Bagging-Based Logistic Regression With Spark: A Medical Data Mining Method

Jian Pan[1,a*], Yiang Hua[2,b], Xingtian Liu[3,c], Zhiqiang Chen[3,d], Zhaofeng Yan[2,e]

[1]Zhijiang College of Zhejiang University of Technology, Shaoxing 312030, China

[2]Jianxing Honors College, Zhejiang University of Technology, Hangzhou, 310023, China

[3]College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou, 310023, China

[a]pj@zjut.edu.cn, [b]hya1994@qq.com, [c]lt1417208@qq.com, [d]frudechen@qq.com, [e]yanzf27@163.com, *corresponding author

Keywords: Medical Data Mining; Bagging; Logistic Regression; Spark

Abstract. Medical data in various organizational forms is voluminous and heterogeneous, it is significant to utilize efficient data mining techniques to explore the development rules of diverse diseases. However, many single-node data analysis tools lack enough memory and computing power, therefore, distributed and parallel computing is in great demand. In this paper, we propose a comprehensive medical data mining method consisting of data preprocessing and bagging-based logistic regression with Spark (BLR algorithm) which is improved for better compatibility with Spark, a fast parallel computing framework. Experimental results indicated that although the BLR algorithm took a little more duration than logistic regression (LR), it was 2.12% higher than LR in accuracy and outperformed LR with other common evaluation indexes.

Introduction

With the rapid development of modern medicine, the real data in diverse organizational forms collected from patients is increasing. Various structured or unstructured data is voluminous and heterogeneous [1]. It is highly meaningful and valuable for disease prognosis, diagnosis and treatment to utilize a variety of data mining techniques to explore the development rules and the correlation of diseases and discover the actual effect of treatments.

There are many data analysis tools such as WEKA and SPSS, their biggest weakness, however, is that they are only able to running at one single node. Provided that the dataset is large enough, it is a challenge for single-node tools with limited memory and computing power. Therefore, distributed and parallel computing has been frequently used. MapReduce and Spark are two of the most popular parallel computing frameworks. In comparison with MapReduce, Spark not only raises processing speed and real-time performance but also achieves high fault tolerance and high scalability based on in-memory computing [2].

In this paper, we propose a comprehensive medical data mining method comprising mainly two steps: 1) data preprocessing; 2) bagging-based logistic regression with Spark (BLR algorithm). Initially, the raw data was normalized by z-score standardization in view of the heterogeneity of medical data. The raw data in CSV format was converted into LIBSVM format subsequently for loading into RDD (Resilient Distributed Datasets), the memory-based data object in Spark. The BLR algorithm is based on bagging and logistic regression and is improved for better compatibility with Spark. To elaborate, predictions came out from each logistic regression classifier were integrated into the final prediction. Experimental results indicated that although the BLR algorithm took a little more duration than logistic regression (LR), it outperformed LR with common evaluation indexes.

Related Work

A number of medical data mining methods have been proposed in past decades, Lavindrasana et al. summarized different types of data mining algorithms including frequent itemset mining, classification, clustering, etc. The data analysis methodology must be suitable for the corresponding medical data [3]. Harper compared four classification algorithms comprising discriminant analysis, regression models (multiple and logistic), decision tree and artificial neural networks and concluded that there was not necessarily a single best algorithm but the best performing algorithm depending on the features of the dataset [4]. Taking the dataset we used into consideration, all values of the features are real and continuous, so logistic regression is more suitable than the other methods. All the above works were completed in single-node environment, however, as one of the most fast parallel computing frameworks, Spark implements common machine learning algorithms in its MLlib [5]. Qiu et al. proposed a parallel frequent itemset mining algorithm with Spark (YAFIM), their experimental results indicated that YAFIM outperformed MapReduce around 25 times in a real-world medical application [6].

Compared to the above methods, the BLR algorithm we propose has two distinctions: 1) it achieves parallel computing over RDD and take advantage of memory for iterative computations. 2) it is based on bagging and logistic regression and is improved for better compatibility with Spark. Experimental results proved that the BLR algorithm outperformed LR with common evaluation indexes.

Data Preprocessing

We used Wisconsin Diagnostic Breast Cancer (WDBC) dataset which is available from the UCI machine learning repository [7]. The dataset has 32 attributes (1 class label and 30 features, the ID number is excluded) and 569 instances. It has two class labels (Malignant, Benign) for binary classification. All values of the features are real and continuous, which indicates that logistic regression is more appropriate to be used instead of decision tree or Apriori algorithm which prefer discrete values. Fallahi et al. excluded the instances containing missing data and used SMOTE to correct data unbalancing [8]. Different dataset should be preprocessed in different method on the basis of its data features. On the one hand, the WDBC dataset has no missing value, so there is no need to exclude any instance. On the other hand, the positive class (357 Malignant) and the negative class (212 Benign) have reached a relatively balanced state.

Bagging-Based Logistic Regression With Spark

We propose the bagging-based logistic regression with Spark (BLR algorithm) on the basis of bagging and logistic regression. Spark MLlib has implemented logistic regression, their experimental results proved that logistic regression in Spark MLlib was 100× faster than MapReduce [5].

Experiment shows that real-time prediction model is lower by 70% compared with the average error of the basic model. The APE value under average real-time prediction model reaches 8.98. This means if the bus starts from the Huanglong stop, the error time is between 2.4 to 3.6 minutes with an average error of 3 minutes, since the total travel time of route K193 is about 30~45min.

Logistic Regression in Spark MLlib. Logistic regression model can be expressed as

$$p_i = P(y_i = 1 \mid x_{1i}, x_{2i}, \cdots, x_{mi}) = \frac{\exp(\alpha + \sum_{m=1}^{M} \beta_m x_{mi})}{1 + \exp(\alpha + \sum_{m=1}^{M} \beta_m x_{mi})} \quad (1)$$

where p_i is the probability of the ith event when given the values of the independent variables x_{1i}, x_{2i}, $\cdots$, x_{mi}. If p_i is larger than the threshold we set, the ith event will be classified into positive class label, and vice versa.

For computing the classification prediction, coefficients β_m must be estimated through optimization. Spark MLlib provides two optimizers: Stochastic Gradient Descent (SGD) and Limited memory BFGS (L-BFGS). Quoc Le et al. held the point that SGD has two weaknesses: 1) it needs a lot of manual tuning of optimization parameters, e.g., convergence criteria and learning rates. 2) it is hard to achieve parallel computing on clusters. L-BFGS is able to simplify and speed up the procedure of training deep learning algorithms significantly [9].

The main step of L-BFGS method can be expressed as

$$\beta_{k+1} = \beta_k - \alpha_k H_k g_k \quad (2)$$

where β_k, β_{k+1} is the estimated coefficients, H_k is Hesse matrix. L-BFGS only stores the pairs $\{y_j, s_j\}$ in memory to update Hesse matrix [10]. It is precisely because L-BFGS is memory-intensive and performs better in limited memory that we choose it to become our optimization. It is also should be noted that we used L2 regularization to prevent overfitting.

The BLR Algorithm. Bagging-based logistic regression with Spark is on the basis of bagging and logistic regression. Bagging is an effective model to improve classification accuracy. By making bootstrap replicates of the training set with replacement, it can generate multiple versions of a classifier with equal weight and aggregate them into one predictor through plurality vote [11].

In the process of implementation with Spark, the training set was randomly sampled for five times with replacement, thus five versions of logistic regression classifiers were generated. The aggregated predictor need to compute the final prediction of one instance by integrating all predictions from the five versions of logistic regression classifiers. For one test tuple, we note that p_i is the prediction of the ith classifier, n_1 is the amount of the positive class label from the five classifiers, n_0 is the amount of the negative class label, thus

$$0 \le p_i \le 1.0 \quad (3)$$

Assume that the threshold is 0.5, if $p_i \ge 0.5$, n_1 plus one, else, n_0 plus one. The plurality vote is accomplished when either n_1 or n_0 reaches 3. Consider that

$$0 \le \frac{p_i}{2} \le 0.5 \quad (4)$$

The final prediction p can be expressed as

$$p = \begin{cases} 0.5 + \dfrac{\sum_{i=1}^{5} \frac{p_i}{2}}{5}, n_1 > n_0 \\ \dfrac{\sum_{i=1}^{5} \frac{p_i}{2}}{5}, n_1 < n_0 \end{cases} \quad (5)$$

A test tuple is classified as positive class if and only if $p \ge 0.5$, and vice versa. Spark improves its fault tolerance by Lineage, which records the coarse-grained conversions to RDD. Once some partitions of a RDD are lost, Spark can get enough information through Lineage to redo operations and recover the partitions which have been lost. The Lineage graph for the RDDs in the BLR algorithm is shown in Figure 1.

Training set was stored as RDD and was converted into bootstrap sample through the operation randomSplit(weights). Further five logistic regression models were generated on the basis of bootstrap sample. Finally, the models output the predictions and the labels for every test tuple. All data stored as RDD can be recovered by Lineage.

Performance Evaluation

Experimental Environment. The BLR algorithm and LR were implemented in Spark 1.2.0 with Scala language 2.10.4. The Hadoop version was 2.2. We made our experiments on a cluster

consisting of 5 nodes shown in Table 1.The computing nodes were all running at Ubuntu 14.04 and JDK 1.8.

Experiments and Performance Evaluation Results. In order to obtain more reliable results, for the training set and the test set, we used bootstrap to sample dataset randomly with replacement.

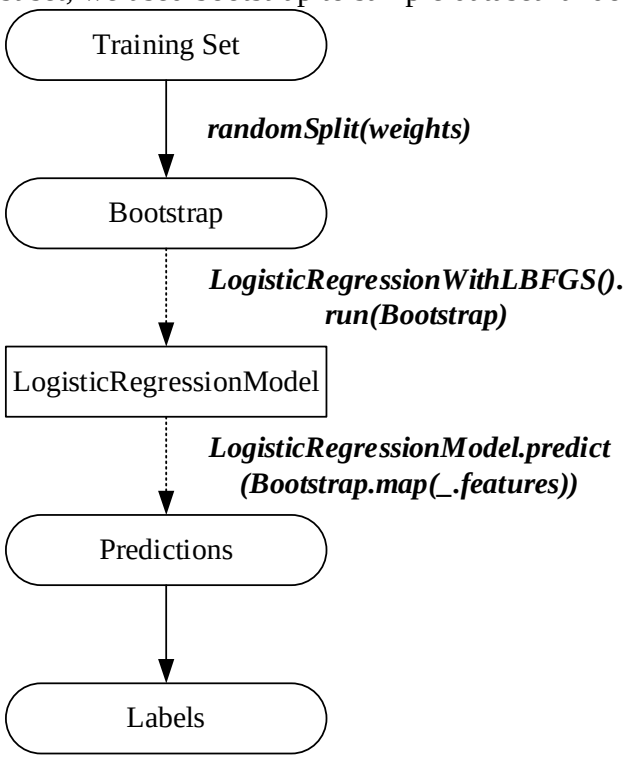

Figure 1 Lineage Graph for the RDDs in BLR

Table 1 The Status of the Cluster

Node	Memory	Cores
Master	2G	1
Worker 1	1G	1
Worker 2	1G	1
Worker 3	1G	1
Worker 4	1G	1

It should be noted that the number of the instances in different test sets may not equal but to keep a relatively constant proportion. Whenever a tuple is chosen, it is also likely to be selected again, which guarantees the hybridity of the data and the reliability of the results. Using aforementioned two algorithms, for each training set, two classification models can be learned automatically in Spark. Once classification model was produced, each test tuple will be classified based on its feature values. The whole procedure was repeated 5 times for two algorithms with the 100 iterations in L-BFGS. Table 2 shows the classification results, i.e., the confusion matrixes.

Table 2 Confusion Matrixes of Two Algorithms by Bootstrap

Bootstrap	Confusion Matrix of LR		Confusion Matrix of BLR	
1	112	5	111	1
	3	165	1	172
2	105	5	104	0
	5	165	1	175
3	111	4	103	1
	1	167	0	179
4	109	3	103	1
	4	171	0	183
5	106	6	112	2

1	172	0	171

As shown in the confusion matrixes, for either LR or BLR, the sum of TP and TN is always far greater than the sum of FP and FN, which qualitatively indicates that both algorithms have a high classification accuracy. More to the point, the quantity of FP and FN of BLR is much less than that of LR. For quantitatively comparing the two algorithms, we computed the average of 5 common evaluation indexes including accuracy, sensitivity, specificity, precision and recall based on the confusion matrixes and presented them in Figure 2.

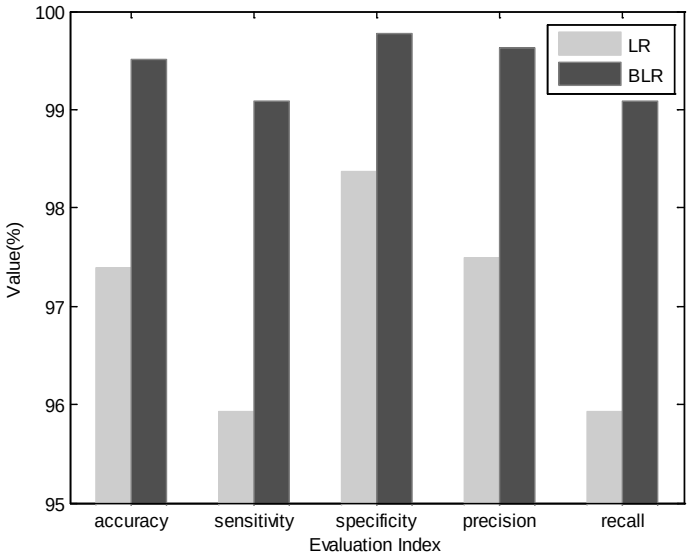

Figure 2 Common Evaluation Indexes of LR and BLR

As illustrated in Figure 2, although the evaluation indexes of both algorithm are over 95%, BLR has higher values than LR in all five indexes. To be more specific, BLR is 2.12%, 3.15%, 1.41% and 2.14% higher than LR in accuracy, sensitivity (recall), specificity and precision respectively.

Another indicator for evaluating the classification performance is Receiver Operating Characteristic (ROC) curve. The accuracy of a classification algorithm is higher if the area under curve (AUC) is close to 1.0. We used Spark MLlib to obtain the discrete points (FPR, TPR) and fit the ROC curve. Given the AUC is so close to 1.0 that we focus on the key part of the ROC curve shown in Figure 3. The AUC of BLR is approximately 0.9981 and that of LR is about 0.9732. It is apparent that the ROC curve of BLR is above LR. The results show better performance for BLR algorithm than that for LR.

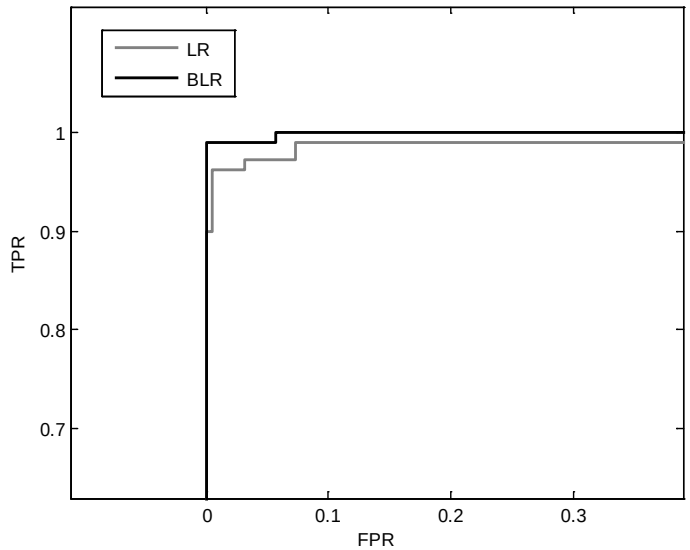

Figure 3 The Key Part of the ROC Curve of LR and BLR

We also recorded the average duration of the two algorithms. The average duration of LR and BLR is increasing simultaneously. From the perspective of comparison, for each number of iteration, the average duration of BLR is slightly higher than LR.

Conclusion

Medical data in various organizational forms is voluminous and heterogeneous, it is meaningful and significant to utilize efficient data mining techniques to explore the development rules and the correlation of diverse diseases and discover the actual effect of treatments. However, it is a challenge for single-node data analysis tools with limited memory and computing power, therefore, distributed and parallel computing is in great demand.

In this paper, we propose a comprehensive medical data mining method consisting of mainly two steps: 1) data preprocessing; 2) bagging-based logistic regression with Spark (BLR algorithm). Initially, the raw data was normalized by z-score standardization in view of the heterogeneity of medical data. The raw data in CSV format was converted into LIBSVM format subsequently for loading into RDD. The BLR algorithm is based on bagging and logistic regression and is improved for better compatibility with Spark. To elaborate, by making bootstrap replicates of the training set with replacement, five versions of logistic regression classifiers were generated. The aggregated predictor computes the final prediction of one instance by integrating all predictions from the five classifiers. Experimental results indicated that although the BLR algorithm took a little more duration than logistic regression (LR), it was 2.12%, 3.15%, 1.41% and 2.14% higher than LR in accuracy, sensitivity (recall), specificity and precision respectively.

Acknowledgements

This work is supported by the Department of Science and Technology, Zhejiang Provincial People's Government (No. 2016C33073) and the Zhejiang Xinmiao Talent Grants(No. 2015R403040).

References

[1] Cios K J, Moore G W. Uniqueness of medical data mining[J]. Artificial intelligence in medicine, 2002, 26(1): 1-24.

[2] Spark[OL]. http://spark.apache.org./

[3] Iavindrasana J, Cohen G, Depeursinge A, et al. Clinical data mining: a review[J]. Yearb Med Inform, 2009, 2009: 121-133.

[4] Harper P R. A review and comparison of classification algorithms for medical decision making[J]. Health Policy, 2005, 71(3): 315-331.

[5] Spark MLlib[OL]. http://spark.apache.org/mllib/.

[6] Qiu H, Gu R, Yuan C, et al. YAFIM: A Parallel Frequent Itemset Mining Algorithm with Spark[C]. Parallel & Distributed Processing Symposium Workshops (IPDPSW), 2014 IEEE International. IEEE, 2014: 1664-1671.

[7] Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

[8] Fallahi A, Jafari S. An expert system for detection of breast cancer using data preprocessing and Bayesian network[J]. Int J Adv Sci Technol, 2011, 34: 65-70.

[9] Ngiam J, Coates A, Lahiri A, et al. On optimization methods for deep learning[C]. Proceedings of the 28th International Conference on Machine Learning (ICML-11). 2011: 265-272.

[10] Liu D C, Nocedal J. On the limited memory BFGS method for large scale optimization[J]. Mathematical programming, 1989, 45(1-3): 503-528.

[11] Breiman L. Bagging predictors[J]. Machine learning, 1996, 24(2): 123-140.