Implementation of FAST Clustering-Based Feature Subset Selection Algorithm for High-Dimensional Data

Smit Shilu, Kushal Sheth and Ekata Mehul

Abstract In feature selection, we are concerned with finding out those features that produces result similar to those of the original entire set of features. We concern ourselves with efficiency and effectiveness while evaluating Feature selection algorithms. Efficiency deals with the time required to find a subset of features and effectiveness, with the quality of subset of features. On these criteria, a FAST clustering-based feature selection algorithm (FAST) has been proposed and experimentally evaluated and implemented in this paper. The dimensionality reduction of data is the most important feature of FAST. First, we use graph-theoretic clustering method to divide features into clusters. Next, we form a subset of features by selecting the feature which is most representative and strongly related to the target classes. Due to features in different clusters being relatively independent; the clustering-based strategy of FAST has a high probability of providing us with a subset of features which are both useful and independent. Efficiency of FAST is ensured by using the concept of minimum spanning tree (MST) along with kruskal's algorithm.

Keywords Feature subset selection · Feature clustering · Filter method · Kruskal's algorithm · Graph-based clustering

S. Shilu (✉) · K. Sheth
eiTRA eInfochips Training, Charusat University, Changa, India
e-mail: smitshilu@gmail.com
URL: http://www.charusat.ac.in

K. Sheth
e-mail: kushalsheth28@gmail.com
URL: http://www.charusat.ac.in

E. Mehul
Research Academy, Ahmedabad, India
e-mail: ekata.mehul@eitra.org
URL: http://www.eitra.org

S.C. Satapathy et al. (eds.), *Proceedings of International Conference on ICT for Sustainable Development*, Advances in Intelligent Systems and Computing 409, DOI 10.1007/978-981-10-0135-2_19

1 Introduction

With the aim of selecting the important and useful features, by reducing the dimensionality of the data and increasing the accuracy, we here test and develop the code for the FAST Clustering-based Subset Selection Algorithm. Generally, many feature subset selection methods have been proposed. These methods can be categorized into four different categories: Embedded, Wrapper, Filter, and Hybrid approaches. Embedded methods use the complete training set and hence have more accuracy compared to other three methods. Decision trees and artificial neural networks are examples of this embedded method. The wrapper method uses the predictive accuracy of predetermined learning algorithm and its accuracy is generally very high. However, computational complexity is on the higher side. The wrapper methods are tending to over fit on small training sets with expensive computationally. The filter methods do not require learning algorithms. Their computational complexity is low, but the accuracy of the learning algorithms is not guaranteed. It is favorable when dataset is having very large number of features. The hybrid methods are a combination of filter and wrapper methods; with the use of filter method it can reduce search space that will be considered by the subsequent wrapper. Here, in our algorithm, we use the filter method. The FAST algorithm does two main tasks, (1) Removing the irrelevant data, (2) Removing the redundant data, and works in three steps. In the first step, the features are divided into clusters using graph-theoretic methods; and in the second step, the most relevant feature from each cluster is selected and inserted into the final cluster. We have tested this algorithm on various datasets and also integrated it with WEKA for public use. The rest of the paper is organized as follows: Sect. 2 consists of the Algorithm part. Section 3 contains the working flow of the algorithm stepwise. Section 4 contains the experimental environment. Section 4 contains the results and finally, Sect. 5 contains information regarding the future enhancements, conclusions and acknowledgments.

2 Algorithm

2.1 Definitions

T-Relevance: The relevance between the individual feature which in our case is F_i and the target concept C is known as T-Relevance. This relevance value is denoted by $SU(F_i, C)$. Here, if the relevance value is greater than a certain predetermined threshold value θ, then we can say that the feature Fi is a strong feature.

F-Correlation: The relation between the pair of features is known as F-Correlation. If the two features are F_i and F_j, then the correlation value is denoted as $SU(F_i, F_j)$.

Irrelevant and Redundant Feature: Irrelevant features are those features which have no relevance to the target class or concept while redundant features are those which do not provide any new information and can be taken out of cluster.

2.2 *Algorithm*

FAST algorithm is having mainly three steps. In the first step, remove irrelevant feature. In the second, generate a MST. In the third, break the MST and select feature for final cluster.

```
   inputs: $D(F_1, F_2, ..., F_m, C)$ - the given data set
           $\theta$ - the T-Relevance threshold.
   output: $S$ - selected feature subset .
   //==== Part 1 : Irrelevant Feature Removal ====
1  for $i = 1$ to $m$ do
2      T-Relevance = SU $(F_i, C)$
3      if T-Relevance $> \theta$ then
4          $S = S \cup \{F_i\}$;
   //==== Part 2 : Minimum Spanning Tree Construction ====
5  $G$ = NULL; //$G$ is a complete graph
6  for each pair of features $\{F'_i, F'_j\} \subset S$ do
7      F-Correlation = SU $(F'_i, F'_j)$
8      Add $F'_i$ and/or $F'_j$ to $G$ with F-Correlation as the weight of
       the corresponding edge;
9  minSpanTree = Prim (G); //Using Prim Algorithm to generate the
     minimum spanning tree
   //==== Part 3 : Tree Partition and Representative Feature Selection ====
10 Forest = minSpanTree
11 for each edge $E_{ij} \in$ Forest do
12     if $\mathrm{SU}(F'_i, F'_j) < \mathrm{SU}(F'_i, C) \wedge \mathrm{SU}(F'_i, F'_j) < \mathrm{SU}(F'_j, C)$ then
13         Forest = Forest $- E_{ij}$
14 $S = \phi$
15 for each tree $T_i \in$ Forest do
16     $F_R^j = \mathrm{argmax}_{F'_k \in T_i} \mathrm{SU}(F'_k, C)$
17     $S = S \cup \{F_R^j\}$;
18 return $S$
```

3 Working Flow

3.1 *Flowchart*

See Fig. 1.

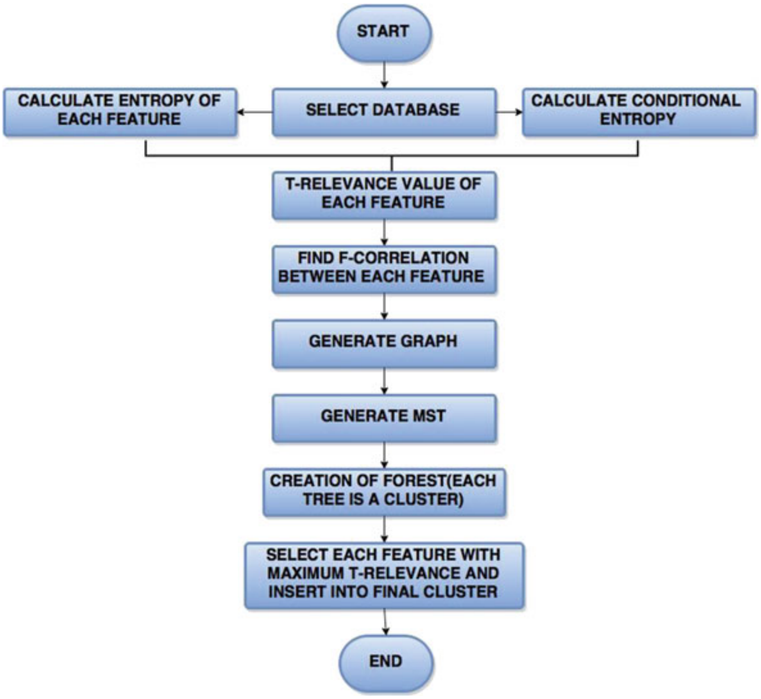

START
SELECT DATABASE
CALCULATE ENTROPY OF EACH FEATURE
CALCULATE CONDITIONAL ENTROPY
T-RELEVANCE VALUE OF EACH FEATURE
FIND F-CORRELATION BETWEEN EACH FEATURE
GENERATE GRAPH
GENERATE MST
CREATION OF FOREST(EACH TREE IS A CLUSTER)
SELECT EACH FEATURE WITH MAXIMUM T-RELEVANCE AND INSERT INTO FINAL CLUSTER
END

Fig. 1 Working flow of algorithm

3.2 Information Gain Calculation

The features which have a strong correlation with the target concept can be considered as the relevant features. However, the features which have strong correlation with other features are considered to be the redundant features. Hence, the concept of redundant feature and relevant feature is based on the correlation values. In this module, Information Gain is calculated to find relevance value of each attribute. Here, we use the concept of Symmetric Uncertainty (SU) which is derived from mutual information and can be used to measure the goodness of features for classification.

$$\mathrm{Gain}(X \mid Y) = H(X) - H(X \mid Y) \tag{1}$$

$$\mathrm{Gain}(X \mid Y) = H(Y) - H(Y \mid X) \tag{2}$$

We can compute gain value by using the values entropy and conditional entropy. The equations for that are given below

$$H(X) = -\sum_{x \in X} p(x) log_2 p(x) \tag{3}$$

$$H(X) = -\sum_{x \in X} p(y) \sum_{x \in X} p(x|y) log_2 p(x|y) \tag{4}$$

3.3 T-Relevance Calculation

T-Relevance is the relevance value of a feature with respect to the target class. This value can be calculated using the Symmetric Uncertainty which uses the concept of information gain and conditional entropy.

$$SU(X, Y) = \frac{2^{*}\text{Gain}}{H(X) + H(Y)} \tag{5}$$

3.4 F-Correlation Calculation

F-Correlation is the correlation value of a feature with respect to other features in dataset. The Symmetric Uncertainty equation used to calculate the relevance value of features with respect to target concept is reapplied here to calculate the correlation among all the features.

3.5 MST (Minimal Spanning Tree) Generation

After the calculation of F-Correlation value, the next step is the construction of graph on basis of correlation values. This connected graph is to be converted into a minimal spanning tree to reduce the dimensionality of the dataset. Kruskal's algorithm is a greedy algorithm which is used for the construction of MST form the graph. Minimal Spanning Tree is a tree that contains all the vertices with minimum weight of edges and without any cycle.

Steps:

1. Generate a graph based on correlation values between each feature.
2. Convert graph into a MST using kruskal's algorithm.
3. At last, we get a single minimal spanning tree consisting of all the nodes.

3.6 Cluster Formation

After generating an MST, we need to reduce the tree to many smaller trees which will form a forest. Each of this tree will be viewed as a cluster.

Steps:

1. If the correlation value between the features (weight of edge) is smaller than the relevance value of both the features, the edge is connecting; then remove that edge.
2. On removing the unnecessary edges, a forest with trees is generated, each representing a single cluster.
3. From each cluster (tree), the feature with the highest relevance value is selected and inserted into the final cluster; in our case S.

4 Results

4.1 System Specification

Required system specifications are as below
System Specification
Processor: Intel(R) Core(TM) i7-3635QM CPU @ 2.40 GHz
Memory: 8.00 GB (7.89 GB usable)
System type: 64-bit Operating System, × 64 âĂß-based processor

4.2 Platform

Java

4.3 Software—Weka Integration

WEKA 3.7

4.4 Summary of Datasets

To find the effectiveness and performance of our implemented FAST algorithm, verifying whether or not the method is fully implemented in practice 24 data sets were used which are publicly available. The numbers of features of the 24 data sets vary from 15 to 26,833. The dimensionality of the 50 % data sets is having more than 5,000, of which 20.8 % data sets have more than 10,000 features (Table 1).

Here,

F: Attributes,
I: Instances, and
C: Class distinct values

Table 1 Summary of datasets

Sr. No.	Dataset	F	I	C
1	ada-agnostic	49	4562	2
2	ada-prior	15	4562	2
3	Chess	37	3196	2
4	connect-4	43	67557	3
5	fbis.wc	2001	2463	17
6	la1 s.wc	13196	3204	6
7	la2 s.wc	12433	3075	6
8	new3 s.wc	26833	9558	44
9	oh0.wc	3183	1003	10
10	oh10.wc	3239	1050	10
11	oh15.wc	3101	913	10
12	oh5.wc	3013	918	10
13	ohscal.wc	11466	11162	10
14	re0.wc	2887	1504	13
15	re1.wc	1657	3759	25
16	sylva.wc	109	14395	2
17	tr11.wc	6430	414	9
18	tr12.wc	5805	313	8
19	tr21.wc	7903	336	6
20	tr23.wc	5833	204	6
21	tr31.wc	10129	927	7
22	tr41.wc	7455	878	10
23	tr45.wc	8262	690	10
24	wap.wc	8461	1560	20

4.5 Results Analysis

Experimental results are presented in this section. This includes selected features and the time taken to select the features. We have tested this datasets in WEKA tool on machine with the following configuration (Table 2).

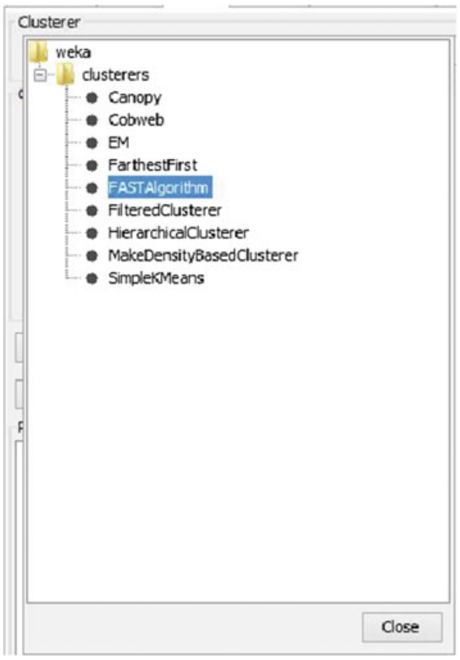
Processor: 2.3 GHz Intel i7
RAM: 8 GB
Operating system: Windows 8.1

Table 2 Number of features selected after applying FAST and runtime

Sr. No.	Dataset	Runtime (ms)	Selected feature
1	ada-agnostic	959.04	3
2	ada-prior	13006.98	4
3	Chess	59.94	3
4	connect-4	3016.98	5
5	fbis.wc	42847.11	19
6	la1 s.wc	123876	17
7	la2 s.wc	399600	18
8	new3 s.wc	1210788	29
9	oh0.wc	6733.26	12
10	oh10.wc	7602.39	11
11	oh15.wc	6383.61	14
12	oh5.wc	6103.89	14
13	ohscal.wc	399600	38
14	re0.wc	9480.51	11
15	re1.wc	13186.8	10
16	sylva.wc	51648.3	7
17	tr11.wc	10989	32
18	tr12.wc	5994	10
19	tr21.wc	11288.7	8
20	tr23.wc	5304.69	8
21	tr31.wc	44455.5	23
22	tr41.wc	24245.73	14
23	tr45.wc	26473.5	5
24	wap.wc	29170.8	17

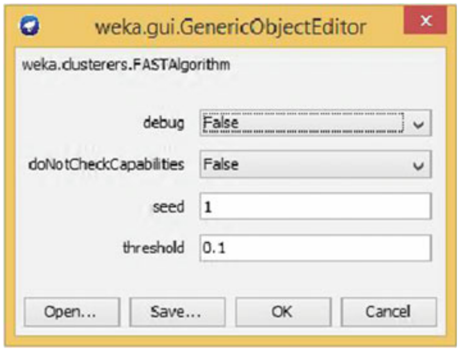
4.6 Experimental Results

See Figs. 2, 3, and 4.


Clusterer

weka
- clusterers
  - Canopy
  - Cobweb
  - EM
  - FarthestFirst
  - FASTAlgorithm
  - FilteredClusterer
  - HierarchicalClusterer
  - MakeDensityBasedClusterer
  - SimpleKMeans

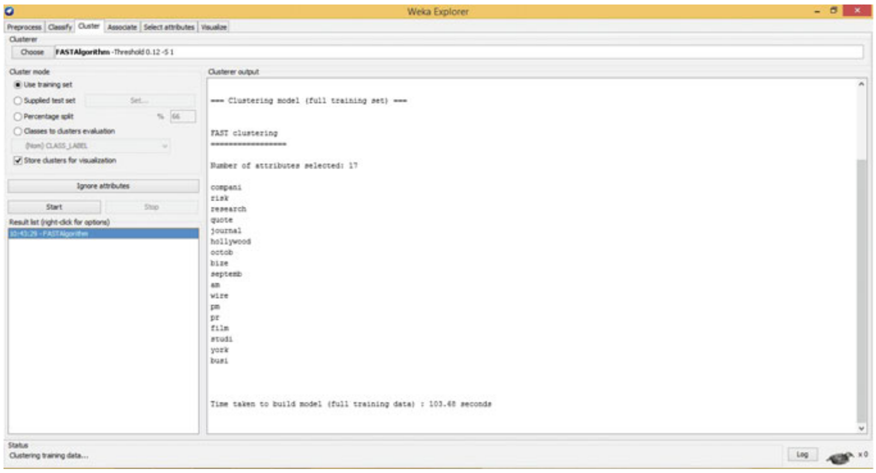
Fig. 2 Select FAST from cluster list


weka.gui.GenericObjectEditor

weka.clusterers.FASTAlgorithm

debug	False
doNotCheckCapabilities	False
seed	1
threshold	0.1

Fig. 3 Set threshold value



Clusterer: FASTAlgorithm -Threshold 0.12 -S 1

Cluster mode: [✓ Use training set] Supplied test set; Percentage split % 66; Classes to clusters evaluation (Nom) CLASS_LABEL; [✓ Store clusters for visualization]

Result list (right-click for options): 10:43:29 - FASTAlgorithm

Clusterer output

```
=== Clustering model (full training set) ===


FAST clustering
=================

Number of attributes selected: 17

compani
risk
research
quote
journal
hollywood
octob
bize
septemb
am
wire
pm
pr
film
studi
york
busi



Time taken to build model (full training data) : 103.48 seconds
```

Status: Clustering training data...

Fig. 4 Output WEKA screen

5 Conclusions and Future Enhancements

In this project, we present a FAST subset selection algorithm which can be used for big data. Basically, this algorithm includes three sub parts. The first part includes the task of removing the features which are not relevant and has low relevance value. In second part, a minimum spanning tree is constructed on basis of relevance and correlation values. Finally, the third part where the features with highest relevance value are included in final cluster. This algorithm reduces the dimensionality to a great level. This can be used as a prerequisite for the analysis purpose. For future work, we plan to work on the correlation between features and also plan to improve the accuracy of algorithm by reconsidering the logic of information gain.

Acknowledgments We thank Dr. Amit Ganatra, Dean and Head of Department Computer Engineering Department, Charusat University for comments that greatly improved the manuscript, and Mr. Chintan Bhatt, Professor Charusat University for assistance in implementation of this algorithm.

References

1. Song, Q., Ni, J., & Wang, G. (2013). A fast clustering-based feature subset selection algorithm for high - dimensional data.
2. Karthikeyan, P., Saravanan, P., & Vanitha, E. (2014). High dimensional data cluster-ing using fast cluster based feature selection. *International Journal of Engineering Research and Applications* , *4*(3), 65–71, (Version 1), ISSN: 2248-9622.

3. Yu, L., & Liu, H. (2004). Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research, 10*(5), 1205–1224.
4. Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research, 3*, 1157–1182.
5. Data used for result and analysis http://tunedit.org/repo/Data/Text-wc.
6. WEKA library documentations https://weka.wikispaces.com/.