J Supercomput (2016) 72:2815–2831
DOI 10.1007/s11227-016-1624-z

Highway traffic accident prediction using VDS big data analysis

Seong-hun Park[1] · **Sung-min Kim**[1] · **Young-guk Ha**[2]

Published online: 20 January 2016

Abstract In modern society, accidents on the roads are one of the most life-threatening dangers to humans. Traffic accidents that cause a lot of damages are occurring all over the places. The most effective solution to these types of accidents can be to predict future accidents in advance, giving drivers chances to avoid the dangers or reduce the damage by responding quickly. Predicting accidents on the road can be achieved using classification analysis, a data mining procedure requiring enough data to build a learning model. However, building such a predicting system involves several problems. It requires many hardware resources to collect and analyze traffic data for predicting traffic accidents since the data are extremely large. Furthermore, the size of data related to traffic accidents is less than that not related to traffic accidents; the amounts of the two classes (classes to be predicted and other classes) of data differ and are thus imbalanced. The purpose of this paper is to build a predicting model that can resolve all these problems. This paper suggests using the Hadoop framework to process and analyze big traffic data efficiently and a sampling method to resolve the problem of data imbalance. Based on this, the predicting system first preprocesses the big traffic data and analyzes it to create data for the learning system. The imbalance of created data is corrected using a sampling method. To improve the predicting accuracy, corrected data are classified into several groups, to which classification analysis is applied.

✉ Young-guk Ha
ygha@konkuk.ac.kr

Seong-hun Park
wolfire@konkuk.ac.kr

Sung-min Kim
allmax75@konkuk.ac.kr

1 Konkuk University, 1007, Newmillenium Hall, Hwayang-dong, Gwangjin-gu, Seoul, Korea

2 Konkuk University, 903, Newmillenium Hall, Hwayang-dong, Gwangjin-gu, Seoul, Korea

Keywords Accident prediction · Big data inference · Imbalance data · MapReduce · Classification

1 Introduction

In modern society, everyday life is intimately concerned with transportation creating lots of issues on it. Among the issues, one of the most important issues is about traffic accidents, and it is important to avoid those accidents or reduce damage from them. Predicting possible traffic accidents can be a solution for those goals.

To predict traffic accidents, we can use video image from cameras on the road [1], or various traffic data to analyze [2,3]. We are concerned with analyzing traffic data so that we can predict possible traffic accidents. The prediction through data analysis is composed of mainly the classification analysis through learning from past data in data mining. The classification analysis learns the training data set and creates a standard predicting model for a prediction result. Based on these, the model predicts the result to imitate the existing content.

Making a new predicting model involves a number of problems, the first of which is imbalance data. Imbalance data means data in which there is a considerable difference between the observed sizes from one data set. To solve this problem, sampling techniques can be used, of which there are two types: under-sampling and over-sampling [4]. Under-sampling involves using all the observation values in a small class and using part of the observation value in a large class. Over-sampling involves using all observation values in a large class and increasing the size of the observation value in the small class and then using this value. Under-sampling calculates the use of lost data While such sampling techniques can speed up processing data, losing the reliability of data cannot be avoided. On the other hand, over-sampling utilizes all data, but requests more resources for processing additional data [5].

The second problem involves data processing to create a training data set. The training data set has a set of multiple features that can affect the prediction result. Therefore, processing the dataset for the many different types of data is time-consuming and more resources are needed, depending on the data size.

We aim to solve these two problems and do efficiently prediction processing using the MapReduce algorithm. So, this paper presents processing steps of a parallel classification based on MapReduce and shows the validity of these steps.

MapReduce is a programming model for processing large data sets with a parallel, distributed algorithm on a cluster, which processes big data efficiently [6]. Efficient processing of big data facilitates the analysis of big data. Therefore, in many studies on big text data processing, MapReduce platforms have been used [7]. Using MapReduce, processing performance for generating a training data set and conducting classification has improved, which can resolve problems about a high processing overhead and low processing speed [8].

Our proposed processing consists of five steps. After preprocessing target data sets, the data sets are combined and training data sets are created. The over-sampling technique is used to solve the imbalance data problem in the training dataset. To generate the training model, the over-sampling technique carries out a classification

based on training datasets. Finally, it verifies the accuracy of the result for processing. The detail explain in Sect. 3.

The purpose of this paper is to build a prediction model and predict traffic accidents which happened on expressways since they are caused by a smaller number of variables than accidents in downtowns. We used data generated on the Gyeongbu Expressway which connects Seoul and Busan to build a learning model for predicting traffic accidents.

We have experimented proposed processing steps. The purpose of experiment is to build a prediction model and predict traffic accidents which happened on expressways since they are caused by a smaller number of variables than accidents in downtowns. We used data generated on the Gyeongbu Expressway which connects Seoul and Busan to build a learning model for predicting traffic accidents. We selected accidents and non-accidents data related to traffic in a large amount of imbalance data, and the size of the data is about 300 GB. To solve the problems we uttered above, we used Hadoop, which is a framework based on MapReduce algorithm, and we showed the results of the prediction of traffic accidents.

The outline of this paper is organized as follows: Sect. 2 explains Hadoop framework and data mining issues, and Sect. 3 explains the proposed processing steps. In Sect. 4, we show the experiments to each step for accident prediction and in Sect. 5 the conclusions and future works are given.

2 Related work

2.1 Hadoop framework

Hadoop architecture consists of a Hadoop distributed file system (HDFS), jobs, and tasks, as illustrated in Fig. 1 [9]. HDFS is a distributed, scalable, and portable file system written in Java for the Hadoop framework. Each node in a Hadoop instance typically has a single namenode; a cluster of datanodes forms the HDFS cluster. This is a typical situation because each node does not require a datanode to be present. Each datanode serves up blocks of data over the network using a block protocol specific to HDFS. The file system uses the TCP/IP layer for communication; clients use RPC to communicate with each other. HDFS stores large files (an ideal file size is a multiple of 64 MB) across multiple machines. Jobs are separated into two phases: a Map phase and a Reduce phase. The execution in the Map and Reduce phases of a node is regarded as a task. The Map phase generates the input data formatted as a list of ⟨key, value⟩ pairs for the Reduce phase. The Hadoop job tracker assigns each data block to each task of the nodes. The Reduce phase receives ⟨key, value⟩ pairs that have the same key. The task of the Reduce phase is to process values with the same key. This programming model is very simple; however, this simplicity makes it easily applicable to various large-scale domains (Fig. 2).

In the latest version of Hadoop framework, yet another resource negotiator (Yarn) is in charge of managing the use of resources of nodes, which is for reducing the burden of the JobTracker the previous Hadoop framework had so that applications on HDFS can perform more efficiently, and the cluster can handle more jobs at a time. As

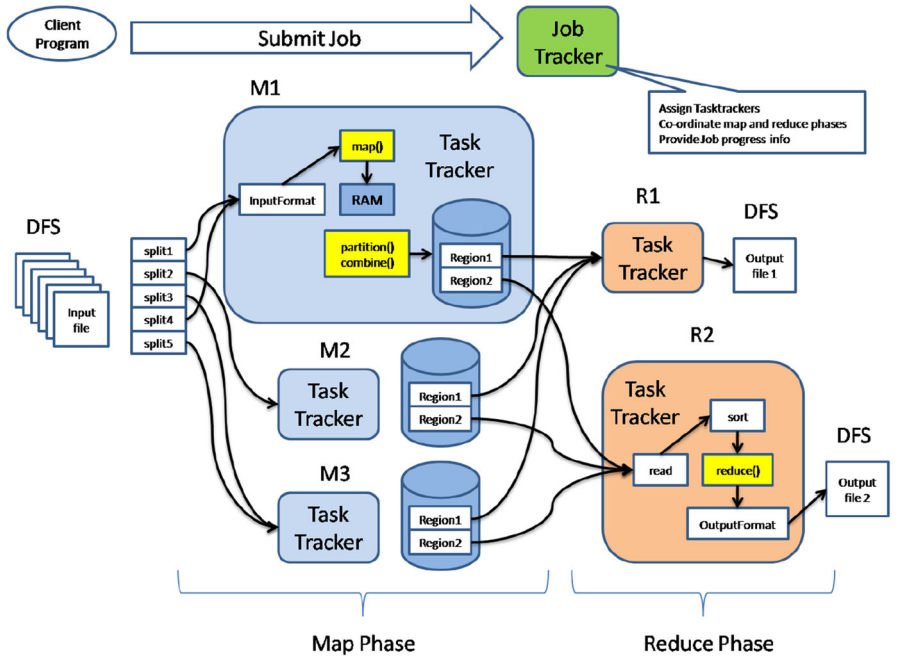

Fig. 1 Hadoop architecture

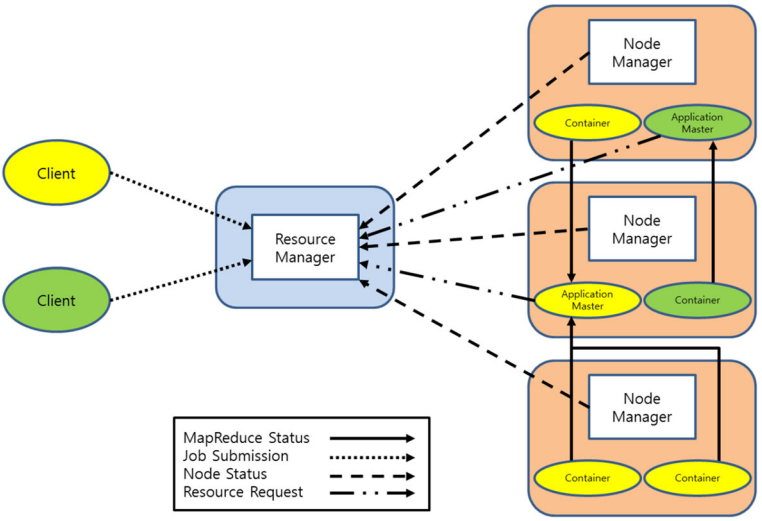

Fig. 2 Yarn architecture

a result of the overhaul in the Hadoop 2.x version, some new components emerged: The ResourceManager, the NodeManager. The ResourceManager is composed of two parts: Scheduler, ApplicationMaster. The Scheduler allocates resources to the applica-

tions. There are several kinds of schedulers according to its resource-allocating policy. The ApplicationMaster is involved in submitting jobs and negotiating resources for applications. The NodeManager runs per a machine and manages containers in it.

We used Hadoop 1.x instead of Hadoop 2.x because we placed importance on making the best of MapReduce algorithm rather than on taking advantage of Yarn's resource managing function.

2.2 Data mining issues

Processing of data mining has some commonly encountered problems like high dimensionality [10], imbalance data [11] and noise [12]. High dimensionality refers to the large abundance of attributes. If the number of attributes in a datasets is large, then it is most likely that most of these attributes are redundant or useless for building an prediction model. Better performance can be achieved if useless and redundant attributes are removed. The primary technique used to deal with this problem is known as feature selection, where a subset of the original features is selected to be used in the learning process. Different feature selection techniques have been proposed, used, and evaluated.

Data imbalance is another major problem encountered in training datasets for data mining. A dataset is imbalanced if the cases of the positive class, also called the class of interest, are outnumbered by cases of the negative class. This can result in high false negatives, mainly harming the minority class, which is the most important class. The primary approach for handling class imbalance is sampling. Sampling transforms the dataset to a more balanced one by adding or removing instances until a desired class ratio is reached.

Another common challenge to machine learning is noise. Noise refers to missing and incorrect values in a dataset. There are two types of noise: class noise and attribute noise. Class noise in particular has received a lot of attention, and is considered more severe. Much work has been done towards studying the effect of noise on learning.

2.3 Highway traffic accidents

What data can we make use of to predict highway traffic accidents by analyzing the data? Zhigang Yans paper [13] says that the causes of highway traffic accidents are human, vehicle, traffic environment and management.

Tibebe Beshah's paper [14] analyzed 14,254 accidents which occurred between May 2005 and September 2008 in Ethiopia to figure out their patterns and knowledge about them. Geetha Ramani's paper [15] analyzed data about 159,417 casualties in 157,463 accidents to figure out road accident patterns related to pedestrian characteristics. Tibebe Beshah and Geetha Ramani's models are shown below in Tables 1 and 2. Those papers tried to figure out traffic accidents patterns based on information of humans. But, it is difficult to find association between static data about humans and accidents because human behavior is hard to predict. What we focus on is the traffic environment. The traffic environment involves Road condition, weather condition, traffic condition and social factor such as an act of war. The traffic environment is

Table 1 Tibebe Beshah's model

S. no.	Attributes	Description
1	PedstrianMovem	Pedestrian movement during the accident
2	VictimHealthSt	Health condition of victim
3	VictimOccup	Occupation of victims
4	VictimAge	Age of victims
5	DrivingLicens	Driving license level of a driver
6	VictimCategory	Category of victims
7	VechileMovement	How the driver was driving the vehicle
8	DrivingExp	Driving experience of the driver
9	AccuDriVehiRelation	Relationship b/n a vehicle and a driver
10	AccuDrivEduLevel	Educational level of a driver
11	DriverAge	Age of a driver
12	DriverSex	Sex of a driver
13	AccidentResult	Whether a collision ended with injury or non-injury

Table 2 Geetha Ramani's model

S. no.	Attributes	Type
1	Accident index	Identifier
2	Casualty reference	Identifier
3	Casualty class (pedestrian)	Binary
4	Gender	Ordinal
5	Age band	Ordinal
6	Casualty severity	Ordinal
7	Car passenger	Nominal
8	Deprived	Ordinal
9	Casualty type	Nominal

described as a collection of various types of static data, which helps to create dataset from them.

Table 3 shows the proposed model by Bimal Ghimire. Bimal Ghimire's paper [16] also focused on the traffic environment to analyze traffic accidents using spatial data mining techniques. There are a large amount of recorded static data about road condition, weather condition, and traffic condition. As mentioned before, using those data takes a long time because of its large size, and the variety of the data types causes the problem of high dimensionality. To tackle this problem, we utilize a distributed framework based on Hadoop to use the traffic environment data, create training dataset and the model and perform the prediction of highway traffic accidents by analyzing the data and learning from them.

Table 3 Bimal Ghimire's model

S. no.	Attributes
1	DayOfWeek
2	RoadTypeRoadType
3	JunctionDetail
4	LightConditions
5	Weather
6	RoadSurface
7	AccidentSeverity
8	Longitude
9	Latitude
10	LocalHighway authority

3 Propose method

The classification analysis process of imbalance data prediction involves five steps, and the entire process is shown in Fig. 3. The overall system is based on Hadoop [17], with the implementation of a data preprocess, learning data creation, and over-sampling by Hive [18]. The cluster and classification analysis were operated by Mahout [19]. In this paper, a novel method to determine each operation step for classification analysis of traffic accident prediction will be explained in detail.

3.1 Preprocessing

The data preprocess was required to process the variable selection of data for training and each ideal value of the variables. The variables of data were selected according to each variable characteristic and omitted variable in the total data. The ideal value process was replaced with omitted or initial values by considering each value of the variables. In addition, the formatting was necessary to match the data.

3.2 Create training dataset and over-sampling

To build a learning data, the preprocessed data were combined, and a unit learning data model then was formed. The data were then modified to implement the final learning data. The features that affected the target variables were always considered when the learning data were created. Therefore, identification (ID) or index value was applied to use the variables of data effectively when the learning data were created.

Unfortunately, these training data contained the data unbalance. To overcome this problem, an over-sampling operation was processed to repair the data. Previous research [20] proposed a method for over-sampling the minority class, which is called synthetic minority over-sampling technique (SMOTE). It achieves its goal by creating synthetic examples using real data. It creates synthetic examples using each examples k nearest neighbor examples. To create each synthetic example, several cal-

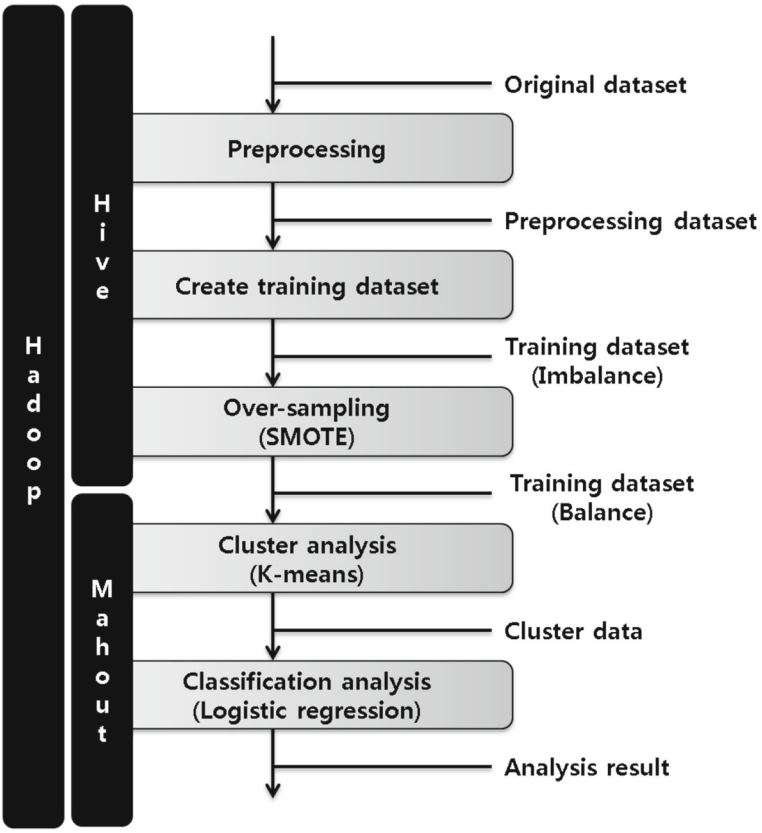

Fig. 3 Steps of proposed classification analysis process

culations are conducted: take the difference between a real examples attribute and that of a nearest neighbor of its. The difference is multiplied by a random number between 0 and 1. And it is added to the real examples attribute. These calculations are done for all attributes of every real example. The number of synthetic examples created by this method is determined by a parameter, N. If N is 200 and the number of real examples is 200, then 200 synthetic examples are created so that the number of all examples increases to 200 %.

3.3 Cluster analysis

Because of the unique characteristics of the data, a cluster classification analysis could provide high accuracy in the process because it was operated by partitioning several clusters and processed by clusters rather than the individual classification analysis method [21]. The influences of each cluster characteristic were obviously recognized when the data were clustered [22]. When the cluster analysis was operated, the numbers of the cluster were determined, and the data ratio of the cluster was confirmed. In addition, each representative value of the clusters was found.

Table 4 Confusion matrix

	Prediction	
	True	False
Actual		
True	True positive (TP)	False negative (FN)
False	False positive (FP)	True negative (TN)

3.4 Classification analysis

A classification analysis algorithm selects the created learning data built learning model. This was the same as the criterion of the categorizing data which was based on learning data. By this model, the results could be predicted by choosing the estimated target variable value when the data formed with the estimated variables was inputted [23]. The classification analysis was processed with several clusters which were created by cluster analysis. A different logistic regression method was applied to each cluster [24], and the results were collected to present a confusion matrix.

3.5 Prediction accuracy

We used the total, and target precision as reference to determine the performance. As Table 4 showed, the confusion matrix was utilized, and the equations for the results were calculated and recorded below.

$$\text{Accuracy} = \frac{(\text{TN} + \text{TP})}{(\text{TN} + \text{TP} + \text{FN} + \text{FP})} \tag{1}$$

$$\text{True positive rate} = \frac{(\text{TP})}{(\text{TP} + \text{FN})} \tag{2}$$

$$\text{True negative rate} = \frac{(\text{TN})}{(\text{TN} + \text{FP})} \tag{3}$$

It can be seen from the table a variety of information. Equation (1) indicates accuracy, while (2) and (3) refer to "true positive rate" and "true negative rate", respectively. accuracy show the validity of proposed data mining model. "True positive rate" measures the proportion of actual positives which are correctly identified as such (e.g., the percentage of sick people who are correctly identified as having the condition), and is complementary to the false negative rate. "True negative rate" measures the proportion of negatives which are correctly identified as such (e.g., the percentage of healthy people who are correctly identified as not having the condition), and is complementary to the false positive rate.

4 Experiment

We experiment our method based on MapReduce for traffic prediction using actual highway traffic data. The system was built on Hadoop and processing performance was increased using multi-node. Our systems environment is shown below in Table 5.

Table 5 Experiment environment

Hadoop environment			
Network	100 Mbps		
Hadoop version	Hadoop 1.0.4		
Nodes	NodeName node	2nd name node	Data node
	1	1	30
Hive version	Hive 0.12.0		
Mahout version	Mahout 0.8		
Node environment			
CPU	3.10 GHz quad-core		
Memory	16 GB		
Operating system	Ubuntu server 12.04 LTS (64bit)		

Table 6 Format of traffic data

Variable	Type
Record time	YYYYMMDDhhmmss
VDS ID	ID (string)
Number of line	Integer
Traffic volume	Integer
Traffic density	Real number
Average speed	Real number

4.1 Highway traffic data

The data used for the experiment is sourced from the Korea Highway Corporation. The data are text files showing traffic data created between Jan 1st, 2011 and June 30th, 2013 on the Gyeongbu line which connects Seoul with Busan. The structure of the data is explained below.

4.2 Traffic data

The traffic data are created using a vehicle detection sensor (VDS) which measures speeds of cars every 30 s and records the number of cars that run on the road. Table 6 shows the format of the data. Record time refers to the time at which the data were recorded. VDS ID refers to the ID of each loop coil. Line number refers to the particular line the information came from. Traffic volume refers to the number of cars per 30 s. Occupation rate refers to the number of cars per 30 s depending on the length of the section. Average speed refers to the average speed of cars that pass per 30 s. The sizes of all the data in each year are shown in Table 7.

Table 7 Data sizes on year and way

Year	Way	Size (GB)
2011	Seoul	62.3
	Pusan	58
2012	Seoul	62.6
	Pusan	56.6
2013 (Jun.)	Seoul	31.1
	Pusan	28.8

Table 8 Format of accident and non-accident data

Variable	Type
Record time	YYYYMMDDhhmmss
Day	Category
Position	Real number (km)
Death	Integer
Casualty	Integer
People#1	Information
People#2	Information
Weather	Category
Accident type	Category
Road shape	Category
Road alignment	Category

4.3 Accident and non-accident data

Accident data are first recorded by the police and the format is shown in Table 8. The data are written to the minute. The day and location of each accident are also recorded in the data. In addition, the death toll, the number of casualties, and information about people related to each accident are recorded in the data. The weather condition and shapes of lines are also given in the data in the form of ID categories. The number of accidents within the entire data is 1888. 848 accidents occurred in 2011, 741 in 2012, and 299 in 2013.

4.4 Preprocessing

The accident and traffic data are input to Hive and were processed using Hadoop. The duplicated and unclear data were removed in 1,888 pieces of the entire data, and 1421 pieces of the data remain.

4.5 Create training dataset and over-sampling

Hadoop and Hive were also used to create learning data and preprocessed data were sorted with Hive query so they could be fitted to Hadoop. Table 6 shows the model of

Table 9 Schema of training dataset

Variable	Type
Year	Numeric
Month	Numeric
Date	Numeric
Day	Choice
Hour	Numeric
Minute	Numeric
Road shape	Choice
Road alignment	Choice
Number of lane	Numeric
Weather	Choice
Accident happen	True or false
Volume	Numeric
Velocity	Numeric
Volume 5 min ago	Numeric
Velocity 5 min ago	Numeric

Table 10 Original SMOTE performance experiment

Number of examples	1000	2000	3000	4000	5000
Elapse time (s)	8	38	92	168	272
Number of examples	6000	7000	8000	9000	10,000
Elapse time (s)	403	580	844	2,202	Failed

training. The number of created accident and non-accident data is 682,884, which is shown in Table 9. In the Hadoop process, 762 mappers and 300 reducers were created. The process was finished in 627 s.

The accident data accounted for 0.14 % and the non-accident data accounted for 99.86 %, which shows the imbalance of the data. To solve this problem, we make SMOTE based on MapReduce and it is running.

SMOTE involves calculating distances among every examples to figure out each examples k nearest neighbor examples. Therefore, as the number of examples increases, the elapse time to finish the process increases rapidly. Table 10 describes how long SMOTE takes to finish creating synthetic examples. In the experiment, we adjusted the number of examples from 1000 to 10,000. Each example has 20 attributes whose values are between 0 and 100.0. To run the SMOTE program, the parameter K is set to 5, and N is set to 600.

To improve its performance, we devised its MapReduce-base algorithm and implemented it based on Hadoop framework. As shown in Fig. 4, it consists of two jobs, each of which is composed of a mapper and a reducer. The first job is to calculate distances among every example. The mappers of the job receives attributes of each example,

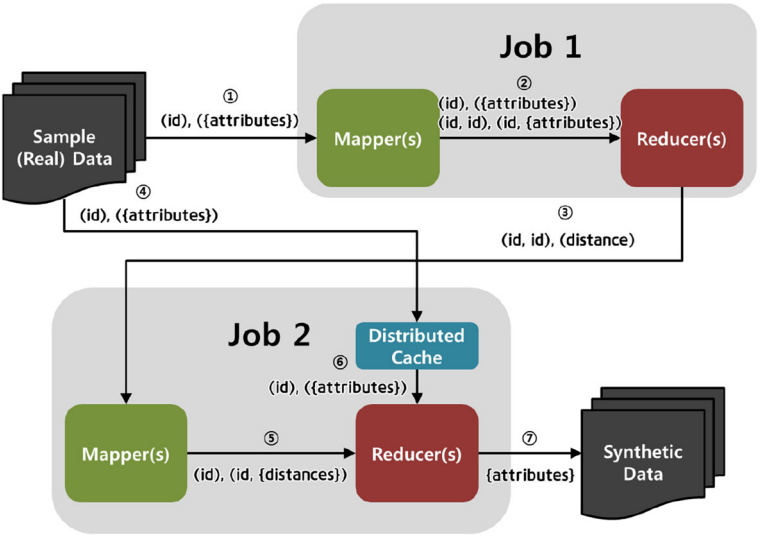

Fig. 4 MapReduce-base SMOTE overview

Table 11 MapReduce-base SMOTE performance experiment

Number of examples	1000	2000	3000	4000	5000
Elapse time (s)	37	42	49	59	73
Number of examples	6000	7000	8000	9000	10,000
Elapse time (s)	92	112	138	168	193

identified by its id, and pair all the examples with the other examples attributes so that the distances among them can be calculated. The paired results are sent to the reducer, which then figures out distances among the examples by calculating Euclidean distances using each examples attributes as a vector. The result files of the first job are sent to the second job as input files. The second job is to sort the results by the distances so that every examples k nearest neighbors are revealed, and the same calculations are conducted to create synthetic examples, which is the final goal of this method. The mappers of it simply receive the input files and set each key to each examples id and send it to the reducers. The reducers receive inter-example distances from the mappers and attributes of the real data from the distributed cache, which holds the original input data, so that the calculations SMOTE involves can be done. As the result of the reducer, synthetic attributes are created. To run the MapReduce-base algorithm, a 23-node cluster, which has virtually 69 nodes, was used. Table 11 shows the improved performance of the MapReduce-base SMOTE algorithm, compared with the original SMOTE based on single process.

While the number of the accident data increased using MapReduce-base SMOTE in whole environment system, the results of classification analysis were checked. The greatest change in the results was observed when the rate of the accident data was between 23 and 25 % of the whole dataset. Running SMOTE algoritm was stopped at 23.5 % when N of SMOTE is 11,200. The result is shown in Table 12.

Table 12 Number of training dataset before and after over-sampling

	Before over-sampling	After over-sampling
Accident	1421 (0.27 %)	160,174 (23.5 %)
Non-accident	522,710 (99.73 %)	522,710 (76.5 %)
Total	524,131 (100 %)	682,884 (100 %)

Table 13 Distribution of data in clusters

Clusters	Number of data in cluster	Number of accident in cluster
Cluster #0	75,799	18,580 (24.51 %)
Cluster #1	92,188	20,822 (22.59 %)
Cluster #2	83,996	20,665 (24.60 %)
Cluster #3	88,091	18,740 (21.27 %)
Cluster #4	94,238	20,181 (21.41 %)
Cluster #5	86,044	20,984 (24.39 %)
Cluster #6	81,945	19,060 (23.26 %)
Cluster #7	80,583	21,142 (26.24 %)
Total	682,884	160,174 (100.00 %)

4.6 *K*-means cluster analysis

When the data were divided into eight groups with a *K*-means algorithm, the result was the most efficient. The accident data were distributed to each group. Table 13 shows the distribution and the rate of the accident data. The rate ranges from 21.41 to 26.24 %, which shows the distribution was achieved properly. The result was more accurate when classification analysis was achieved with a set of groups that represent each feature.

4.7 Logistic regression analysis and prediction accuracy

We used 75 % of the whole data in the training process and predicted 25 % of the result with logistic regression analysis. Logistic regression was conducted on the eight groups, the accuracy of which is shown in Table 14. Based on this result, 76.35 % whole prediction accuracy was obtained. The true positive rate is 40.83 %. This accuracy was attempted with the over-sampling technique.

We compared other experiments. First experiment, it is classification for score of categories and products review in Arabic corpus for opinion mining (ACOM) using SVM algorithm and under-sampling [25]. And second experiment is classification for continuous insurance contract using logistic regression analysis and over-sampling.

The comparison result is shown in Table 15. Our experiment result showed high accuracy than first experiment and showed low accuracy than second experiment. But, our experiment result showed 1.4 % high true positive rate than second experiment. Our dataset is more imbalances and the dataset size is bigger than other experiments.

Table 14 Confusion matrix

	Prediction		
	Accident	Non-accident	Total
Actual			
Accident	16,351	23,692	40,403
Non-accident	16,676	114,002	130,678
Total	33,027	137,694	170,721

Table 15 Confusion matrix

	First experiment	Second experiment	Our experiment
Sampling	Under	Over	Over
Classification	SVM	Logistic regression	Logistic regression
Number of dataset	1846	40,000	682,884
Positive	145 (7.8 %)	1557 (3.9 %)	1421 (0.27 %)
Negative	1702 (92.2 %)	38,443 (96.1 %)	522,710 (99.73 %)
Accuracy	73.63 %	84.77 %	76.35 %
True positive rate	–	39.4 %	40.83 %

5 Conclusion

This article suggested a data mining process for classification of imbalance data based on MapReduce. By applying the process, a classification analysis was carried out to predict traffic accidents using highway traffic data. From the results, the performance of the data mining process was tested using total and target precision. The imbalance data problem was modified using the over-sampling method to prevent the biased result in classification analysis of the imbalance data. In addition, using Hadoop with MapReduce, the big traffic data were efficiently processed and learning data creation, cluster analysis, and classification analysis were thus able to be performed. Furthermore, we devised the MapReduce-base SMOTE method to solve the problem regarding the imbalance among dataset.

According to the experiment, the accuracy and true positive rate were 76.35 and 40.83 %, respectively, which showed a close similarity to the results presented in other research. As for the MapReduce-base SMOTE method, its performance was proved to be much better than the original one in scalability. Hadoop was efficient for processing big data and for creating the learning data and solving the imbalance problem, which could be a solution for continuously increased data in the future. The precision can be improved by adding the locations of traffic accidents, and information of the highway to the data. As for the SMOTE method, to make it perform better on bigger data, more experiments need to be done in the future as well. Additionally, the boosting technique which enhances the precision of classification analysis can also be applied to extract better results.

A few developments need to be addressed in the future works. First, an efficient over-sampling standard should be established. Second, the process time and accuracy of analyzed results should be compared when Hadoop is used to process several classification analyses. Third, other testing methods will be needed to evaluate the analyzed results. Finally, real-time data learning and prediction cannot be applied with Hadoop, since it is impossible to process real-time data. Therefore, other researches are required to find a new method to solve the real-time process problem.

References

1. Lv Y, Tang S, Zhao H (2009) Real-time highway traffic accident prediction based on the *k*-nearest neighbor method. In: International conference on measuring technology and mechatronics automation (ICMTMA), vol 3, pp 547–550
2. Yu R, Liu X (2010) Study on traffic accidents prediction model based on RBF neural network. In: 2nd international conference on information engineering and computer science (ICIECS), pp 1–4
3. Lv Y, Tang S, Zhao H (2010) Research on influence extention of two-lane highway intersections based on traffic accident database. In: International conference on optoelectronics and image processing (ICOIP), vol 2, pp 244–246
4. Kamei Y, Monden A, Matsumoto S, Kakimoto T, Matsumoto K-I (2007) The effects of over and under sampling on fault-prone module detection. In: Empirical software engineering and measurement, pp 196–204
5. Gothenberg A, Tenhunen H (1998) Performance analysis of low oversampling ratio sigma-delta noise shapers for RF applications. In: Proceedings of the 1998 IEEE international symposium on circuits and systems (ISCAS'98), vol 1, pp 401–404
6. Dean J, Ghemawat S (2008) MapReduce: simplified data processing on large clusters. Commun ACM 51:107–113
7. Lee T, Kim H, Rhee K-H, Shin S-U (2013) Implementation and performance of distributed text processing system using Hadoop for e-discovery cloud service. Innov Inf Sci Technol Res Group (ISYOU) 4:12–24
8. Zhang F, Sakr M (2013) Dataset scaling and MapReduce performance. In: 2013 IEEE 27th international on parallel and distributed processing symposium workshops and PhD forum (IPDPSW), pp 1683–1690
9. Guruzon. http://guruzon.com. Accessed 20 Nov 2013
10. Chen T-S, Hu X-Q, Li S-A, Zhou C-L (2008) Multi-class diagnosis classification on high dimension data by logistic models. In: 2008 international conference on machine learning and cybernetics, vol 6, pp 3301–3306
11. Seliya N, Xu Z, Khoshgoftaar TM (2008) Addressing class imbalance in non-binary classification problems. In: 20th IEEE international conference on tools with artificial intelligence (ICTAI'08), vol 1, pp 460–466
12. Maithani S, Tyagi R (2008) Noise characterization and classification for background estimation. In: International conference on signal processing, communications and networking (ICSCN'08), pp 208–213
13. Yan Z, Wang X, Du L (2011) Design method of highway traffic safety analysis model. In: International conference on transportation, mechanical, and electrical engineering (TMEE), pp 151–154
14. Beshah T, Ejigu D, Abraham A, Snasel V, Kromer P (2011) Pattern recognition and knowledge discovery from road traffic accident data in Ethiopia: implications for improving road safety. In: 2011 world congress on information and communication technologies (WICT), pp 1241–1246
15. Ramani RG, Shanthi S (2012) Classifier prediction evaluation in modeling road traffic accident data. In: 2012 IEEE international conference on computational intelligence and computing research (ICCIC), pp 1–4
16. Ghimire B, Bhattacharjee S, Ghosh SK (2013) Analysis of spatial autocorrelation for traffic accident data based on spatial decision tree. In: 2013 fourth international conference on computing for geospatial research and application (COM.Geo), pp 111–115
17. Apache, Apache Hadoop. https://hadoop.apache.org/. Accessed 20 Nov 2013

18. Apache, Apache Hive. https://hive.apache.org/. Accessed 20 Nov 2013
19. Apache, Apache Mahout. https://mahout.apache.org/. Accessed 13 Jan 2014
20. Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP (2002) SMOTE: synthetic minority over-sampling technique. J Artif Intell Res 16:321–357
21. Vapnik VN (1998) Statistical learning theory. Wiley, New York
22. Kukar M (2004) Transduction and typicalness for quality assessment of individual classifications in machine learning and data mining. In: Fourth IEEE international conference on data mining (ICDM'04), pp 146–153
23. Raghavendra PS, Chowdhury SR, Kameswari SV (2010) Comparative study of neural networks and *k*-means classification in web usage mining. In: International conference on internet technology and secured transactions (ICITST)
24. Rahayu SP, Purnami SW, Embong A (2008) Applying kernel logistic regression in data mining to classify credit risk. Inf Technol 2:1–6
25. Mountassir A, Benbrahim H, Berrada I (2010) An empirical study to address the problem of unbalanced data sets in sentiment classification. In: 2012 IEEE international conference on systems, man, and cybernetics (SMC), pp 3298–3303