = \frac{\partial}{\partial a_{ij}} \left( \frac{\sum_{i=1}^{m} D(p_i, x_i)^2}{2} \right)^{1/2} = \frac{1}{2} \cdot (d_1)^{-1/2} \cdot \frac{\partial D(p_i, x_i)^2}{\partial a_{ij}}$$

(6)

The derivative will be zero if the symbol for the $i$-th attribute is not $a_{ij}$, as per the following:

$$\frac{\partial D(p_i, x_i)^2}{\partial a_{ij}} = 2 \cdot D(p_i, x_i) \cdot \left\{ \begin{array}{ll} +1, & \text{if } p_i = a_{ij} \text{ and } x_i \neq a_{ij} \\ -1, & \text{if } p_i \neq a_{ij} \text{ and } x_i = a_{ij} \\ 0, & \text{otherwise} \end{array} \right. \quad (7)$$

In a similar manner we can derive $\frac{\partial d_2}{\partial a_{ij}}$.

By putting the above equations together we have a full version of equation (4) as follows:

$$\Delta r_{ij} = \eta \cdot (L(p) - O(p)) \cdot S(d_1 - d_2) \cdot (1 - S(d_1 - d_2)) \cdot \left( \frac{\partial d_1}{\partial a_{ij}} - \frac{\partial d_2}{\partial a_{ij}} \right)$$

(8)

Figure 2: (a) Schematic of forces acting on points $p$, $x$ and $y$. Forces along each axis are in solid arrow, and net force in dashed arrow. During the learning process, $p$ is getting closer to $x$ and farther from $y$. (b) Example 2-d synthetic data, with two features, each of which has twenty symbols. Red circles and blue stars indicate points from two classes. (c) Subspaces corresponding to the learned mapping.

3.3 Example: Line Moving and Subspace Forming

Given a target point $p$ and the corresponding closest points $x$ and $y$, in class 1 and 2, the amount of assignment change of each symbol is proportional to $\Delta r_{ij}$, which moves the location of a symbol to the left or to the right on the $i$-th axis (if $\Delta r_{ij} < 0$ or $\Delta r_{ij} > 0$).
respectively). In a 2-dimensional space, the change of symbol assignments is equivalent to moving the lines around, which in turn “move” the data points to new locations.

Figure 2 (a) illustrates a case where \( \mathbf{p} = (A_1, B_1) \) and \( \mathbf{x} = (A_3, B_3) \) belong to class 1 but \( \mathbf{p} \) is misclassified, since it is closer to \( \mathbf{y} = (A_2, B_2) \) in class 2 (i.e., \( d_1 > d_2 \)). In this specific case, there are six symbols, \( A_1, A_2, A_3, B_1, B_2, \) and \( B_3 \). Intuitively, when the learning goes on, more and more points nearby tend to get together and form a subspace containing points with the same class label. Subspaces with different class labels tend to be apart. To demonstrate how the subspaces are created, we applied our learning algorithm on an example 2-d synthetic datasets in Figure 2 (b). The learned subspace are shown in (c).

4. Discovering Symbol Redundancy

By modeling each symbol as a variable in the real space, our algorithm explores the distances between symbols on each individual feature. Interestingly, our algorithm is able to discover redundancies among symbols, which provides insights for improving the classification performance.

We ran our learning approach on the Balance Scale dataset from UCI repository (see Table 3). Table 1 shows a typical assignment learned on this dataset. As highlighted, some symbols have very close values (e.g., symbols ‘2’=1.76 and ‘3’=1.96 for attribute left-weight; symbols ‘3’=3.75 and ‘4’=3.47 for attribute left-distance, and so on). Such closeness implies that for an attribute like left-weight having five symbols may not be necessary for classification. We regard this kind of closeness as redundant information, which should be removed to improve the classification accuracy. To verify the above hypothesis, we

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Symbol</th>
<th>‘1’</th>
<th>‘2’</th>
<th>‘3’</th>
<th>‘4’</th>
<th>‘5’</th>
</tr>
</thead>
<tbody>
<tr>
<td>left-weight</td>
<td>-0.85</td>
<td>1.76</td>
<td>1.96</td>
<td>4.98</td>
<td>7.14</td>
<td></td>
</tr>
<tr>
<td>left-distance</td>
<td>-0.72</td>
<td>1.78</td>
<td>3.75</td>
<td>3.47</td>
<td>6.71</td>
<td></td>
</tr>
<tr>
<td>right-weight</td>
<td>-0.78</td>
<td>2.22</td>
<td>3.19</td>
<td>5.08</td>
<td>5.28</td>
<td></td>
</tr>
<tr>
<td>right-distance</td>
<td>-1.04</td>
<td>1.10</td>
<td>3.68</td>
<td>5.49</td>
<td>5.76</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: A typical assignment learned on Balance Scale dataset

<table>
<thead>
<tr>
<th>Original Dataset</th>
<th>Merged Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>C4.5</td>
<td>76.35</td>
</tr>
<tr>
<td>RBFNN</td>
<td>92.20</td>
</tr>
<tr>
<td>NN</td>
<td>85.93</td>
</tr>
</tbody>
</table>

Table 2: Accuracy improvement on merged dataset

merged the two closest symbols into one. For example, we replaced symbols ‘2’ and ‘3’ with only one symbol ‘23’. As shown in Table 2, the classification accuracy is improved with the merged attributes for decision tree, RBF neural network and nearest neighbor classifier (i.e., NN, based on Overlap dissimilarity measure). The merging can be considered as a form of pre-pruning process, which improves the generality of classifiers.
5. Experimental Results

To evaluate the learned dissimilarity measure (short for $LD$), we compare our approach against other data-driven methods discussed in Section 2 and other popular classifiers. We present results on categorical datasets shown in Table 3, that are all taken from the UCI machine learning repository. The number of attributes ranges from 4 to 60, and each attribute takes on 2 to 12 symbols.

Table 3: Dataset Information

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Size</th>
<th>Dimension</th>
<th>Attributes and Symbols</th>
</tr>
</thead>
<tbody>
<tr>
<td>Splice</td>
<td>1330</td>
<td>60</td>
<td>Each dimension takes on ${A,C,T,G}$</td>
</tr>
<tr>
<td>Balance Scale</td>
<td>576</td>
<td>4</td>
<td>Each dimension takes on ${1, 2, 3, 4, 5}$</td>
</tr>
<tr>
<td>Car Evaluation</td>
<td>768</td>
<td>6</td>
<td>buying: ${v\text{-high}, \text{high, med, low}}$; maint: ${v\text{-high}, \text{high, med, low}}$; doors: ${2, 3, 4, 5+}$; persons: ${2, 4, \text{more}}$; lug boot: ${\text{small, med, big}}$; safety: ${\text{low, med, high}}$</td>
</tr>
<tr>
<td>Connect-4</td>
<td>1000</td>
<td>42</td>
<td>Each dimension takes on ${x, o, b}$</td>
</tr>
<tr>
<td>Mushroom</td>
<td>1000</td>
<td>22</td>
<td>Various sizes from 2 to 12, e.g. the first attributes cap shape: ${\text{bell, conical, convex, flat, knobbled, sunken}}$</td>
</tr>
<tr>
<td>Tic-tac-toe</td>
<td>624</td>
<td>9</td>
<td>Each dimension takes on ${x, o, b}$</td>
</tr>
<tr>
<td>Hayes-Roth</td>
<td>100</td>
<td>4</td>
<td>hobby: ${1,2,3}$; age: ${1,2,3}$; educational level: ${1,2,3}$; marital status: ${1,2,3}$</td>
</tr>
</tbody>
</table>

5.1 Comparison with Various Data-Driven Methods

To compare our Learned Dissimilarity approach, with those learned from other ten methods mentioned in Section 2, we evaluate the classification accuracy of the nearest neighbor classifier, where the distances are computed from various dissimilarity measures. More specifically, the distance between two categorical points is calculated according to equation (1). We used 5-fold cross-validation to measure the classification accuracy. The numbers reported in Table 4, correspond to the average classification accuracy and standard deviation (in parenthesis) over ten runs (i.e., we repeat NN ten times for each dissimilarity measure on each dataset). The last row of Table 4 shows the average performance over all the datasets. The highest accuracy is shown in bold for each dataset.

On average, the $LD$ and VDM achieve the best accuracy, indicating that supervised dissimilarities attain better results over the unsupervised counterparts. Among the unsupervised measures, IOF, Lin are slightly superior to others. Goodall1, Smirnov and OF achieve same performance as Overlap. By considering the confidence interval (accuracy +/- standard deviation) to compare the performance of different methods on each dataset, we conclude that $LD$ performed statistically worse than Lin on datasets Splice and Tic-tac-toe but better than Lin on datasets Connection-4, Hayes and Balance Scale. Moreover, $LD$ performed statistically worse than VDM only on one dataset (Splice) but better on two datasets (Connection-4 and Tic-tac-toe). Finally, $LD$ performed statistically at least as well as (and on some datasets, e.g. Connection-4, better than) the remaining methods.
Table 4: Performance comparison on various dissimilarities

<table>
<thead>
<tr>
<th></th>
<th>Overlap</th>
<th>Lin</th>
<th>Smirnov</th>
<th>Goodall1</th>
<th>Eskin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Splice</td>
<td>89.45(0.58)</td>
<td>94.21(0.42)</td>
<td>88.53(0.82)</td>
<td>88.79(0.69)</td>
<td>88.42(0.60)</td>
</tr>
<tr>
<td>Balance Scale</td>
<td>75.31(1.44)</td>
<td>75.52(3.31)</td>
<td>74.65(2.14)</td>
<td>75.69(2.04)</td>
<td>64.23(0.51)</td>
</tr>
<tr>
<td>Car Evaluation</td>
<td>87.86(1.23)</td>
<td>92.64(1.65)</td>
<td>83.72(2.14)</td>
<td>84.96(2.26)</td>
<td>86.65(0.75)</td>
</tr>
<tr>
<td>Connect-4</td>
<td>84.20(0.92)</td>
<td>78.25(1.05)</td>
<td>75.30(0.73)</td>
<td>82.50(0.46)</td>
<td>83.55(0.30)</td>
</tr>
<tr>
<td>Mushroom</td>
<td>100(0)</td>
<td>100(0)</td>
<td>99.75(0.17)</td>
<td>99.90(0.06)</td>
<td>100(0)</td>
</tr>
<tr>
<td>Tie-tac-toe</td>
<td>81.59(1.56)</td>
<td><strong>98.64</strong>(0.69)</td>
<td>84.03(1.07)</td>
<td>86.97(1.58)</td>
<td>63.85(0.93)</td>
</tr>
<tr>
<td>Hayes-Roth</td>
<td>70.90(4.99)</td>
<td>71.00(3.65)</td>
<td>71.00(5.50)</td>
<td>69.50(5.01)</td>
<td>67.00(5.76)</td>
</tr>
<tr>
<td>Average</td>
<td>84.18(1.53)</td>
<td>87.18(1.53)</td>
<td>82.42(1.79)</td>
<td>84.04(1.72)</td>
<td>79.10(1.26)</td>
</tr>
</tbody>
</table>

5.2 Comparison with Various Classifiers

We consider the NN based on our learned dissimilarity as an “enhanced” nearest neighbor classifier, again denoted as LD. The performance of LD is compared with algorithms implemented in Weka 3.6, including decision tree (C4.5 with pruning), Naïve Bayes (NB), RBF neural network (RBFNN, with clustering technique to estimate the number of kernels), and SVM (with RBF kernel and complexity 1.0). Our method uses the learned mapping \( r \), whereas the other methods use the Euclidean distance (corresponding to simple matching) between categorical points. The performance metric is the average classification accuracy over ten runs based on 5-fold cross validation. As shown in Table 5, considering the same confidence intervals as in Sec.5.1, we conclude that LD performed statistically worse than the other methods on only one dataset (Splice) but performed better on at least three other datasets than each of the other methods, which we believe shows a significant improvement over them.

6. Conclusions

In this paper, we propose a task-oriented or supervised iterative learning approach to learn a distance function for categorical data. The algorithm explores the relationships between categorical symbols by utilizing the classification error as guidance. We show that the real value mappings found by our algorithm provide discriminative information, which can be used to refine features and improve classification accuracy. In the future work, we would like to extend the approach to continuous and mixed attribute datasets, as well as “relational” datasets where there are links between data points.
Table 5: Performance comparison on various classifiers

<table>
<thead>
<tr>
<th>Dataset</th>
<th>C4.5</th>
<th>NB</th>
<th>RBFNN</th>
<th>SVM</th>
<th>LD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Splice</td>
<td>95.03(1.00)</td>
<td>97.01(0.42)</td>
<td>97.01(0.64)</td>
<td>96.91(0.60)</td>
<td>93.00(0.67)</td>
</tr>
<tr>
<td>Balance Scale</td>
<td>73.11(1.78)</td>
<td>96.34(1.53)</td>
<td>90.68(1.50)</td>
<td>95.49(1.78)</td>
<td>94.04(1.21)</td>
</tr>
<tr>
<td>Car Evaluation</td>
<td>96.51(1.12)</td>
<td>92.32(2.33)</td>
<td>93.58(2.01)</td>
<td>88.24(1.88)</td>
<td>98.00(1.47)</td>
</tr>
<tr>
<td>Connect-4</td>
<td>87.01(1.71)</td>
<td>87.53(1.10)</td>
<td>88.35(1.62)</td>
<td>87.48(0.89)</td>
<td>87.48(0.92)</td>
</tr>
<tr>
<td>Mushroom</td>
<td>100(0.00)</td>
<td>97.33(1.00)</td>
<td>100(0.00)</td>
<td>100(0.00)</td>
<td>100(0.00)</td>
</tr>
<tr>
<td>Tic-tac-toe</td>
<td>85.44(3.26)</td>
<td>76.67(1.94)</td>
<td>80.75(2.53)</td>
<td>77.21(1.27)</td>
<td>95.30(1.72)</td>
</tr>
<tr>
<td>Hayes-Roth</td>
<td>71.00(8.07)</td>
<td>68.50(9.15)</td>
<td>72.40(5.17)</td>
<td>64.10(12.42)</td>
<td>79.40(1.71)</td>
</tr>
<tr>
<td>Average</td>
<td>86.87(2.42)</td>
<td>87.96(2.49)</td>
<td>88.97(1.93)</td>
<td>87.06(2.69)</td>
<td>92.46(1.10)</td>
</tr>
</tbody>
</table>

References


