Discovering Highly Informative Feature Set Over High Dimensions

Chongsheng Zhang, Florent Masseglia, Xiangliang Zhang

To cite this version:

HAL Id: lirmm-00753807
https://hal-lirmm.ccsd.cnrs.fr/lirmm-00753807
Submitted on 19 Nov 2012

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Discovering Highly Informative Feature Set Over High Dimensions

Chongsheng Zhang*
Henan University
475004 Kaifeng, China
Email: Chongsheng.Zhang@yahoo.com

Florent Masseglia*
Zenith Team, INRIA
34095 Montpellier, France
Email: Florent.Masseglia@inria.fr

Xiangliang Zhang
MINE Team, KAUST
Thuwal 23955-6900, Saudi Arabia
Email: Xiangliang.Zhang@kaust.edu.sa

Abstract—For many textual collections, the number of features is often overly large. These features can be very redundant, it is therefore desirable to have a small, succinct, yet highly informative collection of features that describes the key characteristics of a dataset. Information theory is one such tool for us to obtain this feature collection. With this paper, we mainly contribute to the improvement of efficiency for the process of selecting the most informative feature set over high-dimensional unlabeled data. We propose a heuristic theory for informative feature set selection from high dimensional data. Moreover, we design data structures that enable us to compute the entropies of the candidate feature sets efficiently. We also develop a simple pruning strategy that eliminates the hopeless candidates at each forward selection step. We test our method through experiments on real-world data sets, showing that our proposal is very efficient.

I. INTRODUCTION

Feature selection is the task of selecting interesting or important features, and removing irrelevant or redundant ones. It has been widely used in many application fields, such as discriminative gene selection [3] and text categorization [8]. Before selecting the features, an interestingness measure should be defined to assess the significance of the features or featuresets (i.e. feature sets). For instance, if the interestingness is measured by a given utility function, we select feature sets with the best (highest or lowest) utility scores. Given the measurement for the feature sets, one can design efficient algorithms to search for the best feature sets, where wrapper and filter [5] are the commonly used methods.

Besides the interestingness measures, feature selection is closely related to the characteristics of the data itself. Feature selection techniques for labeled data are rather different from those for unlabeled data [1]. For unlabeled data, we do not have class as references and feature selection always depends on the applications and tasks. For example, when choosing the top-k most important keywords from large documents, the $tf-idf$ method [6] can be used to measure the weight of each keyword; when finding a feature set that contains the most information, one can resort to information theory based feature selection methods [2], [4]. Example 1 addresses the problem of feature selection for document retrieval.

Example 1: We would like to retrieve documents from table I, in which the columns of \{O_1, ..., O_{10}\} are 10 documents, and the attributes of \{A, B, C, D, E\} are some features (key words) in the documents, where the value “1” means that the feature is contained in the document, and “0” not. It is easy to find that \(D, E\) is a frequent featureset, because features \(D\) and \(E\) occur together in nearly every document. However, it provides little help for document retrieval. By contrast, \((A, B, C)\) is an infrequent featureset, as its member features rarely or never appear together in the data. And it is troublesome to summarize the value patterns of featureset \((A, B, C)\). Providing it with the values < 1, 0, 0 > we could find the corresponding document \(O_5\); similarly, given the values < 0, 1, 1 > we will have the according document \(O_6\). Although \((A, B, C)\) is infrequent, it contains lots of useful information which is hard to summarize. We call it an informative feature set.

From the above example, we can see that it is useful to discover informative feature sets. Indeed, information theory provides strong support to measure the informativeness of the feature sets. But for unlabeled data, it requires computing the joint probability not just for the case when all the features have appeared together, but also for all the other cases when some features have appeared in the transactions but others have not. This is an exhaustive work. To tackle this problem, the authors in [4] proposed an effective and efficient method for selecting the informative feature sets over low-dimensions. However, this algorithm does not scale well for high-dimensions.

In this paper, we investigate the problem of efficient discovery of most informative feature set over unlabeled high-dimensional data. This is a very challenging task. First, there are \(\binom{d}{k}\) candidate feature sets having \(k\) features, \(k\) is a user-specified parameter denoting the size of the feature set.
that s/he expects, and \( d \) is the data dimension. Because of the high dimensions, the number of candidates increases greatly w.r.t. \( d \). Second, due to the characteristics of informative feature sets, we have to compute for each candidate feature set, the probability for every existing case of feature composition in all the transactions, and there are up to \( 2^k \) possible cases of feature combinations. If the number of transactions is \( n \), we need as many as \( n \times 2^k \) computations for each candidate feature set. Overall, we need \( n \times 2^k \times \frac{d!}{k!(d-k)!} \) calculations for all the candidates. This is very computation-demanding.

To solve the above problem, we propose a heuristic theory. Exploiting the fact that usually only a few features are correlated in most textual collections, our heuristic theory i) divides the features into independent and dependent sets and ii) reduces the candidate features for informative feature sets to a rather small subset. Upon this theory, we introduce a forward selection algorithm that discovers the most informative feature set based on the selected features. Moreover, we design a data structure that speeds up the computation of the entropies of the candidate featuresets. In all the transactions, and there are up to \( 2^k \) possible cases of feature composition, we have to compute for each candidate feature set, \( w \) r.t. the probability for every existing case of feature composition. This is very computation-demanding.

The remaining of this paper is organized as follows. In section II, we discuss the related work about feature selection. After giving the related definitions in section III, we introduce the theory for feature set mining and give our algorithm and two optimization techniques, as well as the concrete IIS framework in section IV. We evaluate our work in section V. Finally, we conclude and give perspectives in section VI.

II. RELATED WORK

Feature selection is a very important tool for many data mining applications. In [5] we find a systematic review of the foundations, techniques and applications of existing feature selection methods. This paper focuses on selecting the most informative feature set over high-dimensional unlabeled data. We therefore discuss the related work referring the keywords of informative feature set, high-dimensional data and unlabeled data.

Informative feature set selection on unlabeled data has been studied in recent years. Knobbe et al. in [4] proposed a heuristic algorithm to extract informative featuresets of size \( k \) with the highest entropy. Such featuresets are called \( \text{miki} \). The proposed \text{ForwardSelection} algorithm performs multiple scans over the dataset. Within each scan, \( \text{miki} \)'s size gradually increases, by adding a new feature \( f \) to the \( \text{miki} \) at current iteration. They showed the advantage of \text{ForwardSelection} over the brute force algorithms that evaluate the entropies of all the possible subsets of size \( k \). Afterwards, method for low entropy feature set discovery was also investigated. In [2], Heikinheimo et. al. proposed two algorithms designed to extract the sets of both low entropy and high entropy featuresets from static datasets. In [10], we extended \text{miki} to work on streaming data with transient features and proposed to reduce its candidates by removing similar sets according to mutual information criteria. But these methods become inefficient in high-dimensions, because we will have huge candidate feature sets to check in high-dimensions, and for each candidate we have to calculate the counts and compute the probability for all appeared feature combinations, including the ones where some features appeared but others did not. Our work aims to speed up the searching and computation for most informative feature set.

There are many approaches in feature selection for high-dimensional data. But most of them are towards labelled data. Yu and Liu [9] use the class concept as the reference and identify all predominant features which are most capable to predict class concept. The proposed method is based on discovering pair uncertainty correlation with the class concept and other features. They only considered the pair correlation-ship, but the cases that several features jointly decide the class concept have not been considered. Our work is about informative feature set selection over unlabeled and high dimensional data, and the task is to find the feature set with highest joint uncertainty, whereas the features can be dependent or independent of others. \text{FastANOVA} [11] was designed for joint association discovery, but strong association (or correlation) within a feature set does not imply a high information (i.e. joint entropy) contained in the feature set. In fact, even though the features in the set are weakly associated, their joint entropy can be very high.

III. DEFINITIONS

Preliminary Definitions. We refer to book [7] for the preliminary definitions of entropy, joint entropy of a feature set, mutual information and relevance coefficient. Let \( H(X) \) denote the entropy of feature \( X \), and \( H(I) \) represent the joint entropy of featureset \( I \). The mutual information between two discrete variables can be computed by Formula 1. The \text{Relevance Coefficient} (RO) between two features is measured by Formula 2. Features A and B are independent if \( RO(A, B) = 0 \); A and B are mutually redundant if \( RO(A, B) = 1 \).

\[
MI(AB) = H(A) + H(B) - H(AB) \tag{1}
\]

\[
RO(A, B) = \frac{MI(A, B)}{\max\{H(A), H(B)\}} \tag{2}
\]

To relax the conditions of feature independence and redundancy, we introduce the following definition and equation to measure the relationship between two features, using a lower bound threshold \( \xi \) and a upper bound threshold \( \delta \).

Definition 1: The \text{relationship} between two features depends on their RO value.
A and B are \[
\begin{cases}
\text{independent,} & \text{if } RO(A, B) < \xi; \\
\text{redundant,} & \text{if } RO(A, B) > \delta; \\
\text{relevant,} & \text{otherwise}
\end{cases}
\]

Now let us define the highly/most informative feature set.

**Definition 2:** Given a size \( k \), the **Highly/Most Informative Featureset** (HI) is the featureset that has the largest entropy value among all the candidates having size \( k \), as given by formula 4.

\[
HI = \text{MAX}\{H(IS_k), IS_k \in SS_k \}
\]

where \( IS = \{I_1, I_2, ..., I_n\} \) is the set of all \( n \) possible features, \( IS_k = \{I_{m_1}, I_{m_2}, ..., I_{m_k}\} \) is a featureset of size \( k \) \((k < n)\), \( SS_k = \{IS_k\} \) is the set of all possible featuresets of size \( k \).

**Problem Definition.** In this paper, we aim at discovering the most informative featureset over high-dimensional unlabeled dataset. Given such high dimensions, our approach should reduce the search space and be as effective as possible.

**IV. INFORMATIVE FEATURESET SELECTION OVER HIGH DIMENSIONS**

In this section, we first introduce a heuristic theory that forms the foundation of our framework. Then we present the MIKS forward feature selection algorithm. We next introduce two techniques to speed up the searching and computation for highly informative featureset. Finally, we introduce the IIS framework.

**A. The Heuristic Feature Reduction Theory**

Before illustrating the theory, we first give some basic ideas and concepts. \( k \) is a user input parameter, denoting the size of the expected informative feature set. In our theory, we perform the following steps:

1) Sort the features by entropy;
2) Evaluate independence between features, and categorize them into dependent and independent sets;
3) Fetch the top-\( k \) features from the independent sets and put them into BRFS (Brief Reference Feature Set);
4) Get all the features which rank higher than the last feature in BRFS, and insert them into MiniSet (Minimum candidate feature Set);
5) Considering some features in MiniSet could have correlated features, we maintain MaxSet to include both MiniSet and all the features which are correlated with features in it;

Finally, our theory limits the search space of informative feature sets to MaxSet, so we can perform informative feature set selection methods on MaxSet. We explain the reasons for choosing MaxSet in the following.

We first refer to the following theorem in [7].

**Theorem 1:** If and only if variables \( X_1, X_2, ..., X_{n-1}, X_n \) are mutually independent, then

\[
H(X_1, X_2, ..., X_{n-1}, X_n) = \sum_{i=1}^{n} H(X_i)
\]

If the condition in theorem 1 is satisfied, it will be simple to compute the joint entropy. Otherwise, we claim the following inference.

**Lemma 1:** If \( S = \{X_1, X_2, ..., X_{m-1}, X_{m+1}, ..., X_{n-1}, X_n\} \), and \( X_m \leq m \leq n \) is independent of any of the features in \( S \), but the features in \( S \) are not necessarily mutually independent. Then

\[
H(X_1, X_2, ..., X_{m-1}, X_m, X_{m+1}, ..., X_{n-1}, X_n) \\
\neq H(X_1, X_2, ..., X_{m-1}, X_{m+1}, ..., X_{n-1}, X_n)
\]

**Proof:** The reason is that \( X_m \) is not independent of \( S \).

Let us take a small example of three variables \( A, B \) and \( C \). In table II, the left part is the data, and the right part is the joint probability of these three features.

Judging from the joint probability of \((A,B)\) and \((A,C)\), we can see that \( A \) and \( B \) and \( C \) are independent respectively. However, this is not the case for \((A,B,C)\). Actually, \( P(A = 0, B = 1, C = 0) = 1/4 \), whereas \( P(A = 0) = 1/2, P(B = 1, C = 0) = 1/4 \) Thus, \( P(A = 0, B = 1, C = 0) \neq P(A = 0) \times P(B = 1, C = 0). \) So \( A \) is not independent of \((B,C)\).

Based on Theorem 1 and Lemma 1, we infer the importance of independence among features as expressed in Lemma 2.

**Lemma 2:** We can not use Theorem 1 unless all the features in the given set are mutually independent. Otherwise, it is unreliable because of Lemma 1.

In practice, in many real-world high-dimensional datasets, only a few features are correlated, most of them are independent. Therefore, we can safely rely on Theorem 1 to compute the joint entropy for a feature set comprising mutually independent features. When the feature sets include part of the dependent features, theorem 1 can not be used directly. However, it can still be used to estimate the bounds for the joint entropies of the feature sets, such that some unpromising candidates can be filtered out first. Next, for the remaining candidate sets, we still need to compute the exact joint entropy values. Therefore, we introduce Lemma 3 in IIS.

**Lemma 3:** Let \( f_s \) be a subset of size \( k \) from MaxSet. \( f_s \) can never become the most informative featureset, if its maximum entropy is lower than BRFS.

**Proof:** Suppose we have two candidates, \( f_s \) and BRFS. Because the features in BRFS are top-K mutually independent features, the joint entropy of BRFS will be the sum entropy of the individual features according to theorem 1. As \( f_s \)'s entropy is lower than that of BRFS, we will choose BRFS as the most informative featureset.

Given the above lemmas and discussions, we finally provide the following Heuristic Feature Reduction Theory: the most informative feature set is more likely to be a subset from MaxSet.
This heuristic theory can be illustrated in three folds. In the first place, the most informative feature set from the independent set is BRFSet. BRFSet is the last and reference candidate we have. We can see from lemma 3 that if the upper bound of a featureset’s entropy is smaller than the entropy of BRFSet, it can be safely discarded. Next, features in \{MaxSet – BRFSet\} already contain all the top-ranked dependent features and their correlated ones, from which we can select a subset of most informative featuresets. Finally, now that each feature in BRFSet is independent of any other feature in MaxSet, it is more likely that the one from BRFSet, instead of the remaining independent features, will contribute more to the joint entropy of a featureset. In subsection IV-E, we will build the IIS framework based on this theory.

### B. MIFS: Mining Highly Informative Featureset

With the above heuristic feature reduction theory, we can greatly reduce the number of features for highly informative featureset discovery. Even so, we may still have a very large number of candidates. Let 
\( d \)
 denote the number of features, 
\( n \)
 is the number of transactions and 
\( k \)
 is the size of featureset, the number of candidate sets having size 
\( k \)
 is:

\[
nc = \frac{d!}{k! \times (d-k)!} \tag{7}
\]

For instance, if 
\( d = 40, k = 20 \)
, then 
\( nc = 1.3785 \times 10^{11} \)
. Therefore we can not afford to enumerate all the candidates of size 
\( k \)
, instead, we have to resort to the heuristic algorithms for the consideration of efficiency.

Our proposed feature selection algorithm is named MIFS. Initially, the top two features are the pair features with the largest joint entropy. We insert them into a set 
\( mifs \)
. Then iteratively, we first check the first feature 
\( f \)
 in \( \{MaxSet – MIFS\} \). If 
\( f \)
 and features in 
\( mifs \)
 are mutually independent, then we simply add 
\( f \)
 to 
\( mifs \); otherwise, based on the current 
\( mifs \)
 and 
\( f \)
, we generate all the possible candidates having size 
\( i \). At the end of this round, for all the candidates, we scan the data to compute their exact entropy values and find a feature subset Maxsubset (of size 
\( i \)) with the largest entropy value. In the next iteration 
\( i + 1 \), we vacuum the candidate features in the iteration 
\( i \) gradually increases from 2 to 
\( k \), and after 
\( k – 2 \) iterations, we will have the final highly informative feature set 
\( mifs \) of size 
\( k \).

### C. Selecting Only the Promising Featuresets

In this part we present a strategy which can help us filter out the unpromising featuresets at each iteration in the above MIFS algorithm, based on Lemma 3. We compare the bounds of each candidate feature set with the bounds the reference. Initially, the reference set is the top-i features in BRFSet, 
\( i \) is the current iteration number in algorithm MIFS. Let 
\( lb \)
 and 
\( ub \)
 denote the lower and upper bounds. There are 6 cases we need to consider. Bounds comparison is simple so we skip the comparison details. The final strategy is: candidates whose entropy upper bounds are lower than 
\( lb \) will be discarded without consideration; all existing candidates will be removed when there is a featureset whose entropy lower bound is larger than 
\( ub \), and this featureset will be considered as the new reference; in the rest cases we update 
\( lb \) or 
\( ub \) correspondingly.

### D. MFI Data Structure: Pattern Mapping and Feature Indexing for feature selection

The complexity of the MIFS algorithm is 
\( |k| \times |m| \times |n| \), where 
\( k \) is the size of the feature set, 
\( |m| \) is the total number of features in MaxSet, 
\( |n| \) is the size of the data. We find that we have to scan the data several times, which is costly. In order to reduce the cost, we make a data structure MFI to help us efficiently find the related records. We maintain two kinds of data structures in MFI:

(i) A pattern map/table which summarizes all the appeared feature sets and their counts, if two users share the same feature sets, then the count for the pattern in the map is 2;

(ii) Feature indices which keep all the 
\( ids \) of the patterns containing this feature. Note that, as we give an order of the patterns by their occurrence time, the pattern 
\( ids \) in each feature index are monotone increasing. This monotonic feature facilitates the search and merge of 
\( ids \) from different indices.

---

**Example 2:** Data management in MFI. In Figure 1 (1) we have the transactions in the data, and the data structure for
storing and identifying the complete patterns is given in Figure 1 (2). Figure 1 (3) is the indices for different features of A, B, C and D. Each time we read a transaction, we check whether the feature set of the object already exists in the pattern map. If yes, we increase the count for the pattern by 1. Otherwise, we first insert the feature set in the pattern map with pid as the identifier, then add the new pid to the end of related indices. For instance, for the red color transaction $T_{12}$, because the pattern $(A, C)$ does not exist in the pattern map, we insert the pattern. Next we insert the according pattern id in indices for features A and C.

When we are incrementally searching for the most informative feature set, at each step, we need to compute the entropy of all the new feature subsets which are generated by adding one feature to the current most informative feature subset. There we need MFI to quickly find the relevant records.

Example 3: Efficient Computations of Entropy in MFI. Let $(A, B)$ be the most informative feature subset in the current round. In the following round, we will have to compute the entropy of $(A, B, C)$ and the combination of $(A, B)$ with either of the remaining features. Owing to our index, when computing the entropy of featureset $(A, B, C)$, we do not have to navigate all the records, instead, all we need to check are patterns kept in index A, B, or C. Thus the computation cost on the entropy will be greatly cut.

According to Formula 7, the search space rapidly increases with the number of features. Thus, the larger the data and the number of features, the more useful MFI is.

E. The IIS Framework

We build the IIS framework that implements our informative feature sets discovery theory proposed in sub-session IV-A, generates featureset candidates from MaxSet using MIFS described in sub-session IV-B, then removes unpromising candidates using techniques in subsection IV-C. When computing the exact entropies of the remaining techniques, it utilizes techniques in sub-session IV-D. MIFS stops when the length of the selected feature set is $k$.

V. EXPERIMENTS

Data Set. We use two kinds of real-world datasets in the experiments: Web logs kept by the server at our research institute, and data from a Telecom that maintains the portal visits made by the clients using their mobile devices. The information of the datasets are depicted in Table III in which the first 3 datasets belong to the Web log dataset and the rest are part of Telecom dataset. There the field of users denotes the number of unique users appeared in the dataset, but one user could request different pages at separate transactions, so we summarize the transactions by users.

Parameter Setting. There are two thresholds for relevance and redundancy, and one user-specified parameter: $k$, which is the size/length of the feature set. Regarding the thresholds, $\xi$ is set to 0.01, and the value for $\delta$ is 0.99. We tested different parameters, and these two values are empirical ones.

We use MIKI from [4] as the reference for evaluating the informative feature set mining problem. For efficiency, we use CPU time as the criteria. For evaluating the quality of the algorithms, we use the selected feature sets and their entropy values as the standards. We first look at the selected features, if they are the same, then both algorithms are considered having the same effectiveness. Otherwise, we will check the according entropies, and the one with higher entropy is deemed superior.

Results of WWW datasets. (A), (B) and (C) in Table IV show the experimental results on dataset1, dataset2 and dataset3. In Table IV (A), the “selected features” field denotes the selected informative feature sets, given different $k$ values (because the names of the features are long URL addresses, we only unique ids to represent them). The terms such as “+17” mean that the new selected features (e.g. $k = 6$) are composed of feature 17 and the previous result (e.g. $k = 5$). We have the following observations from the results.

(I) No matter how the value of $k$ varies, the selected features and their respective entropy of IIS is always the same as MIKI. This observation verified our heuristic theory. For example, when $k$ is 10, the feature sets discovered by MIKI and IIS are the same, and so it does with the according entropy values.

(II) No matter how the value of $k$ varies, IIS is always much more efficient than MIKI. We can see that for WWW datasets, IIS saves up to 90% of the time cost by MIKI. As an example, when $k$ is 15, MIKI takes 898 seconds to get the result, but IIS only uses 90 seconds.

(III) For the same dataset, when $k$ increases, it takes much more time to have the final results. MIKI suffers a lot when $k$ increases, but the case is much more better for IIS.

The above observations still hold for dataset2 and dataset3. We also observe that when $k$ is the same, the more features a dataset has, the longer time it takes. Because the more features we have, the more candidates need to be evaluated. But since IIS limits the candidate features to a small subset, it suffers less than MIKI. For instance, for the results when $k$ is 30 on dataset2 and dataset3, the computation time for MIKI increased by 38.2%, but IIS only increased 28.2%.

Results of Telecom datasets. We also tested IIS on Telecom datasets, which have much more features than WWW datasets. (D), (E), and (F) in Table IV show the results of both algorithms on dataset4, dataset5 and dataset6. Similar to the results of WWW datasets, the selected features and entropy are always the same for both algorithms, but IIS uses less time. For example, when $k = 5$, MIKI uses 3641 seconds on testdata5, but our approach only uses 52 seconds, which is 1.4% of that for MIKI. It is clear that when the number of features is very large, IIS is much more efficient.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>transactions</th>
<th>features</th>
<th>users</th>
</tr>
</thead>
<tbody>
<tr>
<td>dataset1</td>
<td>120000</td>
<td>224</td>
<td>28581</td>
</tr>
<tr>
<td>dataset2</td>
<td>240000</td>
<td>264</td>
<td>51611</td>
</tr>
<tr>
<td>dataset3</td>
<td>300000</td>
<td>285</td>
<td>65361</td>
</tr>
<tr>
<td>dataset4</td>
<td>120000</td>
<td>1351</td>
<td>76423</td>
</tr>
<tr>
<td>dataset5</td>
<td>240000</td>
<td>1662</td>
<td>67470</td>
</tr>
<tr>
<td>dataset6</td>
<td>300000</td>
<td>1849</td>
<td>74538</td>
</tr>
</tbody>
</table>
TABLE IV
RESULTS OF WWW (A,B,C) AND Telecom (D,E,F) DATASETS

(A) DATASET1

<table>
<thead>
<tr>
<th>K</th>
<th>MIKI</th>
<th>time(s)</th>
<th>selected features</th>
<th>entropy</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>141</td>
<td>2 14 11 18 20</td>
<td>0.4866</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>192</td>
<td>+7</td>
<td>0.5588</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>247</td>
<td>+25</td>
<td>0.6284</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>307</td>
<td>+13</td>
<td>0.6954</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>377</td>
<td>+51</td>
<td>0.7582</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>448</td>
<td>+24</td>
<td>0.8164</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>898</td>
<td>+15</td>
<td>0.9719</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>1502</td>
<td>+27</td>
<td>1.0176</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>2255</td>
<td>+30</td>
<td>1.0560</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>3094</td>
<td>+10</td>
<td>1.1245</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IIS</th>
<th>time(s)</th>
<th>selected features</th>
<th>entropy</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>2 11 18 20</td>
<td>0.4866</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>+17</td>
<td>0.5588</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>+25</td>
<td>0.6284</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>+13</td>
<td>0.6954</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>+51</td>
<td>0.7582</td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>+24</td>
<td>0.8164</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>+15</td>
<td>0.9719</td>
<td></td>
</tr>
<tr>
<td>164</td>
<td>+27</td>
<td>1.0176</td>
<td></td>
</tr>
<tr>
<td>266</td>
<td>+7</td>
<td>1.0560</td>
<td></td>
</tr>
<tr>
<td>398</td>
<td>+10</td>
<td>1.1245</td>
<td></td>
</tr>
</tbody>
</table>

(B) DATASET2

<table>
<thead>
<tr>
<th>K</th>
<th>MIKI</th>
<th>time(s)</th>
<th>entropy value</th>
<th>Time</th>
<th>entropy value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>301</td>
<td>0.44826</td>
<td>23</td>
<td>0.44826</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>408</td>
<td>0.31657</td>
<td>34</td>
<td>0.31657</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>524</td>
<td>0.58088</td>
<td>46</td>
<td>0.58088</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>655</td>
<td>0.64075</td>
<td>60</td>
<td>0.64075</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>803</td>
<td>0.69963</td>
<td>75</td>
<td>0.69963</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>957</td>
<td>0.75351</td>
<td>92</td>
<td>0.75351</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>1934</td>
<td>0.9719</td>
<td>209</td>
<td>0.9719</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>3271</td>
<td>1.13181</td>
<td>380</td>
<td>1.13181</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>4888</td>
<td>1.24471</td>
<td>615</td>
<td>1.24471</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>6800</td>
<td>1.3296</td>
<td>912</td>
<td>1.3296</td>
<td></td>
</tr>
</tbody>
</table>

(C) DATASET3

<table>
<thead>
<tr>
<th>K</th>
<th>MIKI</th>
<th>time(s)</th>
<th>entropy value</th>
<th>Time</th>
<th>entropy value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>410</td>
<td>0.45036</td>
<td>37</td>
<td>0.45036</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>556</td>
<td>0.31672</td>
<td>51</td>
<td>0.31672</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>719</td>
<td>0.58275</td>
<td>67</td>
<td>0.58275</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>894</td>
<td>0.64269</td>
<td>84</td>
<td>0.64269</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>1096</td>
<td>0.70155</td>
<td>105</td>
<td>0.70155</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1313</td>
<td>0.75474</td>
<td>128</td>
<td>0.75474</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>2648</td>
<td>0.97216</td>
<td>276</td>
<td>0.97216</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>4486</td>
<td>1.13377</td>
<td>497</td>
<td>1.13377</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>6748</td>
<td>1.2476</td>
<td>792</td>
<td>1.2476</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>9403</td>
<td>1.33378</td>
<td>1169</td>
<td>1.33378</td>
<td></td>
</tr>
</tbody>
</table>

(D) DATASET4

<table>
<thead>
<tr>
<th>K</th>
<th>MIKI</th>
<th>time(s)</th>
<th>selected features</th>
<th>entropy</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1560</td>
<td>14 17 24 28 60</td>
<td>0.8596</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>2108</td>
<td>+21</td>
<td>0.9284</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>2716</td>
<td>+15</td>
<td>0.9943</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>3367</td>
<td>+37</td>
<td>1.0446</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>4110</td>
<td>+40</td>
<td>1.0849</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>5187</td>
<td>+20</td>
<td>1.1245</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>7649</td>
<td>+19</td>
<td>1.2890</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IIS</th>
<th>time(s)</th>
<th>selected features</th>
<th>entropy</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>14 17 24 28 60</td>
<td>0.8596</td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>+21</td>
<td>0.9284</td>
<td></td>
</tr>
<tr>
<td>55</td>
<td>+15</td>
<td>0.9943</td>
<td></td>
</tr>
<tr>
<td>69</td>
<td>+37</td>
<td>1.0446</td>
<td></td>
</tr>
<tr>
<td>85</td>
<td>+40</td>
<td>1.0849</td>
<td></td>
</tr>
<tr>
<td>102</td>
<td>+20</td>
<td>1.1245</td>
<td></td>
</tr>
<tr>
<td>180</td>
<td>+19</td>
<td>1.2890</td>
<td></td>
</tr>
</tbody>
</table>

(E) DATASET5

<table>
<thead>
<tr>
<th>K</th>
<th>MIKI</th>
<th>time(s)</th>
<th>entropy value</th>
<th>Time</th>
<th>entropy value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>3641</td>
<td>0.956659</td>
<td>52</td>
<td>0.956659</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>4939</td>
<td>1.01765</td>
<td>71</td>
<td>1.01765</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>9333</td>
<td>1.19006</td>
<td>173</td>
<td>1.19006</td>
<td></td>
</tr>
</tbody>
</table>

(F) DATASET6

<table>
<thead>
<tr>
<th>K</th>
<th>MIKI</th>
<th>time(s)</th>
<th>entropy value</th>
<th>Time</th>
<th>entropy value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>4327</td>
<td>0.75494</td>
<td>54</td>
<td>0.75494</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>5859</td>
<td>0.81429</td>
<td>74</td>
<td>0.81429</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>10949</td>
<td>0.98128</td>
<td>177</td>
<td>0.98128</td>
<td></td>
</tr>
</tbody>
</table>

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a heuristic theory to reduce the dimension for informative feature set discovery from high dimensions. We also introduced structures and a strategy to accelerate the computation of exact entropy of the candidates and prune hopeless featuresets. In the future work we will investigate how to adapt our method to high-dimensional streaming data.

REFERENCES
