Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 113 (2017) 559–564

Procedia Computer Science

www.elsevier.com/locate/procedia

The 4th International Symposium on Emerging Information, Communication and Networks (EICN 2017)

Evaluation of classification algorithms for banking customer's behavior under Apache Spark Data Processing System

Wael Etaiwi*, Mariam Biltawi and Ghazi Naymat

Princess Sumaya University for Technology, Amman. Jordan

Abstract

Many different classification algorithms could be used in order to analyze, classify or predict data. These algorithms differ in their performance and results. Therefore, in order to select the best approach, a comparison studies required to present the most appropriate approach to be used in a certain domain. This paper presents a comparative study between two classification techniques namely, Naïve Bayes (NB) and the Support Vector Machine (SVM), of the Machine Learning Library (MLlib) under the Apache Spark Data processing System. The comparison is conducted after applying the two classifiers on a dataset consisting of customer's personal and behavioral information in Santander Bank in Spain. The dataset contains: a training set of more than 13 million records and a testing set of about 1 million records. To properly apply these two classifiers on the dataset, a preprocessing step was performed to clean and prepare data to be used. Experimental results show that Naïve Bayes overcomes Support Vector Machine in term of precision, recall and F-measure.

Keywords: Naïve Bayes, Spark, Machine Learning, Support Vector Machine, Big Data.

1. Introduction

The active generation and analysis of large volumes of structured and unstructured data caused the big data problem, this problem is due to three main characteristics: volume, velocity and variety of the data, which are referred to as the 3Vs, in turn these characteristics lead to system challenges in implementing machine learning framework[1]. Thus a

* Corresponding author. Tel.: +962795744288.
E-mail address: w.etaiwi@psut.edu.jo

1877-0509 © 2017 The Authors. Published by Elsevier B.V.
Peer-review under responsibility of the Conference Program Chairs.
10.1016/j.procs.2017.08.280

powerful machine learning tools, strategies and environment are needed to properly analyze the large volumes of data. The term volume of data refers to the large amount of data collected from many different sources, such as: sensors, databases, multimedia or websites. The speed at which data is collected and analyzed is the key factor when dealing with real time systems such as: sensors and Radio-Frequency Identification (RFID) systems, and this is what the term Velocity means. While Variety of data concerns about different formats of data such as video, audio, email messages, text files, etc. Because big data problems concern about collecting data from its different sources, processing, analyzing and extracting knowledge from it; many frameworks have been proposed in order to deal with such problems, these frameworks are available, but they required to be tested and evaluated in order to select the most suitable framework that can solve a specific big data problem quickly and precisely, knowing that the traditional machine learning algorithms are not applicable on such kind of data[2]. Apache Spark is an open source programming frameworks for data processing originated in the University of California, Berkeley[3]. It has the capability to analyze, manage, process and solve big data problems using an expressive development APIs to allow data workers to develop and execute their works. The Apache Spark operates data processing tasks on many distributed data processing machines, which requires a file management system to collaborate data on those machines such as Hadoop Distributed File System (HDFS), and distribute storage system such as Spark standalone and Hadoop YARN. Because Apache Spark completes data analysis in-memory, it is fast and near real-time framework, in comparison to other big data processing modules, such as MapReduce. Apache Spark can be as 10 times faster for batch processing[4]. Apache Spark architecture consists of three main components: Driver Program, which has the main function to be distributed and executed on other machines. Cluster Manager, which manages cluster resources, and the Worker Node, which is a machine that executes application code. Machine Learning Library (MLlib) is one of the Apache Spark components that consists of common machine learning algorithms and utilities. This paper focuses on two MLlib classification algorithms used for prediction; Naïve Bayes (NB) and support vector machine (SVM). NB is a linear classifier based on the Bayes theorem, it creates simple and well performed models, and it assumes that the features in the dataset are mutually independent, thus the term naïve came along[5]. While, SVM is a learning algorithm that performs classification by finding the hyper plain that maximizes margin between two classes, and the nearest points to the hyper plain are the support vectors that determine the maximum margin[6]. Knowing that Spark MLlib is a new library established in 2014, with little number of published research papers providing evaluation and comparison studies, the goal of this paper is to evaluate and compare two main machine learning algorithms of the MLlib under Apache Spark through predicting bank customer's behaviours. A preprocessing step required to prepare dataset to be analysed. Experimental results were conducted by applying two prediction algorithms on a dataset consisting of customer’s personal information and their behaviour in Santander Bank[†]. The remaining of this paper is structured as follows; section 2 presents the related work. Methodology is presented in section 3, experimental results and evaluation are discussed in section 4, and finally the conclusion is presented in section 5.

2. Related Work

In general two main machine learning tools were considered the best as noted by Richter et al.[7], they presented a multidimensional comparison of four main open source machine learning tools that are used in big data; Mahout, MLlib, H2O, and SAMOA in terms of algorithm availability, scalability, and speed. Although the choice of using one of the tools depends on the goal of the application, the authors conducted that MLlib and H2O are the best tools in terms of algorithm availability, task pipelining and data manipulation. Another research by Landset et al.[8] also claimed that MLlib and H2O are the best machine learning tools in terms of speed, usability, algorithms covered, and scalability to different sizes of datasets. Several research papers were published in the domain of big data, these papers showed that the Spark MLlib is either compared with other machine learning tools or was treated as a part of an architecture/software. A research paper that is an example of comparing the MLlib with other tools is the one presented by Kholod et al.[9]. They proposed a Cloud for Distributed Data Analysis (CDDA) based on the actor model. CDDA is compared with Spark MLlib and Azure ML in terms of performance. Both CDDA and Spark MLlib were tested on a high performance hardware and systems. Experiments were conducted on datasets from Azure ML and results showed

[†] Santander Banks: Retail banking company, https://www.santanderbank.com/us/personal

that Spark MLlib and Azure ML are little bit slower than the proposed approach in terms of execution time. Results also showed that CDDA outperformed both Azure ML and Spark MLlib in terms of the efficiency and acceleration of parallel execution. The research presented by Wang et al.[10] is an example of using Spark MLlib as a part of a proposed architecture. The authors focused on the big data provenance, started by presenting the challenges and gaps faced in it, and ended by proposing a reference architecture for a Big Data provenance platform. In their architecture, the decision tree model which is on the top of the Kepler provenance was implemented using the spark MLlib. Another example presented by Xu et al.[11], the authors proposed an architecture for real-time data-analytics-as-service that aimed to provide real-time data analytics service through integrating the backend training system with data services, prediction services, and the real-time prediction products. The backend training system was wrapped as a service by the dynamic model training services. In the proposed architecture, Spark was chosen as a framework and the MLlib as a machine learning library to train the dataset. Another example was presented by Sewak and Singh[12] that proposed referential architectures for small and medium (SME), and large E-commerce enterprises. The authors started with a comprehensive discussion about the role of Apache Spark in the modern E-commerce platforms and then proposed two referential architectures, one for small and medium E-commerce enterprises and the other for large E-commerce enterprises. They stated that either Spark MLlib or H2O can be utilized in the architectures of the large E-commerce enterprises. Many researches used MLlib library to evaluate the proposed methods, as an example; Peralta et al.[15] presented a MapReduce for Evolutionary Feature Selection (MR-EFS) algorithm, the goal of this paper was to enable EFS models to be applicable on big data by developing the MapReduce algorithm, and to analyze the scalability of the proposed algorithm. The experimental results were conducted over two classification datasets with up to 67 million instances and up to 2000 features. The reduced datasets resulted from executing the proposed algorithm were tested using three different classifiers available in MLlib under the apache Spark and over a cluster of 20 computers.

3. Methodology

The main phases for the proposed methodology are shown in figure 1. Starting from preprocessing the raw dataset, then applying both NB and SVM separately on the processed dataset under the Spark environment. After obtaining the predicted results, a comprehensive evaluation is conducted and presented in section 4, while the subsection 3.1 discusses the dataset used, subsection 3.2 presents the preprocessing steps required to prepare the dataset, and the subsection 3.3 illustrates the applied machine learning classifiers, noting that NB and SVM are the machine learning classifiers used in this paper for the prediction purposes.

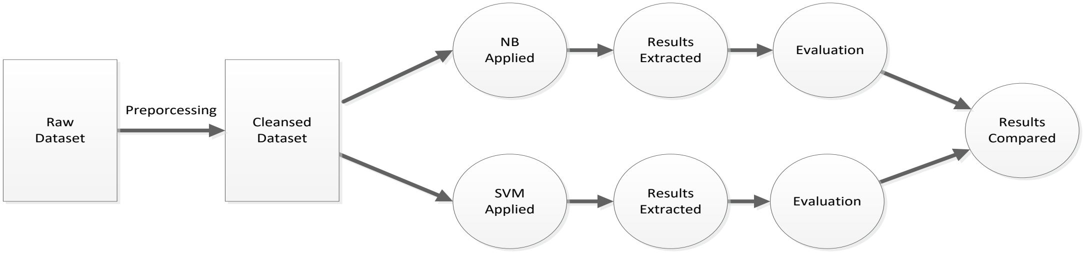

Figure 1: Methodology Steps

3.1. The Dataset

The dataset used for evaluation is freely available online in Kaggle[‡] website, it consists of customer's personal information and behaviour from the Santander Bank, which is an international financial institute that offers financial products for their customers, and through this dataset, the main task is to predict which products the customers will use next month based on their past behaviour and their personal information. The data set contains more than 13 million records for training the model and about 1 million records to test the produced model. The main types of data in dataset are: customer's personal information, and customer's behaviour according to a product they use.

[‡] Kaggle: platform for predictive modelling and analytics competitions, www.kaggle.com.

3.2. The Preprocessing Step:

The preprocessing step is an important step that aims to cleanse the dataset and prepare it to be further used in the prediction algorithm. Four sub-steps applied in the preprocessing step as illustrated in figure 2: the first step is to code all string variables and converting them into numerical variable, a lookup table with ID were built for each string field in the training set, and then we replace all string values from the training set with its corresponding ID from its lookup table. For example: the customer's country residence field (pais_residencia) contains string values, such as: ES, US, FR, etc. In this step, we put all of these values into a new lookup table, as illustrated in Table 1. Then we replace the string values (ex. ES) with its corresponding ID from the table (ex. ES replaced by 1, US replaced by 2), and so on for all other string fields.

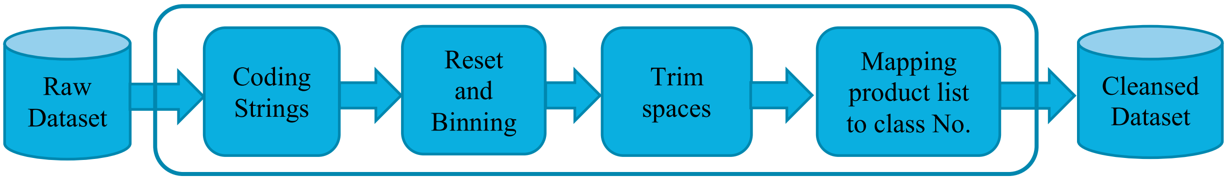

Figure 2: Preprocessing Steps

Table 1: Customer's country residence lookup table

ID	Value
1	ES
2	US
…	…

In Second step, all null values replaced by a default value (for example: zero), and the numeric fields will be binning by replacing all values fall within a specific interval into a bin. For example: "renta" field contains the customer's income value which reside between 1000 to hundreds of thousands, in this case, we bin this field into 30 bins values, for example: incomes between 1000 to 5000 will be binned to bin 1, incomes between 5000 to 10000 will be binned to bin 2, and so on. Then, the values were trimmed in the third step by removing the surrounding spaces (because dataset contains many additional spaces surrounding many features). And finally, the customer's products were merged and mapped into one class number, this is done by merging all products values into one binary number (since each product has a binary value: 1 0r 0) and converting it to decimal class number, for example: if customerX uses the last three product and leaves the others, all product's numbers will be combined together in one binary number (000000000000000000000111) to represent (7) in decimal, so customerX class number is 7.

3.3. Prediction Techniques

3.3.1. Naïve Bayes prediction approach:

Naïve Bayes is a probabilistic multiclass classification algorithm based on the Bayes theorem[21], it aims to compute the conditional probability distribution of each feature. In this paper, features are represented by each single column in customer's personal information of the used dataset, these features should have a numerical value, and therefore, every string value was coded to a numerical format in the preprocessing step. NB assumes that the features are independent of each other, represented by a vector v= (x_1, x_2, x_3, … x_n), where x_1, x_2, x_3, …, x_n are n features. The conditional probability of a vector to be classified into class C equals to the multiplication of probabilities of each vector's features to be in the class C. For example, in our dataset, in order to find the probability of a customer X to get the last three products (class 7 illustrated in Table 2), we have to find the probabilities for each customer's X feature, (for example: Age, employee index, etc.) to be in class 7, and then multiply all features probabilities together to find the customer X probability of getting the class 7 products.

3.3.2. Support vector machine (SVM) prediction approach:

SVM is a binary machine learning classification algorithm that classifies all items to only two classes (each item could be classified into one class). The main task of SVM classification model is to find the maximum margin hyperplane that classify the group of feature vectors among two classes (0 or 1), which is the maximum distance between the hyperplane and the nearest x from either classes.

4. Experiments and Evaluation

4.1. Evaluation metrics

Many measures could be used to evaluate the performance of prediction algorithms such as entropy, purity, true positive rate, true negative rate, accuracy, precision, F-measure, and computation time. According to[22], precision and recall are very informative evaluation metrics for binary classifiers. So, the performance metrics used for evaluation in this paper are precision, recall, and F-measure. Precision, also known as positive predictive value, is the number of the correctly predicted items over the number of all predicted items. While recall, also referred to as the true positive rate or sensitivity, is the number of the correctly predicted items over all related items. However, the F-measure can be calculated using the values of both precision and recall.

4.2. Results and Discussion:

Two prediction algorithms from the SPARK MLlib were used, namely NB and SVM. The implementation was built based on converting customer's personal information (features) into local vector, labelled with customer ID. Experiments were carried out on one standalone Spark equipped environment, with 24 GB RAM and Core i7-6700 CPU. The implementation source code were written in Python2.7 using PyCharm IDE. In this evaluation, time and performance do not matter even though there exists a large performance and speed disparity between the two prediction approaches running time. This disparity comes from the fact that SVM is a binary classifier approach that requires to predict each product individually regardless of the other products, it also needs to train all customer's data for each product, which is a time consuming process. The evaluation metrics extracted and computed after implementing and running NB and SVM algorithms. An observation from the experiment, as illustrated in Table 2, is that NB overcomes SVM results in terms of precision, recall and f-measure. Another observation is that multiclass classification approaches (such as NB) is better than binary classification approaches (such as SVM) in term of precision, recall, f-measure in addition to computation time.

Table 2: evaluation results

	Precision	Recall	F-Measure
NB	4%	49%	7.3%
SVM	0.18%	10%	0.3%

5. Conclusion

This paper analyzed and compared two supervised machine learning classifiers from the MLlib after applying them on a dataset containing more 14 million records, divided into 13 million records as a training set and 1 million records as a testing set, these data represented customer's personal information and behavior of the Santander Bank. The implementation conducted on Apache Spark data processing environment started by a preprocessing step in order to clean and represent the data with a proper numerical format. Then Naïve Bayes and Support Vector Machine classifiers of the Apache Spark MLlib were applied in order to predict the next customer's product to be selected or request. A comprehensive comparison is conducted after obtaining the results. The results showed that NB prediction approach is more efficient than SVM in term of precision, recall and f-measure when applying them on the same dataset with the same preprocessing approach. On the other hand, we conclude that multi-class classifiers are more efficient than binary classifiers for prediction problems.

References

1. Douglas, Laney. "3d data management: Controlling data volume, velocity and variety." Gartner. Retrieved 6 (2001): 2001.

2. Bello-Orgaz, Gema, Jason J. Jung, and David Camacho. "Social big data: Recent achievements and new challenges." Information Fusion 28 (2016): 45-59.
3. M.Zaharia, M.Chowdhury, M.J.Franklin, S.Shenker, I.Stoica, Spark: Cluster computing with working sets, in: Proceedings of the 2Nd USENIX Conference on Hot Topics in Cloud Computing, HotCloud'10, USENIX Association, Berkeley, CA, USA, 2010, p.10. http://dl.acm.org/citation.cfm?id=1863103.1863113.
4. Katherine Noyes, Five things you need to know about Hadoop vs. Apache Spark, InfoWorld 2015.
5. Raschka, Sebastian. "Naive bayes and text classification i-introduction and theory." arXiv preprint arXiv:1410.5329 (2014).
6. Hearst, Marti A., Susan T. Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. "Support vector machines." IEEE Intelligent Systems and their Applications 13, no. 4 (1998): 18-28.
7. Richter, Aaron N., Taghi M. Khoshgoftaar, Sara Landset, and Tawfiq Hasanin. "A Multi-Dimensional Comparison of Toolkits for Machine Learning with Big Data." In Information Reuse and Integration (IRI), 2015 IEEE International Conference on, pp. 1-8. IEEE, 2015.
8. Landset, Sara, Taghi M. Khoshgoftaar, Aaron N. Richter, and Tawfiq Hasanin. "A survey of open source tools for machine learning with big data in the Hadoop ecosystem." Journal of Big Data 2, no. 1 (2015): 1.
9. Kholod, Ivan, Ilya Petukhov, and Andrey Shorov. "Cloud for Distributed Data Analysis Based on the Actor Model." Scientific Programming 2016 (2016).
10. Wang, Jianwu, Daniel Crawl, Shweta Purawat, Mai Nguyen, and Ilkay Altintas. "Big data provenance: Challenges, state of the art and opportunities." In Big Data (Big Data), 2015 IEEE International Conference on, pp. 2509-2516. IEEE, 2015.
11. Xu, Donna, Dongyao Wu, Xiwei Xu, Liming Zhu, and Len Bass. "Making real time data analytics available as a service." In 2015 11th International ACM SIGSOFT Conference on Quality of Software Architectures (QoSA), pp. 73-82. IEEE, 2015.
12. Sewak, Mohit, and Sachchidanand Singh. "A Reference Architecture and Road map for Enabling E-commerce on Apache Spark."
13. Marquardt, Ames, Stacey Newman, Deepa Hattarki, Rajagopalan Srinivasan, Shanu Sushmita, Prabhu Ram, Viren Prasad et al. "Healthscope: An interactive distributed data mining framework for scalable prediction of healthcare costs." In 2014 IEEE International Conference on Data Mining Workshop, pp. 1227-1230. IEEE, 2014.
14. Luo, Gang. "MLBCD: a machine learning tool for big clinical data." Health information science and systems 3, no. 1 (2015): 3.
15. Peralta, Daniel, Sara del Río, Sergio Ramírez-Gallego, Isaac Triguero, Jose M. Benitez, and Francisco Herrera. "Evolutionary feature selection for big data classification: A mapreduce approach." Mathematical Problems in Engineering 501 (2015): 246139.
16. Liu, Bingwei, Erik Blasch, Yu Chen, Dan Shen, and Genshe Chen. "Scalable sentiment classification for big data analysis using naïve bayes classifier." In Big Data, 2013 IEEE International Conference on, pp. 99-104. IEEE, 2013.
17. Seminario, Carlos E., and David C. Wilson. "Case Study Evaluation of Mahout as a Recommender Platform." In RUE@ RecSys, pp. 45-50. 2012.
18. Albadarneh, Jafar, Bashar Talafha, Mahmoud Al-Ayyoub, Belal Zaqaibeh, Mohammad Al-Smadi, Yaser Jararweh, and Elhadj Benkhelifa. "Using big data analytics for authorship authentication of arabic tweets." Utility and Cloud Computing (UCC), 2015 IEEE/ACM 8th International Conference on, pp. 448-452. IEEE, 2015.
19. Liang, Fan, Chen Feng, Xiaoyi Lu, and Zhiwei Xu. "Performance benefits of DataMPI: a case study with BigDataBench." In Workshop on Big Data Benchmarks, Performance Optimization, and Emerging Hardware, pp. 111-123. Springer International Publishing, 2014.
20. Ann Paul. Support Vector Machines in Apache Spark. International Journal of Advanced Research (2013), Volume 4, Issue 8, 76-80
21. Friedman, Nir, Dan Geiger, and Moises Goldszmidt. "Bayesian network classifiers." Machine learning 29.2-3 (1997): 131-163.
22. Saito, Takaya, and Marc Rehmsmeier. "The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets." PloS one 10.3 (2015): e0118432.