Comparision of Classifiers Accuracies from FAVF and NOFI for Categorical Data

D. Lakshmi Sreenivasa Reddy, B. Raveendra Babu, A. Govardhan, A. Kalpana and Mudimbi Krishna Murthy

Abstract Outlier analysis is an important task in data science. Specifically finding outliers in categorical data is a tough task. To build an accurate Classifier, it is needed to eliminate exact number of outliers from the data. If less number of outliers is found, the obstacles will remain in the original data. An accurate classifier cannot be built on this data. Similarly if more number of outliers is found and eliminated, some original records may be missed. From this data too an accurate classifier cannot be built. So it is needed to eliminate the exact number of outliers while modeling a classifier. Since the data is categorical, in classification modeling, most infrequent records are treated as outliers. These infrequent objects disturb the data in modeling classifier. But how many outliers needed to be found is a problem. This paper presents the new approach normally distributed Outlier factor by infrequency (NOFI) to improve the Classifier accuracy. In modeling a classifier for categorical data, high frequent records are most useful and infrequent records are most useless. So the infrequent records are obstacles in modeling the classifier. There are many effective approaches to detect outliers for numerical data. But for categorical datasets there are few numbers of methods exists. The experiments are conducted for this new method has been applied on bank dataset which is taken from UCI ML Repository. This approach is not needed any input of k, the required number of

D. Lakshmi Sreenivasa Reddy (✉)
Department of CSE, RISE Gandhi Group of Institutions, Ongole, India
e-mail: urdlsreddy@yahoo.com

B. Raveendra Babu
Department of CSE, VNRVJIET, Hyderabad, India
e-mail: rboghapathi@yahoo.com

A. Govardhan
SIT, JNTUH, Hyderabad, India
e-mail: govardhan_cse@yahoo.co.in

A. Kalpana · M. Krishna Murthy
Department of MCA, RISE Gandhi Group of Institutions, Ongole, India
e-mail: Kalpana.amirisetti@gmail.com

M. Krishna Murthy
e-mail: krishnamudimbi@gmail.com

J.K. Mandal et al. (eds.), *Information Systems Design and Intelligent Applications*,
Advances in Intelligent Systems and Computing 339,
DOI 10.1007/978-81-322-2250-7_12

outliers. NOFI would find number of outliers automatically using infrequency of all possible combinations framed from attribute values included in any record.

Keywords Outlier analysis · Categorical · AVF score · FAVF score · OFI score · NOFI score

1 Introduction

Outlier analysis is an important task in data analysis. Without eliminating these outliers a correct classifier cannot be built. Some of the applications of outlier analysis are like credit card fraud detection, intrusion detection of networks, medical diagnosis analysis, and business decision analysis. In this approach a simple method for classifier accuracy is presented. AVF [1] method is one of the efficient methods to detect outliers in categorical data. In this method, it calculates frequency of each attribute value in each record and finds their average AVF score for each record. But the major problem in this is how many outliers need to be selected from the dataset. In this method we need to give an input for selecting number of outliers. We don't know how these are reliable outliers. This problem is solved by NAVF [2]. By this method the reliable number of outliers is selected automatically. After deleting these outliers automatically by NAVF, the classifier has been built on the remaining data. In another approach FAVF [3] has been built for the same purpose. This method also finds outliers automatically. But the reliability of outliers found by FAVF is less when compared with NAVF. In another approach FPOF [4] for categorical data is also used frequent patterns which are generated from Apriori algorithm [5]. FPOF calculates frequent pattern item sets from each record in data set. From these frequencies it calculates FPOF score and finds k outliers as the least k-FPOF scores. All these methods are used average frequency of each attribute value. This method is so complex because it needs generation of frequent patterns and also needs a threshold value 'σ' and input 'k' as the number of k outliers need to be eliminated. Another method based on frequency is Greedy [6].

2 Some of Existing Approaches for Categorical Data

2.1 Greedy Algorithm

This method finds the entropy of data set when a record is included in dataset. This method is finds out records from which the datasets give more entropy. Assume that the dataset is denoted by 'D' with 'm' attributes A1, A2 … Am and D(Aj) is the domain of distinct values in the attribute Aj, then the entropy of single attribute Aj is calculated by the E(Aj), it is calculated by the below equation.

$$E(A_j) = \sum_{x \in D(A_j)} p(x) \log_2(p(x)) \quad (1)$$

All the attributes are independent to each other by nature, Entropy of the entire dataset D = {A1, A2 … Am} is equal to the sum of the entropies of attributes, and is defined as follows.

$$E(A_1, A_2, A_3 \ldots A_j) = E(A_1) + E(A_2) + E(A_3) + \cdots E(A_j) \quad (2)$$

Entropy of dataset is calculated each time when one record is selected aside. Among all entropies, k-least entropies are selected. The corresponding records for these least entropies are treated as top k-outliers in this dataset. The complexity of Greedy algorithm is O(k * n * m * v), where k-is the required number of outliers, n is the number of records in the dataset D, m is the number of attributes in D, and v is the number of distinct attribute values per attribute. The terminology used in this paper is given in Table 1.

Table 1 Terminology

Term	Description
DB	Database
K	Target number of outliers
N	Number of objects in dataset
M	Number of attributes in dataset
Xi	ith object in dataset ranging from 1 to n
Aj	jth attribute ranging from 1 to m
D(Aj)	Domain of distinct values of jth attribute
Xij	Cell value in ith object which takes from domain dj of jth attribute Aj
D	Dataset
V	Set of all distinct values in dataset D
P	Set of all combinations of distinct attribute values, where each attribute occurs only once in any combination
I	Item set
F	Frequent item set
IF	Infrequent item set
f(xij)	Frequency of xij value
FS(xi)	Set of frequent Item sets of xi object
IFS(xi)	Set of infrequent Item sets of xi object
Minsup	Minimum support of frequent item set
Support(I)	Support of item set I
OFI	Outlier factor by infrequency score
NOFI	Normally distributed outlier factor by infrequency score

2.2 Attribute Value Frequency (AVF) Algorithm

AVF approach is simpler and faster approach to find outliers. It needs only one scan of entire database and it does not take more space. The AVF method is defined as below. Let us take "x_i" as an object in a dataset. AVF score of this object is defined as below.

$$AVF(xi) = \sum_{j=1}^{m} f(xij) \tag{3}$$

This method also needed input 'k' as the number of outliers to be eliminated. This approach gives us more accuracy and low complexity.

2.3 Fuzzy Distributed Attribute Value Frequency (FAVF)

This method [3] depends on AVF score. It finds the optimal number of outliers automatically. Outliers found by FAVF are more in number when compared with the number of outliers found by NAVF. FAVF model tried to convert the ambiguity left by NAVF. FAVF uses the S-Fuzzy function and finds three seeds based on AVF scores of the objects. These three seeds are used to distribute the entire dataset. Fuzzy seeds and Fuzzy score are given below. This FAVF method also finds the optimal number of outliers automatically from the original database.

$$b = mean(fi) \tag{4}$$

$$a = \begin{cases} b - 3 * STD(fi) & if \max(fi) > 3 * STD(fi) \\ b - 2 * STD(fi) & if \max(fi) > 2 * STD(fi) \\ b - STD(fi) & if\ otherwise \end{cases} \tag{5}$$

$$c = \begin{cases} b + 3 * STD(fi) & if \max(fi) > 3 * STD(fi) \\ b + 2 * STD(fi) & if \max(fi) > 2 * STD(fi) \\ b & otherwise \end{cases} \tag{6}$$

$$Fuzzyscore(xi) = \begin{cases} 0 & if\ fi < a \\ 2\left\{\frac{fi-a}{c-a}\right\}^2 & if\ a \le fi \le b \\ 1 - 2\left\{\frac{fi-a}{c-a}\right\}^2 & if\ b \le fi \le c \\ 1 & if\ fi > c \end{cases} \tag{7}$$

where

"fi" f(xi) = AVF score of ith record.

Max (fi) Maximum of AVF scores in the dataset.

a Mean of AVF scores in the dataset.
STD (fi) Standard Deviation of all AVF scores in the dataset.

2.4 Outlier Factor by Infrequency (OFI)

This approach OFI [7] calculates the outlier factor based on infrequency of each infrequent itemsets involved in a record which is generated by Apriori algorithm [5]. OFI score is calculated by the below formula.

$$= \beta(xi) = \sum_{j=1}^{m} \frac{|DB|}{1 + f(IFS(xi))} \tag{8}$$

Here,

Let "xi" is the record of a database DB,
Aj = Attribute, where j takes the values from 1 to m,
IF = Infrequent Itemset,
IFS (xi) = Set of infrequent Itemsets of "xi",
xij = ith value in jth attribute
$|DB|$ is length of Dataset

OFI score of each record is calculated by the above Eq. (8). The outliers selected by highest OFI score records. This method is also needed an input value "k" to get k-outliers and a threshold value to decide infrequent itemsets. Accuracy of finding outliers is more when compared with BAD score and AVF score methods, but the complexity is more.

2.5 Proposed Optimization Method: Normally Distributed Outlier Factor by Infrequency (NOFI)

OFI method finds k-number of outliers based on the input 'k'. NOFI calculates reliable number of outliers automatically based on the threshold value. This threshold value is calculated as below.

$$\mu_{(\beta)} = \frac{1}{|DB|} \sum_{i=1}^{n} \sum_{j=1}^{m} \frac{|DB|}{1 + Frequency(\inf requenct\_Itemsets\_of\_record)} \tag{9}$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{m} \frac{1}{1 + Frequency(\inf\ requenct\_Itemsets\_of\_record)} \tag{10}$$

$$\sigma_{(\beta)} = \sum_{\mathrm{i}=1}^{\mathrm{n}} \sqrt{(OFI(xi) - \mu_{(\mathrm{OFI})})^2} \tag{11}$$

$$\text{NOFI score}_{(\beta)} = \tau = \mu_{(\beta)} + 3\sigma_{(\beta)} \tag{12}$$

If Xi is said to be an outlier in dataset DB, its OFI score must satisfies the below condition.

$$\text{if } \beta(\text{Xi}) \begin{array}{ll} \geq & \tau, \quad \text{Xi is called Outlier} \quad \forall\ \text{i} = 1 \text{ to n} \\ < & \tau, \quad \text{Xi is called inlier} \quad \forall\ \text{i} = 1 \text{ to n} \end{array} \tag{13}$$

This proposed optimization method also finds optimal number of outliers automatically in perspective of classifier building.

3 Experimental Results and Discussion

The experiments are conducted for this method on bank data with 45221 instances is taken from UCI ML Repository [8]. Only seven categorical attributes are considered for experiments. This method is implemented on PL-SQL. Bank data with 7 attributes and 46 values are considered for experiments. The attributes considered for this experiments are "Job", "Marital status", "Education", "loan", "housing", "contact", 'Y' = "Class label attribute". Bank data has been divided into two parts, first part of data is considered with "Yes" Class label and the number of records are 5,299 in this part and second part with "no" class label has 39,922 records. This petitioning of the Bank dataset has been achieved by using the Clementine tool. The "yes" label records are considered as outliers in this experiment. From the first part, for each 10 records of class "yes" one record is selected by using 10-1 random sampling technique and mixed up with "no" class label records. The mixed up records are 40,427. Both FAVF and NOFI methods have been applied on these mixtures of records to eliminate outliers. After eliminating outliers automatically by NOFI, this method has found 39,899 records as inliers. The total outliers are found by NOFI are 528. Similarly FAVF has been found 332 outliers automatically and eliminated. FAVF has been found 40,095 inliers. After eliminating outliers by both methods classifiers are modeled. Clementine tool has been used to model different classifiers. The classifiers modeled by NOFI show more accuracy than FAVF and direct.

While modeling the classifiers for direct data including outliers the lift values are given in the respective column in Table 2. Each classifier is also used different number of variables. The accuracies of classifiers are also given in Table 2.

Table 2 Comparison of accuracies of classifiers modeled on original data (including outliers)

Model	Lift	Number of fields used	Accuracy achieved (%)
DL	1.721	4	58.435
LR	2.02	6	98.695
NN	1.931	6	98.695
CHAID	1.954	5	98.695

While modeling the classifiers after deleting outlier by FAVF the lift values are given in the respective column in Table 3. Each classifier is also used different number of variables. The accuracies of classifiers are also given in Table 3.

Similarly the results are given in Table 4 for different classifiers modeled by NOFI (Figs. 1, 2, 3, 4, 5; Table 5) .

The Decision logic (DL) classifier gave 58.435 % of accuracy when tested on test data after trained it on the original data. The same classifier gave 38.001 and 35.559 % accuracy respectively, when the classifier trained on the cleaned data by FAVF and NOFI. When the Neural Networks (NN) classifier is modeled on the original data (without cleaning), it has given 98.685 % accuracy. NN has given 98.873 and 99.068 % accuracy respectively when it is developed on cleaned data cleaned by FAVF and NOFI. Linear Regression (LR) gave the 98.695 % accuracy on the test data when tested it on original data (without cleaned) and 98.873 % accuracy on cleaned data cleaned by FAVF and 99.068 % for NOFI. Similarly CHAID Classifiers gave the same results as LR respectively for original data, cleaned data by FAVF and NOFI.

Table 3 Comparison of accuracies of classifiers modeled by FAVF

Model	Lift	Number of fields used	Accuracy achieved (%)
DL	1.742	3	38.001
LR	2.094	6	98.873
NN	2.075	6	98.873
CHAID	1.991	5	98.873

Table 4 Comparison of accuracies of classifiers modeled by NOFI

Model	Lift	Number of fields used	Accuracy achieved
DL	1.788	4	35.559
LR	2.5	6	99.068
NN	2.486	6	99.068
CHAID	2.34	5	99.068

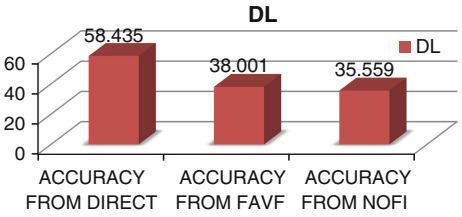

Fig. 1 Comparison of accuracy of classifier DL modeled by direct, FAVF and NOFI

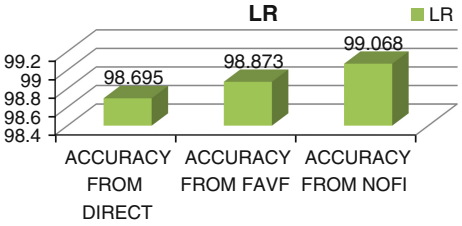

Fig. 2 Comparison of accuracy of classifier LR modeled by direct, FAVF and NOFI

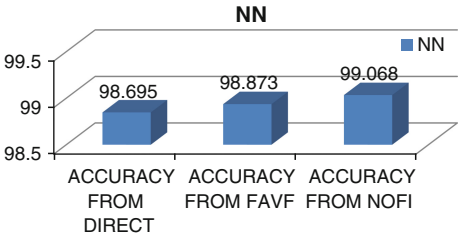

Fig. 3 Comparison of accuracy of classifier NN modeled by direct, FAVF and NOFI

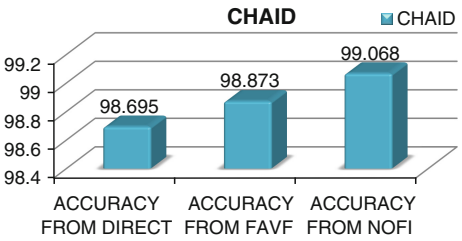

Fig. 4 Comparison of accuracy of classifier CHAID modeled by direct, FAVF and NOFI

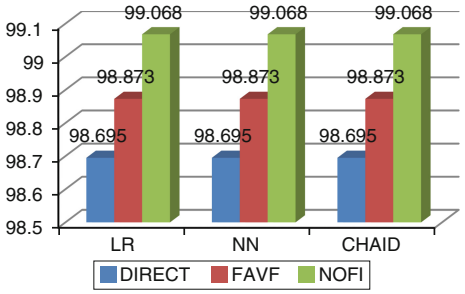

Fig. 5 Comparison of accuracy of classifiers accuracies modeled by direct, FAVF and NOFI

Table 5 Comparison of accuracy of classifiers accuracies modeled by direct, FAVF and NOFI

9i8Model	Accuracies achieved		
	Direct	FAVF	NOFI
LR	98.695	98.873	99.068
NN	98.695	98.873	99.068
CHAID	98.695	98.873	99.068

4 Conclusion and Future Work

This new method has been achieved good results when compared with FAVF method and Direct. NOFI is one of the better methods when compared with others. In future we will compare the precisions and recalls by both methods on different datasets.

References

1. Koufakou, A., Georgiopoulos, M.: A fast outlier detection strategy for distributed high-dimensional data sets with mixed attributes. Data Min. Knowl. Disc **20**, 259–289 (2010)
2. Lakshmi Sreenivasa Reddy, D., Raveendra Babu, B., Govardhan, A.: Outlier analysis of categorical data using NAVF. Informatica Economica **17**(1), 5–13 (2013)
3. Lakshmi Sreenivasa Reddy, D., Raveendra Babu, B.: Outlier analysis of categorical data using FuzzyAVF. Presented at IEEE International Conference ICCPCT-2013, pp. 1259–1263
4. He, Z., Xu, X., Huang, J., Deng, S.: FP-Outlier: frequent pattern based outlier detection. Comput. Sci. Inf. Syst. (ComSIS'05) **2**(1), 103–118 (2005)
5. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: Proceedings International Conference on Very Large Data Bases, pp. 487–499 (1994)
6. He, Z., Deng, S., Xu, X.: A fast greedy algorithm for outlier mining. In: Proceedings of PAKDD (2006)
7. Lakshmi Sreenivasa Reddy, D., Raveendra Babu, B.: Efficient model to find outliers in categorical data using outlier factor by infrequency. Presented at IEEE International Conference ICCPCT-2014, pp. 1324–1328
8. Frank, A., Asuncion, A.: UCI Machine Learning Repository. University of California, School of Information and Computer Science, Irvine, CA. http://archive.ics.uci.edu/ml (2010)