IJDCST @Sep, Issue- V-2, I-6, SW-26
ISSN-2320-7884 (Online)
ISSN-2321-0257 (Print)

A Clustering-Based Feature Subset Selection for High Dimensional Data using Advanced Chameleon Algorithm

[1] Kiran Kumar Jakka, [2]K.Shirin Bhanu, [3] G. Loshma.
[1]Final M Tech Student, [2] Associate Professor,[3] Associate Professor & HOD
[1,3]Dept of Computer Science and Engineering ,
[2]Dept of Information Technology
[1,2,3] Sri Vasavi Engineering College, Tadepalligudem, W.G.dt, A.P.

Abstract: Feature selection involves identifying a subset of the most useful features that produces compatible results as the original entire set of features. A feature selection algorithm may be evaluated from both the efficiency and effectiveness points of view. While the efficiency concerns the time required to find a subset of features, the effectiveness is related to the quality of the subset of features. Based on these criteria, a fast clustering-based feature selection algorithm(FAST) and Advanced Chameleon is proposed and experimentally evaluated in this paper.

FAST is Tree-Based Algorithm and Advanced Chameleon is Graph-Based Algorithm. Features in different clusters are relatively independent; the clustering-based strategy of Chameleon has a high probability of producing a subset of useful and independent features. To ensure the efficiency of FAST, we adopt the efficient minimum-spanning tree clustering method, for Chameleon we adopt the K – Nearest neighbor graph clustering method. The efficiency and effectiveness of the FAST and Chameleon algorithms are evaluated through an empirical study. Extensive experiments are carried out to compare Chameleon, FAST and several representative feature selection algorithms, namely, FCBF, ReliefF, CFS, Consist, and FOCUS-SF, with respect to four types of well-known classifiers.

Key Words: Index Terms - Feature subset selection, filter method, feature clustering, graph-based clustering, Tree-based Clustering.

I. INTRODUCTION

Feature selection, also known as variable selection, attribute selection or variable subset selection, is the process of selecting a subset of relevant features for use in model construction. The central assumption when using a feature selection technique is that the data contains many redundant or irrelevant features. Redundant features are those which provide no more information than the currently selected features, and irrelevant features provide no useful information in any context. Feature selection techniques are a subset of the more general field of feature extraction. Feature extraction creates new features from functions of the original features, whereas feature selection returns a subset of the features. Feature selection techniques are often used in domains where there are many features and comparatively few samples (or data points).

A feature selection algorithm can be seen as the combination of a search technique for proposing new feature subsets, along with an evaluation measure which scores the different feature subsets. The simplest algorithm is to test each possible subset of features finding the one which minimizes the error rate. This is an exhaustive search of the space, and is computationally intractable for all but the smallest of feature sets. The choice of evaluation metric heavily influences the algorithm.

The majority of real-world classification problems require supervised learning where the underlying class probabilities and class-conditional probabilities are unknown, and each instance is associated with a class label. In real-world situations . Therefore, many candidate features are introduced to better represent the domain. Unfortunately many of these are either partially or completely irrelevant/redundant to the target concept. A relevant feature is neither irrelevant nor redundant to the target concept; an irrelevant feature does not affect the target concept in any way, and a redundant feature does not add anything new to the target concept. In many applications, the size of a dataset is so large that learning might not work as well before removing these unwanted features.

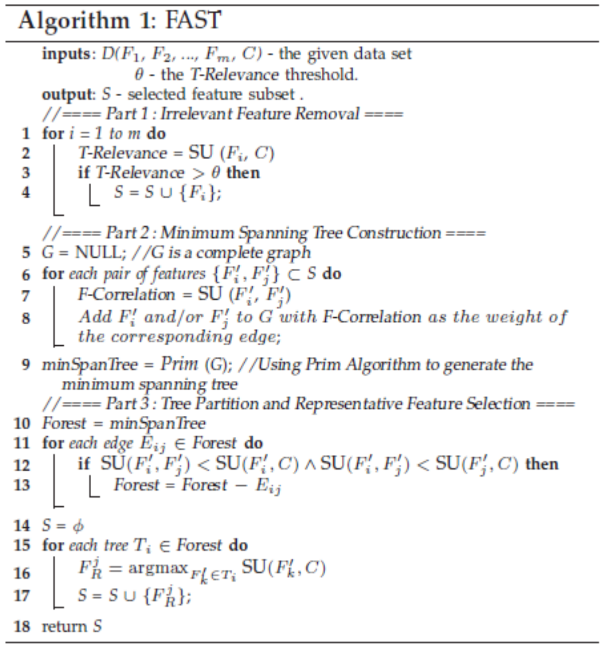
This helps in getting a better insight into the underlying concept of a real-world classification problem. Feature selection methods try to pick a subset of features that are relevant to the target concept. Feature selection is defined by many authors by looking at it from various angles. But as expected, many of those are similar in intuition and/or content. So in this paper we discuss about the feature subset algorithms FAST and Chameleon classification algorithms.

And following sections are 2. Existing system of FAST algorithms 3. Proposed System of Chameleon classification 4.Experment Results 5.Peformances 6.Conclusion

II. EXISTING SYSTEM

In this section we describe feature selection sub set process generation in recent application management with specified result accumulation. Based on the methodology presented before, we develop an algorithm, named FCBF(Fast Correlation-Based Filter). As in Figure, given a data set with
N features and a class C , the algorithm finds a set of predominant features S best for the class concept. It consists of two major parts. In the first part (line 2-),
it calculates the SU value for each feature, selects relevant features into S0 list based on the predefined threshold ± , and orders them in descending order according to their SU values. In the second part (line 8-20), it further processes the ordered list S0 list to remove redundant features and only keeps predominant ones among all the selected relevant features. According to Heuristic 1, a feature Fp that has already been determined to be a predominant feature can always be used to filter out other features that are ranked lower than Fp and have Fp as one of its redundant peers. The iteration starts from the Ørst element (Heuristic 3) in S 0 list (line 8) and continues as follows. For all the remaining features (from the one right next to F p to the last one in S 0 list), if F p happens to be a redundant peer to a feature Fq Fq will be removed from S0 list (Heuris- tic 2).
After one round of filtering features based on Fp , the algorithm will take the currently remaining feature right next to Fp as the new reference (line 19) to repeat the filtering process. The algorithm stops until there is no more feature to be removed from S0 list.
Feature subset selection can be viewed as the process of identifying and removing as many irrelevant and redundant features as possible. This is because irrelevant features do not contribute to the predictive accuracy and redundant features do not redound to getting a better predictor for that they provide mostly information which is already present in other feature(s). Of the many feature subset selection algorithms, some can effectively eliminate irrelevant

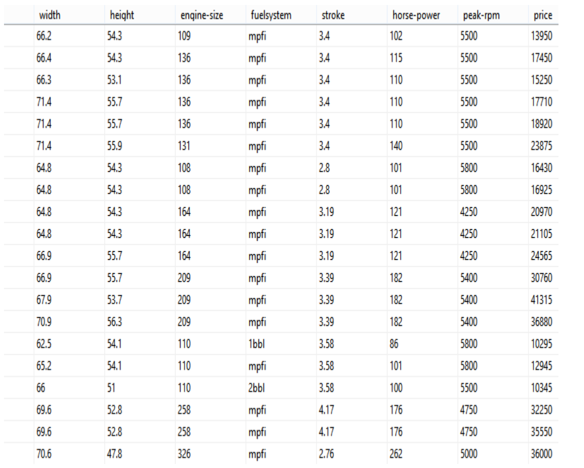
features but fail to handle redundant features yet some of others can eliminate the irrelevant while taking care of the redundant features. Our proposed FAST algorithm falls into the second group. Traditionally, feature subset selection research has focused on searching for relevant features. A well-known example is Relief which weighs each feature according to its ability to discriminate instances under different targets based on distance-based criteria function. However, Relief is ineffective at removing redundant features as two predictive but highly correlated features are likely both to be highly weighted. Relief-F extends Relief, enabling this method to work with noisy and incomplete data sets and to deal with multiclass problems, but still cannot identify redundant features.



```
Algorithm 1: FAST
    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 ====
    for $i = 1$ to $m$ do
        T-Relevance = SU $(F_i, C)$
        if T-Relevance $> \theta$ then
            $S = S \cup \{F_i\}$;
    //==== Part 2 : Minimum Spanning Tree Construction ====
    $G$ = NULL; //G is a complete graph
    for each pair of features $\{F'_i, F'_j\} \subset S$ do
        F-Correlation = SU $(F'_i, F'_j)$
        Add $F'_i$ and/or $F'_j$ to $G$ with F-Correlation as the weight of
        the corresponding edge;
    minSpanTree = Prim (G); //Using Prim Algorithm to generate the
        minimum spanning tree
    //==== Part 3 : Tree Partition and Representative Feature Selection ====
    Forest = minSpanTree
    for each edge $E_{ij} \in$ Forest do
        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
            Forest = Forest $- E_{ij}$
    $S = \phi$
    for each tree $T_i \in$ Forest do
        $F^j_R = \mathrm{argmax}_{F'_k \in T_i} \mathrm{SU}(F'_k, C)$
        $S = S \cup \{F^j_R\}$;
    return $S$
```

Figure 2: FAST Clustering algorithm specification with data set extraction.

Dataset for FAST Algorithm prepared from cars company data prepare the data sets and the store the previous data by using this previous data to give the input to FAST algorithm for the purpose of the to remove un relevant data and make decision for relevant data.



	width	height	engine-size	fuelsystem	stroke	horse-power	peak-rpm	price
	66.2	54.3	109	mpfi	3.4	102	5500	13950
	66.4	54.3	136	mpfi	3.4	115	5500	17450
	66.3	53.1	136	mpfi	3.4	110	5500	15250
	71.4	55.7	136	mpfi	3.4	110	5500	17710
	71.4	55.7	136	mpfi	3.4	110	5500	18920
	71.4	55.9	131	mpfi	3.4	140	5500	23875
	64.8	54.3	108	mpfi	2.8	101	5800	16430
	64.8	54.3	108	mpfi	2.8	101	5800	16925
	64.8	54.3	164	mpfi	3.19	121	4250	20970
	64.8	54.3	164	mpfi	3.19	121	4250	21105
	66.9	55.7	164	mpfi	3.19	121	4250	24565
	66.9	55.7	209	mpfi	3.39	182	5400	30760
	67.9	53.7	209	mpfi	3.39	182	5400	41315
	70.9	56.3	209	mpfi	3.39	182	5400	36880
	62.5	54.1	110	1bbl	3.58	86	5800	10295
	65.2	54.1	110	mpfi	3.58	101	5800	12945
	66	51	110	2bbl	3.58	100	5500	10345
	69.6	52.8	258	mpfi	4.17	176	4750	32250
	69.6	52.8	258	mpfi	4.17	176	4750	35550
	70.6	47.8	326	mpfi	2.76	262	5000	36000

Figure 2.1: Car company dataset

III. PROPOSED APPROACH

In this section we present Advanced CHAMELEON, a new clustering algorithm that overcomes the limitations of existing agglomerative hierarchical clustering algorithms discussed in Section 3. Figure 6 provides an overview of the overall approach used by Advanced CHAMELEON to find the clusters in a data

Advanced CHAMELEON operates on a sparse graph in which nodes represent data items, and weighted edges represent similarities among the data items. This sparse graph representation of the data set allows Advanced CHAMELEON to scale to large data sets and to operate successfully on data sets that are available only in similarity space [GRG+99] and not in metric spaces [GRG+99]. Advanced CHAMELEON finds the clusters in the data set by using a two phase algorithm. During the first phase, CHAMELEON uses a graph partitioning algorithm to cluster the data items into a large number of relatively small sub-clusters. During the second phase, it uses an agglomerative hierarchical clustering algorithm to find the genuine clusters by repeatedly combining together these sub-clusters.

The key feature of CHAMELEON's agglomerative hierarchical clustering algorithm is that it determines the pair of most similar sub-clusters by taking into account both the inter-connectivity as well as the closeness of the clusters; and thus it overcomes the limitations discussed .that result from using only one of them. Furthermore, CHAMELEON uses a novel approach to model the degree of inter-connectivity and closeness between each pair of clusters that takes into account the internal characteristics of the clusters themselves. Thus, it does not depend on a static user supplied model, and can automatically adapt to the internal characteristics of the clusters being merged. In the rest of this section we provide details on how to model the data set, how to dynamically model the similarity between the clusters by computing their relative inter-connectivity and relative closeness, how graph partitioning is used to obtain the initial fine grain clustering solution, and how the relative inter-connectivity and relative closeness are used to repeatedly combine together the sub-clusters in a hierarchical fashion.

Algorithm:

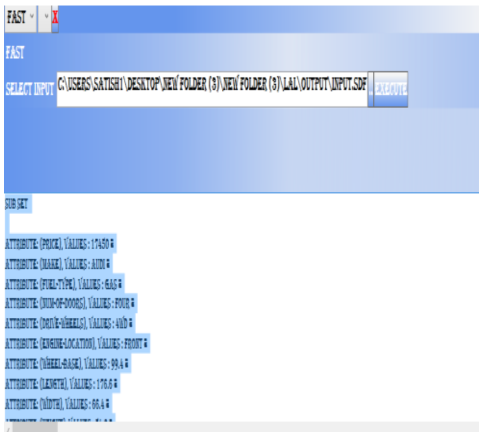
```
1void heapsort (array_of_nos, int n)
2 {
3 buildHp(array_of_nos,n);
4 shrinkHp(array_of_nos,n);
5 }
6 void buildHp (array_of_nos,n)
7 {
8loop the three steps bellow till all nodes are checked;
9 chld = I - 1;
10 prnt = (chld - 1) / 2;
11 make maximum of children as parents
12 }
13 void shinkhp(array_of_nos,n)
14 {
15 //here each thread is assigned to a particular parent node
16 prnt=0;
//start from root
17 compare left and right child and make maximum as parent;
18 take the max heap from each thread thereby getting each parent node ;
19 i.e the nodes having right and left child ;
20 knowing the position of these set of nodes construct others;
21 }
22 levelorder()
23 {
24 traverse heap in level - order by dividing these levels to threads;
25 connect only siblings to form a graph ;
26 }
```

Parallel K - NN clustering with heap sorting algorithm Pseudo code of parallel merging algorithms for final clusters

```
1 RI – Relative inter connectivity 2 RC
Relative Closeness 3 α - user defined parameter 4 β
– RI x RC 5 th – threshold value to take merging
decision
6 n be number of clusters to be merge
7 Algorithm : 8
for i=0 ... n // i and j are used for clusters
a . for j=i+1 ... n I . Assign task to work pool merge
(i,j); ii .
End for // iteration
```

IV. EXPERIMENTAL ANALAYSIS

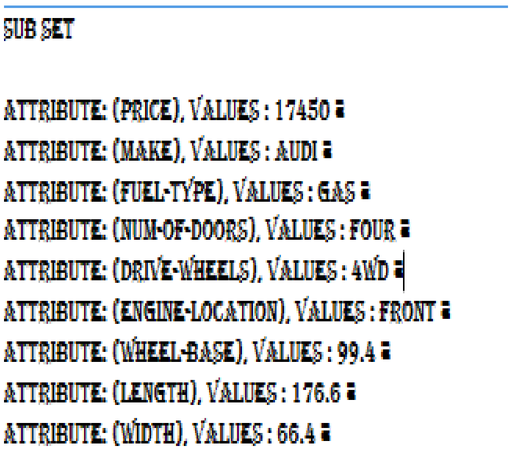
The quality of data partitions generated by this technique is assessed against those created by different categorical data clustering algorithms and cluster ensemble techniques. By knowing the result we cannot estimates the performances of the algorithm that why we apply the data sets on both algorithm get the results and compares the algorithm on the basis on time complexity and space complexity.



```
FAST
SELECT INPUT C:\USERS\SATISH1\DESKTOP\NEW FOLDER (3)\NEW FOLDER (3)\LAL\OUTPUT\INPUT.SDF  EXECUTE
SUB SET

ATTRIBUTE: (PRICE), VALUES : 17450
ATTRIBUTE: (MAKE), VALUES : AUDI
ATTRIBUTE: (FUEL-TYPE), VALUES : GAS
ATTRIBUTE: (NUM-OF-DOORS), VALUES : FOUR
ATTRIBUTE: (DRIVE-WHEELS), VALUES : 4WD
ATTRIBUTE: (ENGINE-LOCATION), VALUES : FRONT
ATTRIBUTE: (WHEEL-BASE), VALUES : 99.4
ATTRIBUTE: (LENGTH), VALUES : 176.6
ATTRIBUTE: (WIDTH), VALUES : 66.4
```

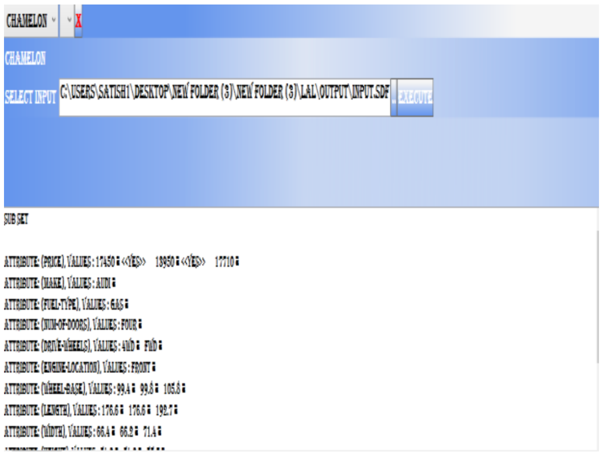
Figure 4.1:FAST Clustering algorithm results



```
SUB SET

ATTRIBUTE: (PRICE), VALUES : 17450
ATTRIBUTE: (MAKE), VALUES : AUDI
ATTRIBUTE: (FUEL-TYPE), VALUES : GAS
ATTRIBUTE: (NUM-OF-DOORS), VALUES : FOUR
ATTRIBUTE: (DRIVE-WHEELS), VALUES : 4WD
ATTRIBUTE: (ENGINE-LOCATION), VALUES : FRONT
ATTRIBUTE: (WHEEL-BASE), VALUES : 99.4
ATTRIBUTE: (LENGTH), VALUES : 176.6
ATTRIBUTE: (WIDTH), VALUES : 66.4
```

Figure 4.2:FAST Clustering algorithm results remove the un relevant data sets



```
CHAMELON
SELECT INPUT C:\USERS\SATISH1\DESKTOP\NEW FOLDER (3)\NEW FOLDER (3)\LAL\OUTPUT\INPUT.SDF  EXECUTE
SUB SET

ATTRIBUTE: (PRICE), VALUES : 17450 <<YES>>   13950 <<YES>>   17710
ATTRIBUTE: (MAKE), VALUES : AUDI
ATTRIBUTE: (FUEL-TYPE), VALUES : GAS
ATTRIBUTE: (NUM-OF-DOORS), VALUES : FOUR
ATTRIBUTE: (DRIVE-WHEELS), VALUES : 4WD  FWD
ATTRIBUTE: (ENGINE-LOCATION), VALUES : FRONT
ATTRIBUTE: (WHEEL-BASE), VALUES : 99.4  99.8  105.8
ATTRIBUTE: (LENGTH), VALUES : 176.6  176.6  192.7
ATTRIBUTE: (WIDTH), VALUES : 66.4  66.2  71.4
```

Figure 4.3: **Advanced CHAMELEON** sHierarchical model algorithm results

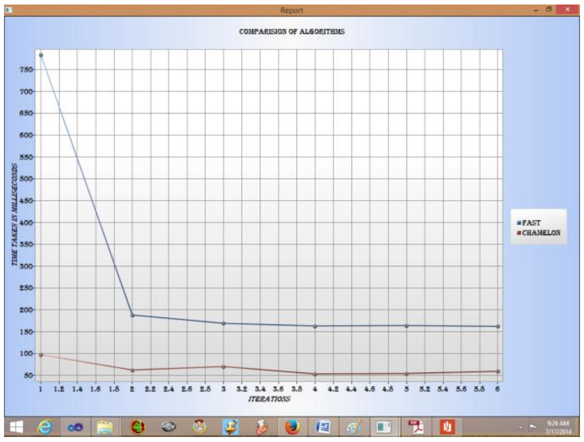


Figure 4.4:Time Taken for **Advanced CHAMELEON** and FAST algorithms.

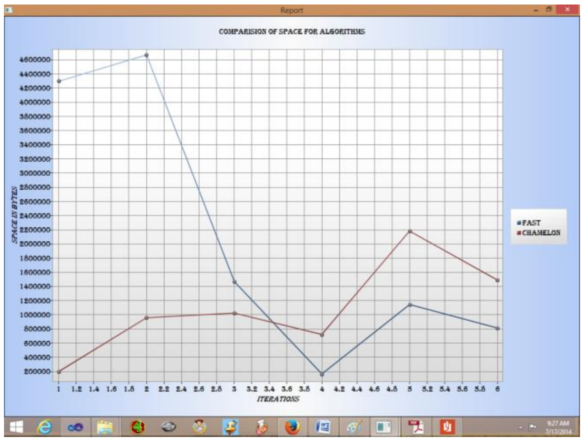


Figure 4.5:Space Taken for **Advanced CHAMELEON** and FAST algorithms.

V. CONCLUSION

We use the used dataset of car company data set by the users who are mostly used the preferable choices are users . and make it as a data sets and applied on both FAST clustering algorithm and CHAMELEON hierarchical model algorithm and both are used for to remove the irrelevant data and make it as relevant data usage preparing for the clustering data sets . in this paper we are compare the performance by using of time complexity the CHAMELEON algorithm give the best results comparative FAST clustering algorithm. And FAST Clustering algorithm give the best results for space occupation comparative to the CHAMELEON algorithm. So finally on the basis of time complexity CHAMELEON algorithm advanced algorithm for to remove irrelevant data sets and proved by the results

VI. REFERENCES

[1] Qinbao Song, Jingjie Ni and Guangtao Wang "A Fast Clustering-Based Feature Subset Selection Algorithm for High Dimensional Data".

[2] Almuallim H. and Dietterich T.G., Learning boolean concepts in the presence of many irrelevant features, Artificial Intelligence, 69(1-2), pp 279- 305, 1994.

[3] Arauzo-Azofra A., Benitez J.M. and Castro J.L., A feature set measure based on relief, In Proceedings of the fifth international conference on Recent Advances in Soft Computing, pp 104-109, 2004.

[4] Baker L.D. and McCallum A.K., Distributional clustering of words for text classification, In Proceedings of the 21st Annual international ACM SIGIR Conference on Research and Development in information Retrieval, pp 96- 103, 1998.

[5] Battiti R., Using mutual information for selecting features in supervised neural net learning, IEEE Transactions on Neural Networks, 5(4), pp 537-550, 1994.

[6] Bell D.A. and Wang, H., A formalism for relevance and its application in feature subset selection, Machine Learning, 41(2), pp 175-195, 2000.

[7] Biesiada J. and Duch W., Features election for high-dimensionaldatała Pearson redundancy based filter, AdvancesinSoftComputing, 45, pp 242C249, 2008.

[8] Butterworth R., Piatetsky-Shapiro G. and Simovici D.A., On Feature Se- lection through Clustering, In Proceedings of the Fifth IEEE international Conference on Data Mining, pp 581-584, 2005.
[9] Cardie, C., Using decision trees to improve case-based learning, In Pro- ceedings of Tenth International Conference on Machine Learning, pp 25-32, 1993.
[10] Chanda P., Cho Y., Zhang A. and Ramanathan M., Mining of Attribute Interactions Using Information Theoretic Metrics, In Proceedings of IEEE international Conference on Data Mining Workshops, pp 350-355, 2009