Analysis and implementation of abnormal user data for large scale communication based on spark

Siqin Huang
School of IoT Engineering, Jiangnan University
Wuxi, China
e-mail: 1059949572@qq.com

Wei Fang
School of IoT Engineering, Jiangnan University
Wuxi, China
e-mail: fangwei@jiangnan.edu.cn

***Abstract*—Abnormal communication refers to the user in the call traffic and business management and other daily consumption of abnormal behavior in the user. Communication operators have large-scale user data sets, and using data sets reasonably to make good guidance and recommendation to businesses can bring better economic benefits. However, for large-scale user feature datasets, serial machine learning and analysis methods spend a lot of time in feature processing, and data sets training is facing a huge time cost. In order to process and train abnormal user data better and more efficiently, this paper uses Spark to implement feature engineering and analyze large-scale anomaly user datasets to highlight the efficiency of Spark in analyzing feature data and implement distributed training to accelerate the algorithm model . The training algorithm takes the SVM distributed training dataset as an example and compares it with the stand-alone serial SVM and scikit-learn SVM. The experimental results show the advantages of distributed computing as well as good training results. Finally, common logistic regression and Bayesian algorithms and other distributed computing models to compare the training effect.**

Keywords-abnormal users; SVM; distributed

I. INTRODUCTION

The phone is an indispensable communication tool in this era. People no longer have to contact friends by letters. Almost all people have mobile phones, but there are some people applying for phone numbers maliciously or never using the number without cancellation or using phone number irregularly for some reason etc.The reasons above all may lead to abnormal behavior of users. It is unrealistic to survey customers one by one, which requires a lot of manpower and resources.User feedback data can indirectly reflect user' behavior. Mining valuable information with large-scale data is promising. Machine learning technology is a good solution to this problem. Machine learning technology can learn knowledge like the user's current situation based on the existing data, and reasonable classification or regression model is used based on the learned knowledge,thus helping people make decisions.

Users' communication data has always been relatively large-scale.The processing efficiency of the user data in feature engineering or data mining is not high. Sklearn is a well-known machine learning library in the field of data mining [1]. The proceesing efficiency of stand-alone serial SVM and stand-alone sklearnSVM is not high. Therefore, distributed storage computing technology is used to improve efficiency and solve the problem of excessive cost.

Spark is an open source, highly reliable, high-performance distributed computing framework for big data [3]. In recent years, many scholars and companies at home and abroad use Spark to research or develop. Spark-based machine learning is one of the most important fields. Many scholars have made a lot of contributions in this field. There are many Spark tools including Mahout, MLlib [5,12], H2O and SAMOA .The processing engines the tools use include Hadoop MapReduce, Apache Spark, and Apache Storm [2,3].

In this paper, some data of communication users is obtained in some way.The feature engineering and training speed of the data are obviously very slow in the stand-alone mode. Much time can be reduced based on MLlib classification training and the distributed feature engineering method. The time cost of standalone SVMs, sklearnSVMs, and SparkSVMs is calculated and compared. Several common classification algorithms are implemented distributedly to compare the results.

II. THEORETICAL SUPPORT

A. Stochastic gradient descent

Gradient descent algorithm core iterative is defined as:

$$\omega_{t+1} = \omega_t - \eta_t \nabla p(\omega_t) = \omega_t - \eta_t \left[\lambda \omega_t + \frac{1}{N} \sum_{i=1}^{N} l'\left(\omega_t^T x_i, y_i\right) \right] \quad (1)$$

t is the number of iterations , $\eta_t = \frac{\eta_0}{1+\lambda \eta_0 t}$ is the step size , known as the learning rate.

In each iteration, all the gradient values $\nabla p(\omega_t)$ need to be calculated, that is to say, the gradient values of the entire training set need to be stored, which is computationally intensive for a large-scale environment [8,10,12] .

Stochastic gradient descent algorithm allows the reduction of the original problem:

$$\omega_{t+1} = \omega_t - \eta_t \nabla p(\omega_t) = \omega_t - \eta_t \left[\lambda \omega_t + l'\left(\omega_t^T x_i, y_i\right) \right] \quad (2)$$

As can be seen from the formula above, in each iteration of the stochastic gradient descent algorithm only a sample point $\left(x_i, y_i\right)$ needs to be selected from the training set.

B. Spark

Spark pays special attention to two cases: iteration jobs and interactive analyzes. Spark needs to hold a set of working data in memory in a distributed environment for both the tasks to be execuuted. RDD [4,10,12] is used to save the data set. RDD provides a distributed fault-tolerant storage way for Spark. Spark defines a series of

978-1-5386-7445-1/18/$31.00 ©2018 IEEE
DOI 10.1109/DCABES.2018.00043

RDD operations that support parallel computing in addition to these new distributed data sets [10,12].

RDD operations can be broadly divided into two categories: transformation and action. Transformation uses a user-defined function to convert one type of RDD A to another type of RDD B. Transformation includes map(), jiatMap(), and filter(). On the other hand, the actual calculations need to be carried out. The action handles a specific RDD and produces some type of result.

Action includes reduce() and collect(). Spark executes both transformation and action. Figure 1 shows a simple data flow in Spark. First, data is loaded from the file system into the RDD. After loading, a series of transformations are performed on the RDD. Finally, after the implementation of an action, the program terminates, and the result is still RDD format.

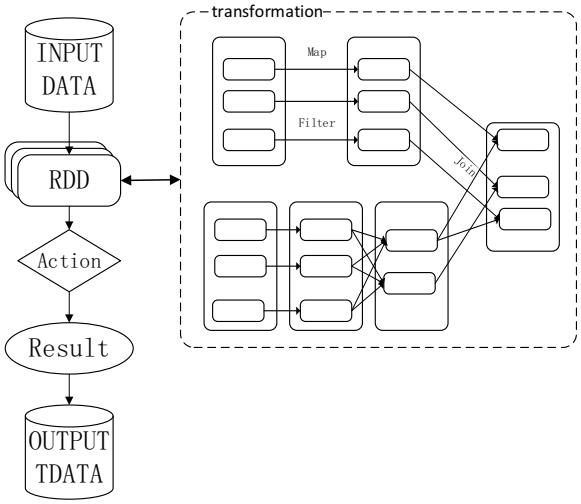


Figure 1. Data flow

III. EXPERIMENT PROCEDURE

A. Environmental configuration and modeling instructions)

Spark works on the cluster, with a master node and four computing nodes.Spark uses Hadoop2.6 and Sprak1.6.1 partially. With python as the development language, the installation of pyspark and unifying environment variables are required .

The work is to achieve a machine learning model to identify the abnormal users. Figure 2 presents the flow chart of preparing and pre-processing the data.The basic steps are as follows:

- Collect communication data.
- Preprocess data, screen subjectively, converse feature and so on.
- Data analysis, principal component analysis, correlation analysis, noise removal.
- Model adjustment, data adjustment according to the training model.
- Create a model, test the data, view the training model, and view the classification results.
- Present the results and visually analyze training efficiency and error.

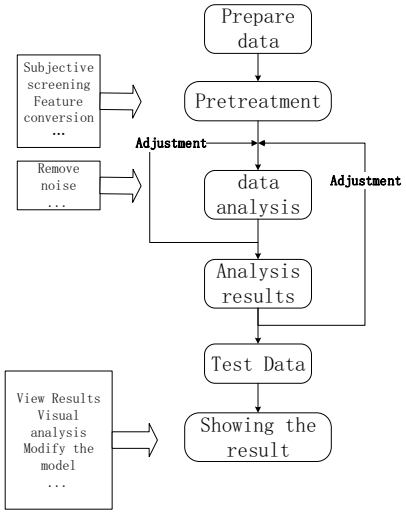


Figure 2. Experiment process

B. Data collection and visual analysis

During the entire data analysis process, the data processing stage accounts for half of the workload, and to find the right model according to data, data preprocessing is crucial. The quality of data processing directly affects the classification results.

The purpose is to find the classification model based on the user communication data, so that operators can intuitively understand the user situation. The number of the communication user is 50,000, with more than 180 features. The speed is faster intuitively than general methods when preprocessing and analysising the data. First, delete some meaningless features artificially, such as characteristics like anetwork market, network cities and counties and so on.These features has no connection to the exception.

C. Micro and macro analysis

Before classifying the data, make statistics on the data, delete some data with obvious deviations from the values, and calculate the correlation coefficients and distances.

The first step is to make statistics on data including the maximum,the minimum,the Euclidean distance, the null value, the zero value,the standard deviation and so on in the row.The abnormal attributes like the terminal usage duration attribute are deleted. The null value, the maximum and the minimum are all zero, so this feature can be judged as a noise feature.Noise can be the background recording error or the data crawling error that cannot be used.

We need to judge and handle it if there are null values or the value is too large.Methods like average fill, gradient fill, and some related features can be directly filled zero can be adopted. The Euclidean distance can be calculated without much discovery. The correlation coefficient is calculated and a few repetitions are removed.

There can be interference in the acquisition when alculating the average, standard deviation, variance, the larger the deviation of the data, so the data needs to be properly transformed. MLlib can be used to calculate the mean, variance and Euclidean distance, and the speed is

much faster than sklearn. The calculation results show that some features are more discrete. After analyzing the anomalous characteristics, some features such as monthly fee and redeemable points are removed.Some features are stored in subjectively with high density, such as average ARPU in the recent 3 months and ARPU in the current month. Finally, a better data set is achieved.

D. Spark training

In the era of big data, distributed computing that can be used to accelerate speed has always played a decisive role. Taking SVM as an example, we use SGD (Stochastic Gradient Decreasing) to randomly select datasets for optimization. We store datasets distributedly to optimize calculation to return updates.Broadcast weights for the primary node, and updates iteratively to get the optimal parameters. A better SVM classifier is achived [3,4].

When the dataset is ready, the dataset can be trained distributedly on Spark [6,7,15]. There are many applications in this area [9-12], which are applied to many fields. Distinguishing abnormal users from all others is a classification problem. The purpose is to train a good classifier to identify abnormal users.The problem lies in the time cost of large-scale data, so using spark to accelerate training is necessary. Store distributedly, train with SGD at each sub-node, and then return to the main node to update the parameters weights. After updating, broadcast a copy to each node, and iteratively update parameters until returning to the master node.

Figure 3 shows a simple diagram of iteratively updating of weights:

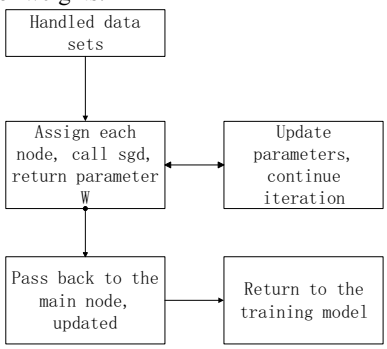


Figure 3. training

Steps to run on Spark:

- Configure spark environment on the cluster with four nodes.Build Hadoop first, and make some basic configurations.Then install jdk, scala, and finally the layout Spark.Select a machine as the master, extract the installation package and then distribute it to other nodes.
- Load and parse the data file, move the label to the last column, and convert the format before the SVM loads the data.
- Train the data to generate classification model.SVM kernel function selects linear kernel.The optimization algorithm is sgd, with iteration 100 times. Each worker performs the stochastic gradient optimization. The result is updated to the master, and then broadcast back to worker for iteration.
- Output the model parameters and save the model.
- Evaluate training sample models and calculate training errors.

IV. COMPARATIVE RESULTS

The processed data sets are input, and the training set are selected to broadcast by dividing training sets and testing sets randomly. First of all, the experiment records the time consumption of single serial SVM and distributed SVM many times. As shown in TABLE I, the stand-alone SVM uses the grid parameter optimization to find the penalty parameter c and the RBF kernel function g. The distributed SVM is optimized with sgd with 100 iteration times:

TABLE I. RUNNING TIME COMPARISON

times	Serial svm 90.796%	Spark SVM 89.899%
1	255.839934 s	real 0m23.387s user 0m0.976s sys 0m3.211s
2	253.738127 s	real 0m23.841s user 0m0.960s sys 0m3.206s
3	240.080624 s	real 0m23.385s user 0m0.950s sys 0m3.219s
4	256.416608 s	real 0m23.277s user 0m0.961s sys 0m3.193s
5	246.987717 s	real 0m23.634s user 0m0.942s sys 0m3.251s

As the table shows, the distributed operation efficiency is obviously much higher comparing the training results and time. There is not much difference in the accuracy rate, both about 90%.

The data set is randomly selected, but the step size is randomly selected, so the random number is fixed every time. Since the method iteratively updates many times, the selection of random numbers has no significant effect on the result. Figure 4 shows that the result of the data set with uncertain steps that are randomly selected.The effect is relatively stable.With the setting of different step length, the variance of the result is small and stable. The horizontal axis of the legend is the number of times, and the vertical axis is the training test result.

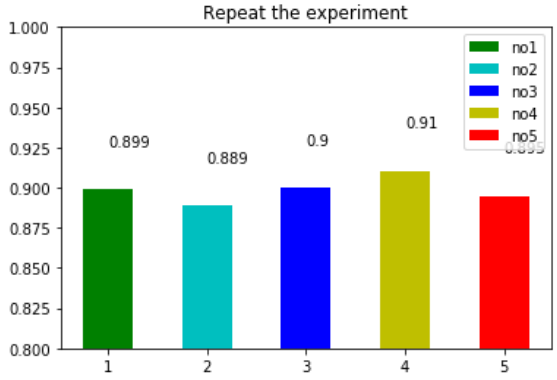
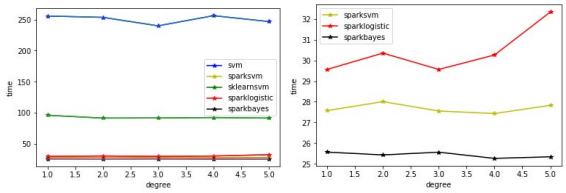


Figure 4. Five times the results of the operation contrast

Figure 5(a) shows that the distributed training has a significant acceleration effect and saves a lot of time compared with single machine when using general SVM, sklearnSVM, SparkSVM, Sparklogistic and SparkBayes. Single serial machine takes 8 times longer and the sklearn library function takes 3 times longer than training with Spark. As can be seen, distributed computing can improve the training speed well.

Figure 5(b) shows the time needed that the three training models on Spark, SVM, Logistic regression using the SGD optimization method and Bayes.



(a) (b)

Figure 5. All algorithms compare the efficiency of experiments

As shown in TABLE II, the training results of stand-alone SVM, sklearn library SVM, distributed SVM, distributed logistic regression and distributed Naïve Bayes are recorded separately. Each method has been adjusted to show the best situation. Recognizing abnormal users rate reaches about 90%.

TABLE II. EXPERIMENTAL RESULTS

SVM	sklearn SVM	Spark SVM	Spark Logistic	Spark Bayes
90.796 %	89.520%	89.899%	89.925%	83.278%

V. CONCLUSION

Combining feature engineering and experimental results map, distributed trainig achieved excellent training efficiency. The above methods have achieved good classification results, but the time cost is very different. The general optimization methods mainly using methods like gradient descent optimization. If the data set much bigger, time cost is terrible when running on single serial machines. The importance of training data distributedly is obvious. Choosing a good algorithm model will help us greatly.

Due to the fact that users'communincation data noise is harder to clean and the situation is more complicated in the real-world, compared with some experimental data sets on the network. The effect is not so good, and there are only 50,000 users in this experiment. Therefore, the recognition rate reaching 90% is not bad relatively.

It is obvious that processing large-scale data distributedly can effectively reduce the time and improve efficiency. I believe that with much more data, the results will be much better. Study will go on and more efficient methods will be found for acceleration [13-16]. Methods as follows will be taken to get more data and achieve better results. Implement data mining technology to get more valuable information. Find more excellent distributed and parallel machine learning algorithms to accelerate data mining.

REFERENCES

[1] Scikit-learn: Machine learning in Python. Pedregosa, Fabian,Varoquaux, Gael,Gramfort, Alexandre,Michel, Vincent,Thirion, Bertrand,Grisel, Olivier,Blondel, Mathieu,Prettenhofer, Peter,Weiss, Ron,Dubourg, Vincent,Vanderplas, Jake,Passos, Alexandre,Cournapeau, David,Brucher, Matthieu,P. Journal of Machine Learning Research . 2011.

[2] A. N. Richter, T. M. Khoshgoftaar, S. Landset and T. Hasanin, "A Multi-dimensional Comparison of Toolkits for Machine Learning with Big Data,"2015 IEEE International Conference on Information Reuse and Integration, San Francisco, CA, 2015, pp. 1-8.

[3] J. Fu, J. Sun and K. Wang, "SPARK – A Big Data Processing Platform for Machine Learning,"2016 International Conference on Industrial Informatics - Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII), Wuhan, 2016, pp. 48-51.

[4] D. Siegal, J. Guo and G. Agrawal, "Smart-MLlib: A High-Performance Machine-Learning Library,"2016 IEEE International Conference on Cluster Computing (CLUSTER), Taipei, 2016, pp. 336-345.

[5] Tong Xiao. Medical health data analysis based on Spark Mllib[A]. Wuhan Zhicheng Times Cultural Development Co..Proceedings of Joint 2016 International Conference on Artificial Intelligence and Engineering Applications (AIEA 2016)[C].Wuhan Zhicheng Times Cultural Development Co.:,2016:4.

[6] Wael Etaiwi, Mariam Biltawi, Ghazi Naymat, Evaluation of classification algorithms for banking customer's behavior under Apache Spark Data Processing System, In Procedia Computer Science, Volume 113, 2017, Pages 559-564, ISSN 1877-0509.

[7] L. Yang, F. Wang and T. Wang, "Analysis of dishonorable behavior on railway online ticketing system based on k-means and FP-growth,"2017 IEEE International Conference on Information and Automation (ICIA), Macau, 2017, pp. 1173-1177.

[8] D. Jayanthi and G. Sumathi, "Weather data analysis using spark − An in-memory computing framework," *2017 Innovations in Power and Advanced Computing Technologies (i-PACT)*, Vellore, India, 2017, pp. 1-5.

[9] *P. Adib, S. Alirezazadeh and A. Nezarat, "Enhancing trust accuracy among online social network users utilizing data text mining techniques in apache spark,"* 2017 7th International Conference on Computer and Knowledge Engineering (ICCKE), *Mashhad, 2017, pp. 283-288.*

[10] Spark*., 2016, [online] Available: https://spark.apache.org/.*

[11] Xiangrui Meng, Joseph Bradley, Burak Yavuz, Evan Sparks, Shivaram Venkataraman, Davies Liu, Jeremy Freeman DB Tsai, Manish Amde, Sean Owen et al.,*Mllib: Machine learning in apache spark.*, 2015.

[12] *Machine Learning Library (MLlib) Guide.*, 2016, [online] Available: https://spark.apacheorg/docs/latest/mllib-guide.html.。

[13] Zhendong Bei, Zhibin Yu, Ni Luo, Chuntao Jiang, Chengzhong Xu, Shengzhong Feng, Configuring in-memory cluster computing using random forest, In Future Generation Computer Systems, Volume 79, Part 1, 2018, Pages 1-15, ISSN 0167-739X.

[14] Álvaro Brandón Hernández, María S. Perez, Smrati Gupta, Victor Muntés-Mulero, Using machine learning to optimize parallelism in big data applications, In Future Generation Computer Systems, 2017, , ISSN 0167-739X.

[15] Lina Zhou, Shimei Pan, Jianwu Wang, Athanasios V. Vasilakos, Machine learning on big data: Opportunities and challenges, In Neurocomputing, Volume 237, 2017, Pages 350-361, ISSN 0925-2312.

[16] Zhigao Zheng, Jinming Wen, Shuai Liu, Introduction to the Special Section on Heterogeneous Computing Era, In Computers & Electrical Engineering, Volume 60, 2017, Pages 45-47, ISSN 0045-7906.