Mesopotamian journal of Big Data
Vol. **(2022)**, 2022, **pp**. 1–11
DOI: https://doi.org/10.58496/MJBD/2022/001 ISSN: 2958-6453
https://mesopotamian.press/journals/index.php/BigData

Research Article

Big Data Machine Learning Using Apache Spark Mllib

Ziaul Hasan[1,*], Hong-Jie Xing[2],M. Idrees Magray[3]

[1]*Department of Biosciences, Jamia Millia Islamia, New Delhi-110025, India*

[2]*College of Mathematics and Information Science. China*

3National Chiao Tung University, Hsinchu, Taiwan

ARTICLE INFO

Article History

Received 17 Nov 2021
Accepted 11 Jan 2022
Published 15 Jan 2022

Keywords

Apache

Big Data

Spark

Mllib

Machine Learning

ABSTRACT

The examination local area has utilized man-made brainpower, and specifically machine learning, in various ways to change various unique and, surprisingly, heterogeneous data sources into excellent realities and information, offering driving capacities to exact example finding. In any case, utilizing machine learning strategies on enormous and convoluted datasets is computationally costly and utilizes a great deal of coherent and actual assets, including central processor, memory, and data record space.In the current study collected the review of different researchers from 2010 to 2022. As how much data produced consistently arrives at quintillions of bytes, it is turning out to be more pivotal than any other time in recent memory to have a vigorous stage for powerful big data examination. Quite possibly of the most notable big datum investigation stages is Apache Spark MLlib, which gives various extraordinary capabilities for machine learning applications like relapse, grouping, aspect decrease, bunching, and rule extraction. This study's hidden reason is that Spark ML's big data execution and precision are fundamentally better than Spark Mllib's. The dataset for bank client exchanges is utilized in the correlation. We are probably not going to have the option to handle the sums and sorts of data we are managing with conventional programming arrangements. Thus, present day big data handling innovations that can disperse and deal with data in a versatile way are either coordinated into or taken over by conventional business knowledge (BI) frameworks. Big data innovation can likewise assist us with learning more about security, which can be found from colossal databases. The big data examination motor Apache Spark is utilized in the review to introduce a security-related data investigation.

1. INTRODUCTION

Big data advancements have as of late acquired fame and are utilized in various enterprises. For most of entrepreneurs, tracking down the most ideal ways to robotize their undertakings and dissect their tremendous measures of data is a pivotal issue. There is a genuine competition to deal with the steady deluge of data from a few sources effectively and at an elevated degree of productivity. Foreseeing client wearing down in view of their data and conduct is quite possibly of the most ordinary need. [1] This request is more prominent for ventures like banking and telecoms that arrangement with huge quantities of clients. In this work, an examination investigation was done utilizing conditional data from bank clients. Stir expectation is the method involved with predicting a client's craving to leave. Perhaps of the most argumentative concentrate as of late is this one. To gauge bank clients' probability of leaving, this study utilizes a dataset of their exchanges.

Apache Spark is progressively widely used in light of its superb exhibition and productivity in handling tremendous measures of data, which is 100 x faster than Apache Hadoop. It has additionally included answers for MapReduce's inadequacies.[2] The Versatile Circulated Datasets (RDD) in the Mllib bundle filled in as the establishment for Spark's machine learning APIs, yet presently the primary Programming interface is the ML bundle, a more significant level Programming interface based on Data Approaches that makes it simpler to carry out certifiable ML pipelines, especially highlight changes. The Apache Spark ML documentation makes reference to that Data Edges offer an additional congenial Programming interface than RDDs. Spark Data sources, SQL/Data Casing inquiries, Tungsten and Impetus

Corresponding author. Email: zhasan.biochem@gmail.com

improvements, and predictable APIs across dialects are only a couple of the many benefits of Data Edges. With Spark 2.0, the Mllib bundle is in upkeep mode. In this review, a correlation of the two projects is made concerning model preparation, model assessment, and precision.

One of the most famous open-source, stage autonomous big data machine learning libraries, Apache Spark Mllib, benefits from conveyed engineering and programmed data parallelization. Relapse, aspect decrease, grouping, bunching, and rule extraction are only a couple of the machine learning exercises that Apache Spark Mllib accommodates an assortment of machine learning errands. [3] While established researchers has long investigated machine learning and its helpful applications, little has been found out about big data machine learning systems like Apache Spark Mllib. This commitment is perhaps quick to utilize the Apache Spark Mllib 2.0, a big data machine learning bundle, to resolve the issue of huge data investigation. Getting state of the art computational foundations is one objective of big data investigation so huge measures of data might be mined and handled rapidly and really. This fills in as the essential main impetus behind the ongoing work. Different equipment as well as programming designs would influence execution and client experience on the grounds that big data investigation requires a ton of handling. In this review, we utilize different big data scientific errands to survey the impacts of different equipment and programming designs. We give ideas for future big data machine learning-centered equipment, programming, and model plan in light of the review's discoveries.

1.1 Apache Spark Mllib 2.0

In 2010, Spark was created in the UC Berkeley AMPLab and made accessible for use. Spark was at first furnished with test machine learning calculations and is intended for powerful iterative processing. Preceding the improvement of Mllib, it missing the mark on assortment of dependable and adaptable learning calculations. As a part of the Mlbase project, Mllib's improvement began in 2012 (Kraska et al., 2013). In September 2013, Mllib was made openly accessible. Starting from the start, Mllib has been packaged with Spark; the main form of Mllib was essential for the Spark 0.8 delivery. The Apache 2.0 permit oversees Spark, an Apache project, and Mllib subsequently. Besides, beginning with Spark variant 1.0, Mllib and Spark both follow a three-month discharge cycle. A consolidated choice of normal machine learning strategies was presented by the main cycle of Mllib, which was made at UC Berkeley by 11 colleagues. Since this underlying delivery, the quantity of supporters of Mllib has expanded emphatically. As of the Spark 1.4 delivery, under two years after the fact, Mllib has in excess of 140 benefactors from in excess of 50 associations.

1.2 History and Growth

In 2010, Spark was created in the UC Berkeley AMPLab and made accessible for use. Spark was at first furnished with test machine learning calculations and is intended for compelling iterative processing. Before the improvement of Mllib, it missing the mark on assortment of dependable and versatile learning calculations. As a part of the Mlbase project, Mllib's improvement began in 2012 [4]. In September 2013, Mllib was made freely accessible. Starting from the start, Mllib has been packaged with Spark; the principal variant of Mllib was essential for the Spark 0.8 delivery. The Apache 2.0 permit oversees Spark, an Apache project, and Mllib thus. Besides, beginning with Spark form 1.0, Mllib and Spark both follow a three-month discharge cycle. A dense choice of normal machine learning procedures was presented by the principal emphasis of Mllib, which was made at UC Berkeley by 11 teammates. Since this underlying delivery, the quantity of supporters of Mllib has expanded decisively. As of the Spark 1.4 delivery, under two years after the fact, Mllib has in excess of 140 supporters from in excess of 50 associations. Figure 1 shows what the delivery adaptation means for the extension of the Mllib open source local area. The heartiness of this open-source local area has provoked the making of an immense range of new capacities.

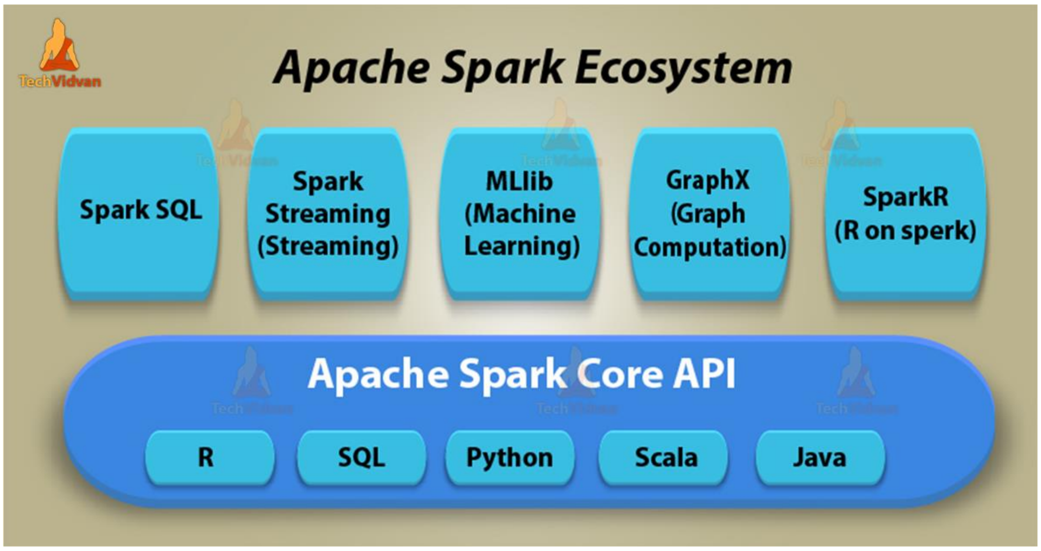

Fig. 1. Apache Spark ecosystem

2. LITERATURE REVIEW

Work in the disciplines of big data, Spark, machine learning, profound learning, and fountain learning is remembered for the writing in this field. To plan a framework that can deal with these limitations, we dissect this writing fully intent on examining earlier examination there and grasping their impediments. The four segments of the writing audit for this paper are as per the following: I Work utilizing Apache Spark; (ii) Work including highlight set and data handling; (iii) Work includes multi-facet perceptions for profound learning; and (iv) Work including overflow learning.

2.1 Work Related to Apache Spark

The creators of M. Zaharia, 2010[4] were quick to acquaint the local area with the design and usefulness of Apache Spark. The programming model, which consolidates RDDs, equal handling, and so forth, is momentarily depicted. A couple of new executions are likewise brought into the environment. X. Meng, 2016 [5] presents Mllib, the machine learning library for Apache Spark. This includes the fundamental qualities of the Mllib library along with the different parts that make it up. By running a test ML example on Spark's fundamental structure, the exploration in J. Fu 2016 [6] assesses it. In light of their examinations, the creators give exhaustive discoveries that show Sparks' advantages. The creators of L. R. Nair 2015[7] show a connected utilization of Spark by utilizing Spark's structure to investigate twitter data. The significance of utilizing a device like Spark is featured overwhelmingly of the pre-owned twitter dataset. Comparative Twitter feeling examination was completed by the creators of utilizing Apache Spark.

2.2 Work Related to Feature Set and Data Manipulation

We have handled the class lopsidedness issue in this review since it is a subject of extraordinary interest to the machine learning local area. The creators of give an outline of various techniques and advances that can be applied to address the lopsidedness issue.
Sonak, 2016[8] Gives a broad outline of the various methodologies planned to resolve the issue. The control and extraction of relevant highlights from both datasets structures a critical part of the examination depicted in this paper. I. Guyon 2003 shows this type of component designing. The creators of M. C. Popescu 2014[9] explore include extraction and element determination for picture arrangement utilizing machine learning. Utilizing express list of capabilities creation, K. Dey 2016[10] offers a state of the art answer for the test of reword distinguishing proof in normal language handling. Furthermore, creators of N. Lavrac 2010 [11] give an intensive representation of express element age and control for tending to govern learning frameworks.

2.3 Work Related to Deep Learning using MLPs

In the field of man-made reasoning, profound learning utilizing various layer perceptrons has been quickly acquiring significance, especially while taking care of gigantic measures of data. Multi-facet perceptrons are utilized to present data grouping by the creators of L. M. Silva2008.
An enormous scope investigation of tremendous data utilizing MLPs is shown in C. Sharma, " 2014 [14] This shows the advantages that MLPs give while working gigantic datasets instead of little or medium-sized ones. Y.- C. 2010[15] is one more illustration of the utilization of MLPs to arrangement issues.
The actuation capability utilized in this strategy depended on fluffy integrals. The principal learning system utilized in multi-facet perceptrons is backpropagation. A fluffy brain network model in light of MLP is introduced by the creators in S. K. Buddy 1992 [16] using the backpropagation calculation to show how learning functions in such a multi-facet brain organization.

2.4 Work Related to Cascade Learning

Flowing associates the two stages of the framework talked about in this paper — multi-facet perceptron examination and Spark examination. At the point when a sensible derivation can't be drawn from the data because of the classes' outrageous unevenness, flowing or outpouring learning is frequently utilized. This system has proactively been utilized effectively in different areas, including PC vision and regular language handling. S. M. Sarwar 2015[17] set forth an energy compelling, two-stage flowed classifier for anticipating purchase meetings and the products that would be bought during such meetings. M. Simonovsky 2016 [18] presented a component sharing profound classifier overflow known as OnionNet, in which later stages added new layers and element channels to the previous ones. Cutting edge results on the issue were as of late accomplished by P. F. Christ 2018 [19] by flowing two completely convolutional brain networks J. Long, 2015for a consolidated picture division of the liver and its sore.
In the wake of exploring the writing, we need to utilize the strategy introduced in this review to address the previously mentioned challenges. We give a careful examination of the proposed structure and the strategies utilized in it in the segment that follows.

2.5 Research Objective

- To utilize Apache Spark MLlib 2.0 and to initiate a study of big data machine learning on massive datasets including tens of millions of data records.
- To bridge the gap between the two areas, opening the doors for different interesting directions from the big data community to the fast-growing machine learning application area.
- To perform several large-scale real world experiments to examine a set of qualitative and quantitative attributes of Apache Spark MLlib 2.0.

3. MATERIALS AND METHODS

The materials and systems utilized in the ebb and flow research study are additionally made sense of in this segment. We'll begin with the parts of Apache Spark MLlib prior to presenting the datasets.

TABLE I. THE DEVELOPMENT PATHWAY OF APACHE SPARK MLLIB 2.0 [12]

Number of Column	Apache Spark MLIB 2.0 Pathway
16	2.3
21	2.9
23	3.6
36	3.9

42	4.2
49	4.6
59	5.3

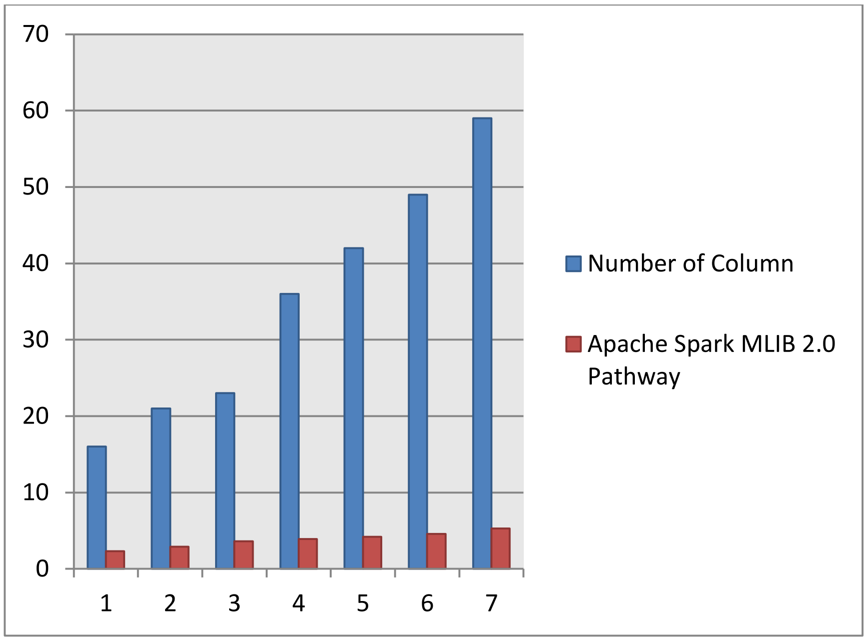

Figure 2 The Development Pathway Of Apache Spark Mllib 2.0

3.1 Machine Learning Components

We focused on a bunch of directed (order) strategies, like SVM (Backing Vector Machine), Choice Tree, Na ve Bayes, and Irregular Woods, alongside a famous unaided (grouping) machine learning calculation, specifically KMeans, to survey the presentation of the Apache Spark MLlib 2.0 library in examining big data sets. The machine learning calculations in the Apache Spark MLlib 2.0 library and the Weka library (Rendition 3.7.12) working on Hadoop2.7 were analyzed to address a relative report. We utilized Java2SE 8.1 for the entirety of the code and executions to work with Apache Spark MLlib 2.0 and Weka parts. We utilized the indistinguishable arrangements, (for example, regularization, cost, misfortune, bit type, seed, and so on) with equivalent upsides of boundaries given by both Apache Spark MLlib 2.0 and Weka libraries for each arrangement of ML calculations.

3.2 Datasets

The presentation of Apache Spark MLlib 2.0 was investigated and differentiated utilizing six unique enormous datasets. A dataset from the US government's Department of Transportation Exploration and Imaginative Innovation Organization (RITA) Site, as well as five datasets from the UCI Machine Learning Vault.
The first dataset, named "HEPMASS" is connected to a parallel characterization work and contains high-energy material science examinations to search for the hints of fascinating particles. The subsequent one, called "SUSY," is connected to a paired grouping issue to isolate a foundation action from a sign cycle that produces very symmetric particles. The third dataset, named "HIGGS," is associated with a paired grouping issue to isolate a foundation action that doesn't create Higgs bosons from a sign interaction that does. The fourth dataset, "FLIGHT", contains flight data between October 1987 and April 2008. There are factors connecting with the length of the flight, its starting point and objective, and its passed time.

TABLE II. DATASET'S ATTRIBUTES

Data Set	Characteristics	Attributes	Data Records	Size
HEPMASS	Multivariate	Real	8.1253	1.30GB
SUSY	Multivariate	Real	9.5893	2.10GB
HIGGS	Multivariate	Real	8.6932	3.16GB
FLIGHT	Multivariate	Integer Nominal	9.3289	3.50GB
HETROACT I	Multivariate	Real	8.4778	4.63GB
HETROACT II	Multivariate	Real	9.8863	5.36GB

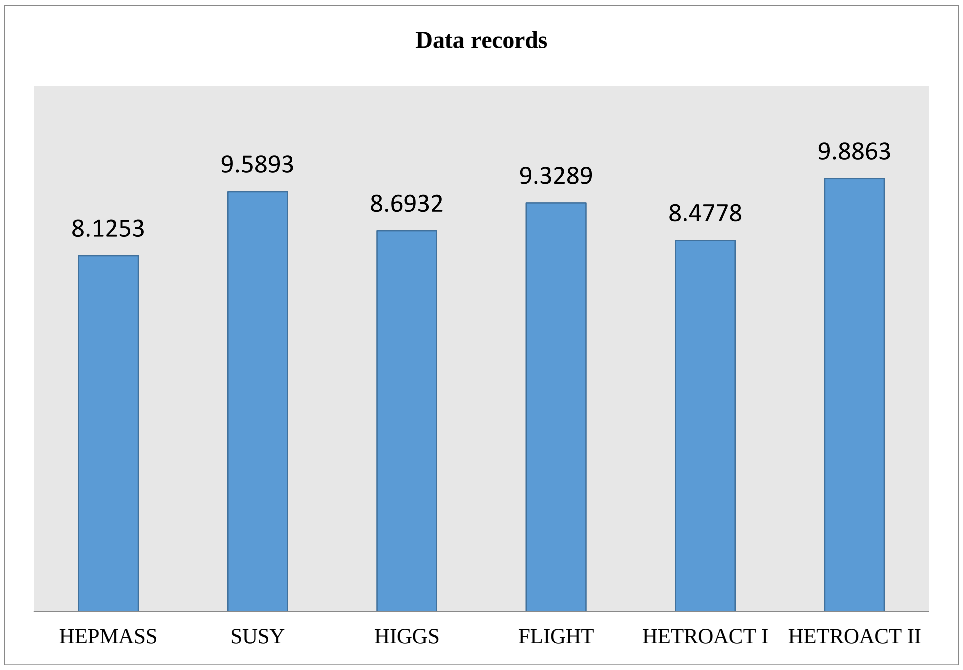

Fig. 3. Data records attribute

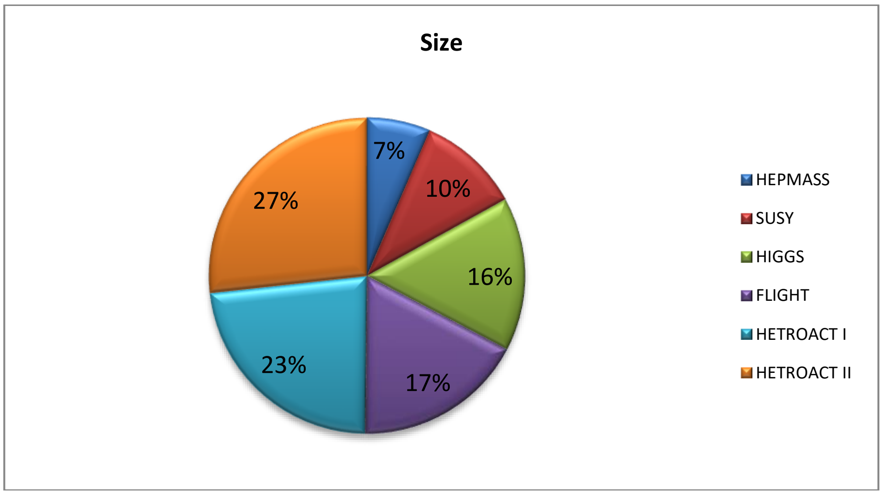

Fig. 4. Size Attributes

TABLE III. THE EMPLOYED CONFIGURATIONS OF TWO VMS ON A VMWARE CLUSTER ENVIRONMENT

Environments	Number Of Nodes	HDD	RAM	CPU
ENV 1	5	1 TB (In Total)	6 GB (Each)	5 Vcpu (Each)
ENV 2	6	1 TB (In Total)	20 GB (Each)	9 V CPU (Each)

The dataset's time, appearance air terminal, postpone time, etc. This dataset was utilized by us for classification purposes. "HETROACT I" and "HETROACT II" are the fifth and 6th datasets, separately. These heterogeneity datasets for human movement identification from Cell phone or potentially Savvy sensors are planned to investigate what sensor heterogeneities mean for calculations for perceiving human action , and we involved them for grouping. To get the Region under the ROC (auROC) for grouping, we utilized 75% of each dataset for preparing and 25% for testing, all with 4-crease cross approval. The particular attributes of each dataset are displayed in Table III

3.3 Test Environments

To get trial discoveries, two VMs in a VMWARE Group arrangement were utilized. On both VMs, we utilized two unique arrangements, "ENV1" and "ENV2," as additional displayed in Table 2. Both virtual machines were running the 64-bit CentOS 6.8 working framework with a Xeon E5-2690V3 2.6 GHz computer chip.

4. RESULTS AND DISCUSSION

We focused on the SVM (Backing Vector Machine), Choice Tree, Gullible Bayes, and Irregular Backwoods regulated (grouping) procedures. To lead exploratory examination on both managed and solo machine learning procedures, we likewise involved K-Means as an unaided (grouping) approach. Here is a summation of the six datasets utilized in our exploration, as displayed in Table 1 Outcomes show what amount of time Mllib and Weka required to run on a similar equipment setup.
Table 3 shows the regions under the ROC(s) acquired by Weka and Apache Spark MLlib utilizing SVM, Choice Tree, Nave Bayes, and Irregular Backwoods calculations on four different datasets. The trial discoveries from the K-Mean strategy across the datasets are talked about in Table IV.
The running seasons of four unique characterization calculations on the proposed datasets are displayed in Figure4 and Figure 5, give a rundown of the running times for K-implies on the HETROACT I and HETROACT II datasets. Here is a rundown of the experimental outcomes:
• Both Weka and Apache Spark MLlib's outcomes for region under the ROC are very practically identical to each other, and the thing that matters isn't genuinely critical. There might be a slight connection between's the Region under the ROC accomplished by Weka and Apache Spark MLlib.

TABLE IV. RUNNING TIME OF APACHE SPARK MLLIB COMPARED TO WEKA UNDER DIFFERENT CLASSIFICATION EXPERIMENTS. NB STANDS FOR NAIVE BAYES, DT STANDS FOR DECISION TREE, AND RF REFERS TO RANDOM FOREST.

Running Time	
Mlib	**Weka**
2.3	3.2
3.2	3.6
3.6	3.9
4.2	4..5
4.6	5.3
5.2	6.3

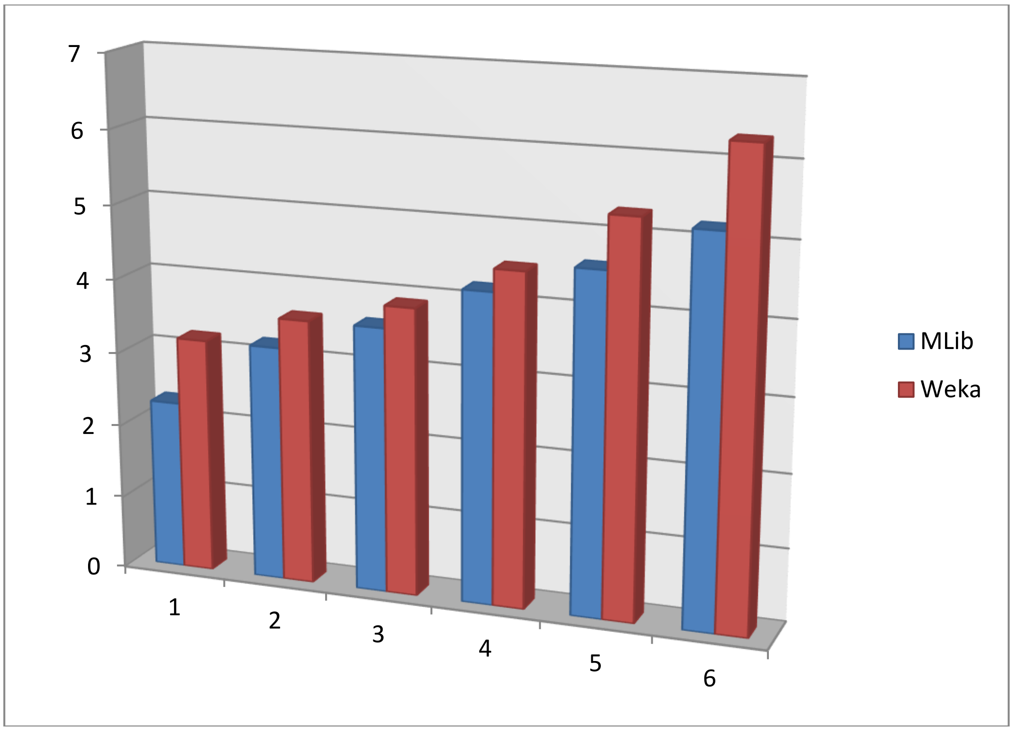

Fig. 5. Running time of Apache Spark MLlib compared to Weka under different classification experiments. NB stands for Na¨ıve Bayes, DT stands for Decision Tree, and RF refers to Random Forest.

TABLE V. THE RUNNING TIME OF APACHE SPARK MLLIB K-MEANS COMPARED TO WEKA K-MEANS CLUSTERING COMPONENT

Hetroact I	
Mlib	**Weka**
2.6	3.2
3.6	3.9
3.9	4.2
4.8	4.9
5.3	6.2
5.9	6.3

.

TABLE VI. THE GENERAL PERFORMANCE OF WEKA AND APACHE SPARK MLLIB K-MEANS ALGORITHM

Environment	Dataset	Algorithm	Mlib	Weka
ENV1	**SUSY**	**SVM Random Forest Decision Tree Naves Bayes**	**0.2563**	**1.236**
	HIGGS	**SVM Random Forest Decision Tree Naves Bayes**	**0.3623**	**1.352**
	FLIGHT	**SVM Random Forest Decision Tree Naves Bayes**	**0.4856**	**1.425**
	HEPMASS	**SVM Random Forest Decision Tree Naves Bayes**	**0.5393**	**1.536**
ENV2	**SUSY**	**SVM Random Forest Decision Tree Naves Bayes**	**0.2365**	**1.562**
	HIGGS	**SVM Random Forest Decision Tree Naves Bayes**	**0.3256**	**1.666**
	FLIGHT	**SVM Random Forest Decision Tree Naves Bayes**	**0.4395**	**1.756**
	HEPMASS	**SVM Random Forest Decision Tree Naves Bayes**	**0.5369**	**1.893**

While we are performing 4-crease cross approval, the particular boundaries of every calculation in arbitrarily choosing train and test models, or they could emerge out of the very
A t-test on the running time matched by either grouping calculations or the bunching strategy uncovers genuinely massive contrasts (at p 0.01) between Apache Spark MLlib and Weka, exhibiting that Apache Spark MLlib, true to form, can be quicker in contrast with the Weka parts we have utilized.

The study of data investigation and handling ought to be all the more innovatively progressed on the grounds that how much advanced data being gathered is expanding every day. This will permit data researchers to change the huge measure of organized and, surprisingly, unstructured data into top notch data and realities. To all the more really address the issue of example disclosure from enormous data sources, PC researchers and architects has previously started to foster big data machine learning parts. Apache Spark MLlib is quite possibly of the most famous big datum machine learning libraries at present accessible to the local area. An incredible asset for big data examination, Apache Spark MLlib has excellent execution concerning running time, as seen by the discoveries of our review. Oppositely, Weka performs less rapidly than Apache Spark MLlib while managing a lot of data tests. This correlation probably won't be viewed as fair, however, considering that Apache Spark MLlib and Weka utilize unmistakable record frameworks and setups. We involved Hadoop's disseminated record framework for Weka while running MLlib on Spark's.
Weka is used in this setting as a reliable gauge that is generally acknowledged by the examination local area in light of the fact that our point is to exhibit the way in which Spark performs with tremendous assortments. Weka has various highlights that Spark can't coordinate, including: Clients approach countless records and assets, it is extremely straightforward and easy to understand for non-master clients, and it has an ideal graphical UI. An extensive variety of machine learning techniques are upheld by Weka.

Head part investigation (PCA), outfit learning, bunching examination, and enhancement are only a couple of the machine learning parts that are rapidly, effectively, and versatile carried out by Apache Spark MLlib. Also, Apache Spark MLlib upholds huge data devices that utilization disseminated models and offers decisions for circulated handling utilizing equal handling. These prerequisites will bring about less handling time being required while likewise considering additional opportunity to look at investigation results. [20]At the point when there are various expectations to work out for the machine learning task, this turns out to be very critical. To streamline complete running time, the dispersed engineering can likewise profit from a portion of the big data device sets now accessible to help dismantle the machine learning part. One more advantage of Apache Spark MLlib is joining, which permits it to profit from different Spark environment programming parts like Spark GraphX, Spark SQL, and Spark Streaming. Moreover, the machine learning local area approaches a large number of efficient documentations, including code tests.

5. CONCLUSION

Support vector machines, a very effective classification model in machine learning, have several benefits including good generalisation, minimal parameters, and the capacity to produce globally optimal answers. For employees to process, analyse, and predict data, it is a very wise decision. Due to the excellence of their architecture and methods, support vector machines—a conventional classification method—remain useful in the context of today's huge data. However, to eliminate the need for processing huge sample data sets, people will need to improve their algorithm. Due to its high space-time complexity and lengthy training period, this method has a low efficiency issue. According to the findings of the trial, the issues we mentioned have been successfully fixed.

Problems were found during the research for this article, which examined how support vector machine techniques could be improved in the context of huge data. This study aimed to address the issues of big sample feature scale, heterogeneous information, and feature space, which standard SVM was unable to address due to its sensitivity to noise points and outliers. The issue of unequal distribution follows. The time complexity of the original problem is dictated by the dimension, and the time complexity of the dual problem is determined by the amount of data, it is found during the study process of introducing fuzzy membership into multicore learning. The scale of the data is determined by its dimension and quantity, therefore it can depend on the features we choose for the various solution spaces. By converting the dual problem solution into the original space's classification surface, the algorithm's speed can be increased. In conclusion, it is shown that by increasing the calculation rate of traditional machine learning algorithms, the accuracy of the fitting prediction between the predicted data and the actual value is as high as 98%, which can enable traditional machine learning algorithms to meet the needs of the big data era in the future. Big data applications can make extensive use of it.

Conflicts of Interest

The authors declare that there is no conflict of interests regarding the publication of this paper.

Funding

The authors receive no funding for this work.

Acknowledgment

The author appreciates the academic community at the institution for their valuable feedback and discussions that contributed to the refinement of this study.

References

[1] F. Pedregosa, G. Varoquaux, A. Gramfort, et al., "Scikit-learn: Machine learning in Python," Journal of Machine Learning Research, 2011.

[2] E. R. Sparks, A. Talwalkar, V. Smith, J. Kottalam, X. Pan, J. E. Gonzalez, M. J. Franklin, M. I. Jordan, and T. Kraska, "MLI: An API for Distributed Machine Learning," in International Conference on Data Mining, 2013.

[3] A. Talwalkar, D. Haas, M. J. Franklin, M. I. Jordan, and T. Kraska, "Automating model search for large scale machine learning," Symposium on Cloud Computing, 2015.

[4] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: Cluster computing with working sets," HotCloud, vol. 10, no. 10-10, p. 95, 2010.

[5] X. Meng, J. Bradley, B. Yavuz, E. Sparks, S. Venkataraman, D. Liu, J. Freeman, D. Tsai, M. Amde, S. Owen, et al., "Mllib: Machine learning in apache spark," Journal of Machine Learning Research, vol. 17, no. 34, pp. 1–7, 2016

[6] J. Fu, J. Sun, and K. Wang, "Spark–a big data processing platform for machine learning," in Industrial Informatics-Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII), 2016 International Conference on, pp. 48–51, IEEE, 2016

[7] L. R. Nair and S. D. Shetty, "Streaming twitter data analysis using spark for effective job search," Journal of Theoretical and Applied Information Technology, vol. 80, no. 2, p. 349, 2015.

[8] A. Sonak, R. Patankar, and N. Pise, "A new approach for handling imbalanced dataset using ann and genetic algorithm," in Communication and Signal Processing (ICCSP), 2016 International Conference on, pp. 1987–1990, IEEE, 2016.

[9] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," Journal of machine learning research, vol. 3, no. Mar, pp. 1157–1182, 2003.

[10] M. C. Popescu and L. M. Sasu, "Feature extraction, feature selection and machine learning for image classification: A case study," in Optimization of Electrical and Electronic Equipment (OPTIM), 2014 International Conference on, pp. 968–973, IEEE, 2014.

[11] K. Dey, R. Shrivastava, and S. Kaushik, "A paraphrase and semantic similarity detection system for user generated short-text content on microblogs.," in COLING, pp. 2880–2890, 2016.

[12] N. Lavrac, J. F ˇ urnkranz, and D. Gamberger, "Explicit feature construction and manipulation for covering rule learning algorithms," in Advances in Machine Learning I, pp. 121–146, Springer, 2010.

[13] L. M. Silva, J. M. de Sa, and L. A. Alexandre, "Data classification ´ with multilayer perceptrons using a generalized error function," Neural Networks, vol. 21, no. 9, pp. 1302–1310, 2008.

[14] C. Sharma, "Big data analytics using neural networks," 2014.

[15] Y.-C. Hu, "Pattern classification by multi-layer perceptron using fuzzy integral-based activation function," Applied Soft Computing, vol. 10, no. 3, pp. 813–819, 2010.

[16] S. K. Pal and S. Mitra, "Multilayer perceptron, fuzzy sets, and classification," IEEE Transactions on neural networks, vol. 3, no. 5, pp. 683–697, 1992.

[17] S. M. Sarwar, M. Hasan, and D. I. Ignatov, "Two-stage cascaded classifier for purchase prediction," arXiv preprint arXiv:1508.03856, 2015.

[18] M. Simonovsky and N. Komodakis, "Onionnet: Sharing features in cascaded deep classifiers," arXiv preprint arXiv:1608.02728, 2016.

[19] P. F. Christ, M. E. A. Elshaer, F. Ettlinger, S. Tatavarty, M. Bickel, P. Bilic, M. Rempfler, M. Armbruster, F. Hofmann, . DAnastasi, et al., "Automatic liver and lesion segmentation in ct using cascaded fully convolutional

neural networks and 3d conditional random fields," in International Conference on Medical Image Computing and ComputerAssisted Intervention, pp. 415–423, Springer, 2016.

[20] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3431–3440, 2015.