Performance of Machine Learning Algorithms in Distributed Environment: A Study

[1]M. Bhargavi Krishna , [2] Prof.S.Jyothi

[1]Research Scholar, Department of CSE, SPMVV, Tirupati, Email ID: bhargavimandara@gmail.com

[2]Professor, department of Computer Science, SPMVV, Tirupati, Email ID: jyothi.spmvv@gmail.com

Article Info
Page Number: 2057-2076
Publication Issue:
Vol. 72 No. 1 (2023)

Article History
Article Received: 15 April 2023
Revised: 24 May 2023
Accepted: 18 June 2023
Publication: 12 July 2023

Abstract:
In present scenario, to extract and study meaningful data from vast volumes of data using modern tools and techniques are necessary for making decisions. For analysing large volumes of data using a Distributed Environment with big data is efficient because it gives a solution to manage contents in the distributed system. To implement and analyse the data, three different sizes of datasets from various fields are considered. Then data is analysed in both Distributed Environment and Standalone Environment because it will provide which Environment is scalable and also to know the performance and improvement of data analysis. In this process for predicting and analysing the different sizes of data various algorithms in machine learning are applied to datasets with extensive data framework. Therefore, all applied algorithms in both environments are compared with their accuracies, precision, and re-call with time to know the best algorithm and environment.

Keywords: Machine Learning Algorithms, Big Data, Distributed Environment, Standalone Environment.

1. INTRODUCTION

In present era, a huge volume of data is generated easily at very high speed using the cloud, social media, sensor networks, and other emerging content delivery network technologies. The problems in datasets need to be classified with step-by-step analysis by applying different algorithms to various datasets. Understanding and studying issues in datasets can be known by predicting and analysing the data by using different techniques of machine learning. For analysing the data, choosing the environment is a big task because it provides tools and services for data processing, data querying, training and tuning model, authoring code, containerization application, code testing most importantly running the code smoothly. To figure out problems in datasets dissimilar algorithms were applied from machine learning in different environments to know the best environment and algorithm for analysing the data.

There are two types of environments to analyse the data first is Standalone Environment and second is Distributed Computing Environment. As a result, a new class of distributed computing platforms and software components have been developed using traditional distributed computing technology, making the implementation of big data analytics simpler.

Distributed computing environment works together with multiple computers to solve a common problem. Whereas Standalone Environment completes the program which resides on the computer for execution. In big data it is the main technology in both distributed computing and Standalone Environment because for storing and retrieving the data big data analytics is the best and easier way.

In this paper, data processing techniques are explored for parallel and distributed computing. Specifically, the algorithms have any time properties that can be interrupted while execution and it may generate a valuable result at the point of interruption. The generated results may have an increasing quality of prediction that is designed and implemented in both parallel and distributed algorithms for a selected problem.

Different sizes of large data sets are considered to predict in both Distributed Computing Environments and Standalone Environment by applying different kinds of Machine Learning Algorithms (MLA) and the best algorithm is known with their accuracies and time.

1.1 Big Data Analytics

Big data analytics is the practise of finding correlations, trends, patterns, and in massive amounts of raw data in directive to make decisions on data. It utilises further modern tools to analyse larger datasets using familiar statistical analysis techniques like regression and clustering due to its ability for massively aggregate and streamline data from numerous sources, the Spark environment is taken into consideration for such data analyses.

The spark is a distributed data processing engine that can be used in a variety of scenarios. There have been libraries for SQL, machine learning, graph computation, and streaming analytics on top of Spark that can be used in applications.

By leveraging memory computing and other optimizations. It can be hundred times faster than Hadoop for large-scale data processing. When data is stored on disc, it is quick and efficient. It is currently the world record holder for large-scale on-disk sorting. Using a Distributed Environment and a Personal Computing Environment in big data analytics is a logical next step for organisations looking to maximise their data's potential value. Machine learning tools analyse data sets that are processed using data-driven algorithms and statistical models to draw inferences or make predictions based on them.

1.2 Two Ways of Hadoop and Spark Integration

They are two methods in the Spark Hadoop Integration project.

a. Independence

Apache Spark and Hadoop can run jobs independently. It is simple and clear. Even when Spark is pulling data from HDFS based on business priorities**.**

b. Speed

Despite MapReduce, we can use Spark in Hadoop YARN. It allows for faster reads and writes from HDFS.

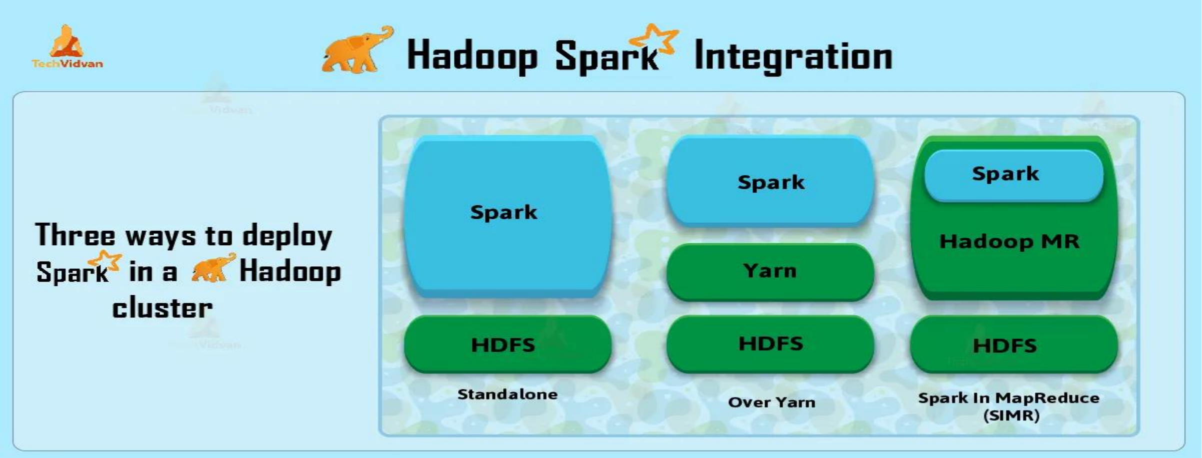


Figure 1: Hadoop spark integration (Google Courtesy)

2. RELATED WORKS

J. Magdum et al.[1] solved problems in distributed environment using Distributed machine learning algorithms and also huge datasets are designed using systems and MLA by multi node. The author used parallel computing environment to study and understand distributed MLA and applied on LightGBM, Cat Boost, AdaBoost, and XGBoost revealed that by analysing the accuracy of these algorithms CatBoost algorithm is the best algorithm with of 81.31%.

M. Khoshlessan et al.[2] described local intelligence and ability to host distributed energy resources, advanced micro-grids have been implemented as a crucial component in smart grids. It can make fast decisions, precise and fast decisions intelligent using machine learning algorithms, allowing it to achieve the aforementioned goals. The dataset used contains test and training data, weather, the state of charge (SOC) grid voltage of the storage system in each micro grid and its neighbouring unit in day time. Random Forests, Decision Trees, Logistic Regression, Gradient Boosting and SVM algorithms applied and compared in energy distribution.

J. Chen et al.[6] described the widespread applied use of decision tree algorithms in data mining industry. Big data platforms demonstrate the deployment of the distributed decision tree method. In a distributed setting, a decision tree is built using a novel KS-Tree technique. Modern decision tree implementations in R and Apache Spark are compared to KS-Tree as it is applied to several real-world data mining issues. Results indicate that KS-Tree can produce superior outcomes, particularly for large data sets.

S. Ramírez-Gallego et al.[4] identified mining massive and fast data streams as one of the major contemporary challenges in machine learning. A new distributed and incremental classifier adapted using nearest neighbour algorithm. To perform faster searches there is a distributed metric-space ordering in Apache Spark. It reduces the original classifier's high computational requirements, making it suitable for the considered problem.

M. Klymash et al.[17] discussed the challenges of processing large amounts of data in databases in order to execute user queries more efficiently. It enables distributed machine learning to analyse large amounts of data more quickly. A change to the distributed database system architecture ensures that machine learning methods are used effectively in software modelling of data array processing using distributed machine learning. The results show that the distributed machine learning method improves the efficiency of processing large amounts of information in databases.

3. DATA SETS

3.1 Login Data Set For Risk Base Authentication

One TeraByte of data is taken from a large-scale online service across the globe with over 33 million login attempts and 3.3 million users. Data were gathered between February 2020 and February 2021. The goal of these data sets is to accelerate research and development for Risk-Based Authentication systems. The information was derived from the real-world login behavior of over 3.3 million users at a large-scale single sign-on online across the globe.

3.2 Internet Traffic Management System

Systems collect real-time traffic data and take the necessary steps to reduce internet traffic congestion. The scores of Unified Traffic Management system and the protocols like IP, UDP, TCP, HTTP, FTP, DNS, and TFTP are captured and the best score is given for proper maintenance. The scores are matched with Distributed Machine learning Algorithms. A medical recommendation system is recommending the doctors for a particular disease based on patient reviews. It is very essential in the fast-growing technological world, which can save the lives of many patients. Rating will be given by the patients based on the performance of doctors.

4. METHODOLOGY

4.1 Machine Learning Algorithms in Distributed Environment

Three different datasets of different sizes are considered for analysis in Distributed Environment and Standalone Environment by applying various Linear and Nonlinear models. Linear model single columns are targeted based on yes or no. Nonlinear models focused on multiple parameters with multiple targets. Machine learning algorithms like Random Forest, K-Nearest Neighbors, Naïve Bayes and Decision tree come under linear Models. While Ada Boost, cat Boost. XG Boost comes under nonlinear models. The methodology shown in figure 2.

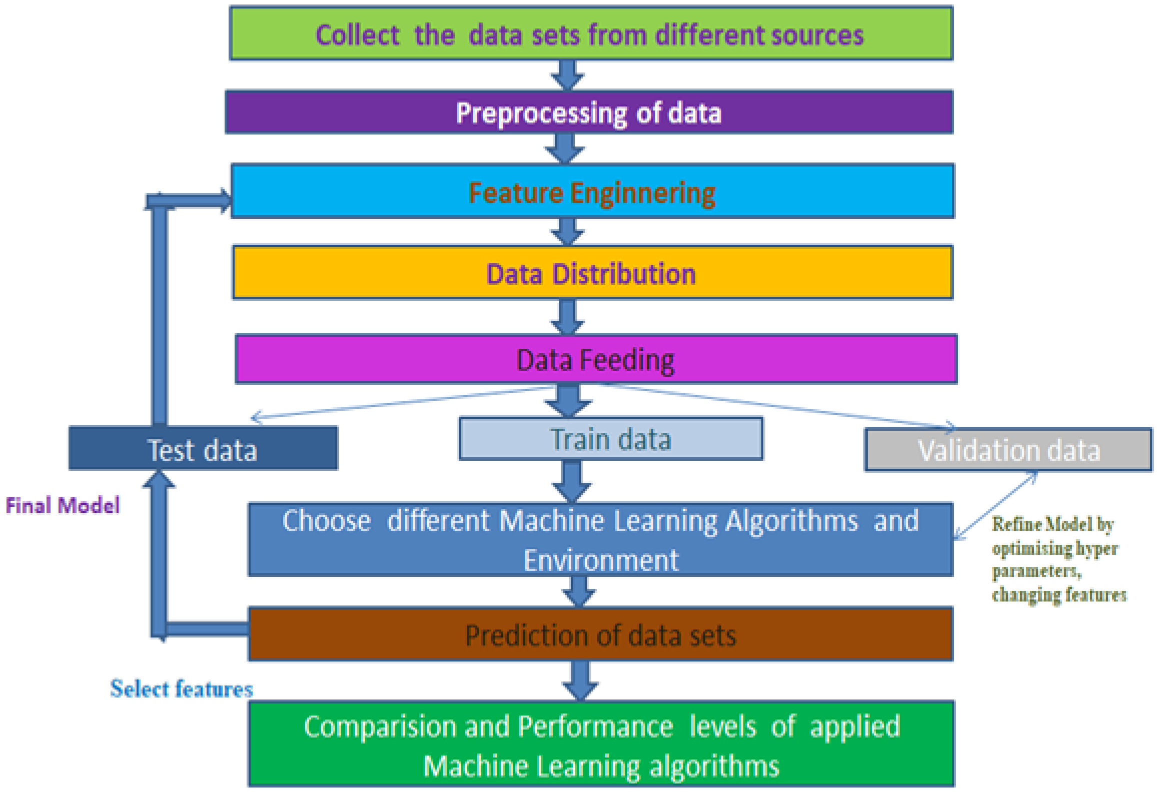

Collect the data sets from different sources
Preprocessing of data
Feature Enginnering
Data Distribution
Data Feeding
Test data
Train data
Validation data
Final Model
Choose different Machine Learning Algorithms and Environment
Refine Model by optimising hyper parameters, changing features
Prediction of data sets
Select features
Comparision and Performance levels of applied Machine Learning algorithms

Figure 2: Work Flow Procedure in Distributed Environment and Standalone Environment

4.1 K-Nearest Neighbors

An unsupervised algorithm where the behaviour of the data point itself can be inferred from its neighbours. At each point, the query set matches the training set of the nearest neighbour at a distance of zero, allowing a sparse graph depicting the relationships between neighbouring points to be generated. The dataset is organised in index order and parameter space, resulting in an approximately block-diagonal matrix of K-nearest neighbours. A sparse graph is useful for unsupervised learning of spatial relationships between points in a variety of situations. A simple majority vote of the algorithm determines the value assigned to a query point. In some cases, weighing the neighbours is preferable so that closer neighbours contribute more to the fit.

To determine the class of a data point, the KNN classifier employs the majority voting principle. When k value is set to five, the five closest points' classes are examined to make predictions. Similarly, the mean of the five closest locations is used in kNN regression. The distance between data points is determined. Distance can be calculated in a variety of ways. The Euclidean distance (Minkowski distance with p=2) is one of the most widely used measurement units for distance. It demonstrates how to compute the Euclidean distance between two points in two dimensions and square the difference between the locations' x and y coordinates.

Euclidean distance is

$$\text{Euclidean (A,B)} = \sqrt{(x2 - x1\ )2 + (y2 - y1)2}$$

The first step is to import the modules and creating a data set for visualization. It is critical to divide a dataset into test and training sets for any supervised machine learning method. Train it first Prior to testing the model on various dataset segments. The model is only tested with pre-existing data if the data is not separated. It can simply separate the tests using the train test split method and using the train and test dataset it can determine the extent to which the original data is used. The default separation for the train set is 75% and for the test set is 25%, and KNN classifiers are used for prediction, accuracy and time.

4.2 Random Forest Algorithm

It is well-known supervised learning algorithm based on trees. It is the most adaptable and user-friendly ideal solution. The algorithm is applicable to classification and regression problems. Before training each decision tree on a different sample of the observations, random forest typically combines hundreds of decision trees. To produce the final predictions of each individual tree are average to prefer overfit by training data for a variety of reasons, but this difficulty can be mitigated by averaging the prediction outcomes of multiple trees. As a result, in terms of predictive accuracy, random forests outperform single decision trees. By subtracting node impurity from probability of reaching node determines the feature importance and calculated node probability by dividing the total reach node by number of samples .The more feature is significant the value will be high. In the Spark implementation, each decision tree evaluates the importance of a feature by calculating the gain, scaled by the number of samples passing through the node.

$$fi_i = \sum_{j:nodes\ j\ splits\ on\ feature\ i} s_j C_j$$

Where fi sub (i) = the importance of feature I,s sub(j) = a number of samples reaching node j, C sub(j) = the impurity value of node j.

It draws random samples from the user-supplied dataset. For each sample chosen for creation, the algorithm will generate a decision tree, and a prediction result will be obtained for each predicted outcome. It uses mode for classification problems and mean for regression problems. The prediction result that receives the highest votes will be chosen by the algorithm. Finally, we compute prediction, accuracy, and time.

4.3 Decision Tree

It is a well-known and widely used supervised machine learning method for classification and regression problems. The logic is relatively simple and ability to comprehend. Each attribute in dataset is represented by a node .The most important attribute belongs to the root node to evaluate the project it begins at the top of the tree and make the way down, followed by the node that corresponds to condition or decision. Repeat the procedure until reaches a leaf node which contains Decision Tree's prediction or outcome for the information contained by each attribute is estimated. The amount of impurity in a given dataset is measured by the randomness or uncertainty of a random variable X. It refers to impurity in a group of examples in information theory. The difference in entropy between before and after splitting is shown below for computed information gain based on given attribute values.

$$Entropy = \sum_{i=1}^{C} -p_i * \log_2(p_i)$$

Here, c is the number of classes and pi is the probability associated with the ith class. Importing dataset for exploratory data analysis is done by renaming the column names. The frequency distribution of values in variables and feature vectors with respect to the target variable is Splitted into training and testing sets. Feature Engineering is done by Encoding categorical variables with Decision Tree Classifier entropy for Predicting Test Set Results and Calculating Performance of Accuracy Score with Time.

4.4 Adaboost

AdaBoost is the initialism for adaptive boosting. It is the first truly effective binary classification boosting algorithm that has been developed to understand. Furthermore, modern boosting techniques are notably stochastic gradient boosting machines.Short decision trees are frequently combined with the first tree and then made on each training instance's performance and used to gauge how much attention the next tree should receive. As a result, it is designed to focus on each training instance. As a result, difficult-to-predict training data is given more weight. However, instances that are easily predicted are given less weight .Weight (xi) = 1/n is set as the initial weight for each instance in the training dataset.

Where xi represents the i^{th} training instance and n represents the total number of training instances. Weighted samples are used to train a weak classifier using the training set of data. Only Problems of binary classification are supported. Because of this, each decision stump considers only one option and provides a value of +1.0 or-1.0 based on the positive or negative value of the first or second class. The trained model's misclassification rate is computed. This is typically calculated using the formula error = (correct - N) / N.

Error represents the frequency of misclassification despite the model's correct prediction of the number of training instances; N denotes the total number of training instances. Weak models are incrementally added and trained with weighted training data. Typically, the process is repeated until a certain number of poor learners are produced. When the work is completed, a pool of weak learners for each stage value is formed. The weighted average of weak classifiers is used to make predictions. If the sum is positive, the first class is predicted; if it is negative, the second class is predicted, as is the accuracy over time.

4.5 XG Boost

XG Boost algorithm is a predictive modelling problems for Classification and Regression(CART)algorithm and it is build based on decision tree algorithm .CART trees can be used to demonstrate operation of algorithm's Here root node contains a variable with single input x and a partition variable. Whereas leaf node contains variable output y which is used for prediction. A tree can be thought of as set of rules or graph. As a result, using a tree to predict becomes relatively simple. The traversed tree is compared with a particular root node in a input tree when a new input is given. Based on input tree is significantly splitted into sections when repeating variables is reached to leaf node from two or more homogeneous sets. Each node chooses variable centred on predicted data. The hierarchical model is made with a set of questions are asked with many observation; yield either value or a class label. It operates in the same manner as protocol, with series of 'if this happens, then this happens' conditions that gives a specific result based on given data. Despite fact overfitting outperforms other methods in gradient-boosting models for regression and classification predictive modelling problems on structured or tabular datasets. It is also very accurate, but the accuracy decreases when parameters numbers are increased. For future use the data matrix model can be saved and loaded many times. The property is useful for dealing large and complex datasets. It was meticulously designed with system optimization.

Weights are important in this algorithm's ability to generate sequential decision trees. Each independent variable is given a weight, and the decision tree uses these weights to predict the results. Increases in the weight of a tree's variables that are incorrectly predicted are fed into the second decision tree. The separate predictors and classifiers are then combined to create a more powerful and accurate model. Problems with regression, classification, ranking, and user-defined prediction can all be solved. The prediction scores of each single decision tree are added together to produce, for example, two trees that attempt to complement one another while also calculating accuracy and time performance levels. The model can be written mathematically in the form of $\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), f_k \epsilon \mathcal{F}$

Where, K represents number of trees, f derives functional space of F, and F is the set of possible CARTs. The objective function for the above model is given by:

$obj(\theta) = \sum_{i}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$

4.6 Cat Boost

CatBoost algorithm is an effective technique for Gradient Descent-based supervised machine learning tasks. It is a popular algorithm for regression and classification tasks and is ideal for categorical data problems. It is based on gradient decision trees and generates a sequence of decision trees when trained. Each subsequent tree is built with a lower loss than the previous tree as training progresses.

It is essential to prepare the model's data before training the machine learning model. So the next few steps are as follows: First, eliminate the Survived column since it will serve as the target variable. Then divide the data into two Data Frames, such as x and y, with the target variable in one and the useful model features in the other. Then change “PClass" column to a string data type and populate features with null values.

It is a powerful and effective categorical feature machine learning algorithm; in this case, two functions are used to produce a list of column indicates categorical data. After which all columns must be changed to the category data type .20% of the data will be used for testing and 80% for training. After training, we can make predictions about the output values, accuracy, and timing using the testing data.

4.7 Naive Bayes Algorithm

Bayes Theorem is the basis for the Naive Bayes classification method which is most realistic supervised learning algorithm. It is a dependable, quick, and accurate method having large datasets; high accuracy and speed are required. The Naïve Bayes classifier makes the assumption which affects each feature within a class and is independent to one another. These features are assessed independently even though they are interdependent. Because it makes computation easier, this presumption is deemed naive. The term "class conditional independence" describes this.

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y)P(x_1, \ldots, x_n \mid y)}{P(x_1, \ldots, x_n)}$$

Estimate P(y) and P (xiy) using Maximum A Posteriori (MAP) estimation; the former is relative frequency of class y in the training set. Dataset is then imported for exploratory data analysis, feature vectors are defined for output targeting, and data is splitted o training and test sets. For Feature Scaling and Training the Feature Engineering is utilised to predict outcomes and track the accuracy score with time.

5. EXPERIMENTAL ANALYSIS

For data analysis Spark is used in Distributed Environment spark is used whereas, in a personal computing environment python is used. For each dataset pre- processing is done to clean the data for prediction of different algorithms. In data cleaning, multiple sub categories are involved. First one is the column by renaming it we find removing of the null columns and checking duplicate records by changing the data frame from strings to numeric. Next feature Engineering is performed for finding the input features because to know which data is suitable for each column and is passed to the machine algorithms in Distributed Environment and Personal computing environment.

In Data Exploration we are doing data sampling which means

- How the data is
- Finding the data types
- Finding the source and targets parameters

In Data Distribution the data is coming in huge volumes and data is processed chunk by chunk by taking one Tera Byte of data every time. The Machine Learning algorithms applied to the trained and tested data to find out based on test data of different data sets the accuracy, precision, Recall, F1-score, time to train, time to predict, total time and discrimination.

6. RESULT ANALYSIS

6.1 Data Set: Login Data Set For Risk Base Authentication

By applying Machine Learning Algorithms in Distributed Environment and Standalone Environment on the login data set for risk base authentication by predicting the data set of entire global data getting more attacks at windows OS in United States as well as in India. In Distributed environment XG Boost is Performing best accuracy with 91.22% with time of one second. whereas in the personal computing environment is Decision Tree Algorithm performed the best accuracy 91.19% with time of 7 seconds.

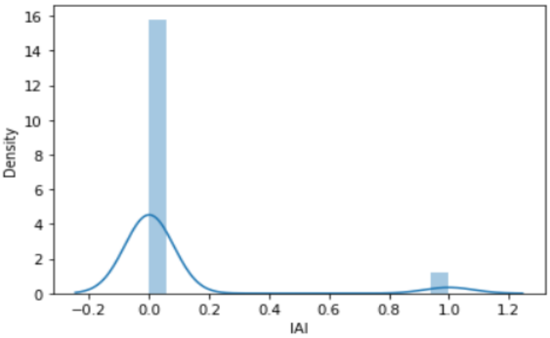


Figure 3: Pre Predicting No .of Attacks going to happen in IP

In above figure 3 the x axis represents Is Attack IP, y axis represents Density of internet attacks happen in a certain density of every 15seconds across the globe is predicted. Next, the attacks are predicted based on User_ID and OS Name and version.

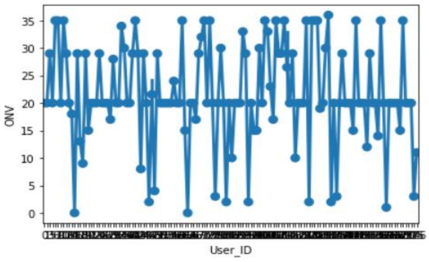


Figure 4: User relationship with attacks at OS Name and Version

In above figure 4 the x axis represents User_ID, y axis represents OS Name and Version number of users getting tracked on various environments (operating systems) are predicted.

Table 5: The Performance analysis of Machine Learning Algorithms in Distributed Environment and Standalone Environment are shown in table 1 and 2 of login data set for risk base authentication.

Machine learning Algorithms in distributed environment -	Accuracy	Recall	Precision	F1-Score	Time to train in min	Time to prediction in sec	Total time in sec

spark							
KNN	88.89%	88.89%	79.01%	83.66%	1.0	3.0	4.0
Random Forest	92.22%	92.22%	92.85%	90.29%	5.4	1.1	1.4
Decision Tree	91.11%	91.11%	91.92%	88.36%	1.5	2.0	3.0
Naïve Bayes	77.78%	77.78%	89.53%	81.48%	2.6	2.0	4.0
Ada Boost	91.11%	91.11%	89.99%	89.37%	1.3	1.0	1.3
XG Boost	92.22%	92.22%	92.85%	90.29%	1.3	2.0	1.0
Cat Boost	88.89%	88.89%	79.01%	83.66%	1.8	1.0	0.8
Machine learning Algorithms in Personal computing Environment	Accuracy	Recall	Precision	F1-Score	Time to train in sec	Time to predict in sec	Total time in sec
KNN	67.19%	56.09%	56.39%	69.89%	8.0	6.0	14.0
Random Forest	81.19%	80.09%	82.39%	87.89%	3.0	9.0	8.0
Decision Tree	91.19%	90.09%	92.39%	96.79%	2.0	1.0	7.0
Naïve Bayes	71.19%	70.09%	71.24%	75.68%	3.5	5.0	5.0
Ada Boost	91.51%	92.50%	91.24%	91.57%	4.0	85.0	85.0
XG Boost	85.51%	80.50%	81.24%	80.16%	0.0	85.0	85.0
Cat Boost	87.51%	86.50%	75.12%	77.02%	9.0	71.0	80.0

6.2 Data Set: Internet Traffic Management System

In prediction, HTTP is properly connected in the US, and a number of more dropout issues and dropout connections are happening in Saudi Arabia. In a Distributed environment, ADAboost Algorithm performed best accuracy with 96.79 % with time of 23 seconds. Whereas in a personal computing environment is CAT Boost Algorithm performed the best accuracy with 97.37 % with time of 47 seconds.

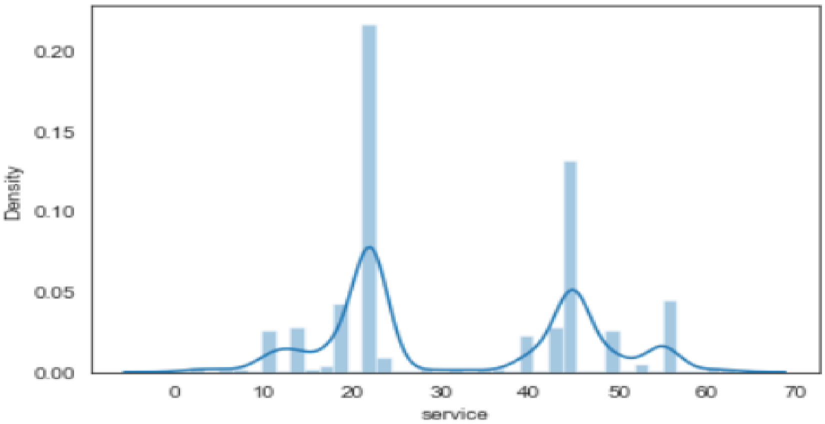


Figure 6: Average drops happening in service

By analysing above figure 6 the x axis represents Service, y axis represents Density of various services which are providing internet connectivity in n no of cities.

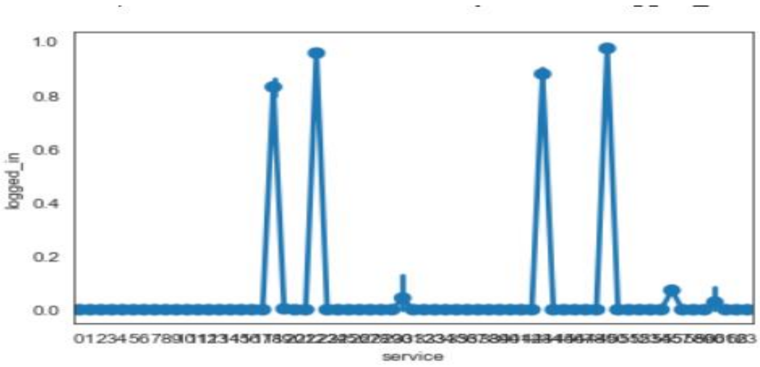


Figure 7: User relationship with service and no of Logged in

By analysing the above figure 7 the x axis represents Service , y axis represents Logged in happened on various services which are providing internet connectivity in n no of cities.

Table 8: The Performance analysis of Machine Learning Algorithms in Distributed Environment and Standalone Environment are shown in table 1 and 2 of Internet Traffic Management System data set

Machine learning Algorithms in distributed environment - spark	**Accuracy**	**Recall**	**Precision**	**F1-Score**	**Time to train in sec**	**Time to predict in sec**	**Total time in sec**
KNN	81.65%	81.65%	75.38%	77.98%	0.0	3.7	3.7
Random Forest	82.17%	82.17%	74.94%	77.92%	1.2	0.1	1.3
Decision Tree	85.36%	85.36	72.87%	78.62%	3.0	2.0	5.0

		%					
Naïve Bayes	70.98%	70.98%	86.70%	74.27%	3.5	4.0	7.5
Ada Boost	96.79%	96.79%	96.91%	96.83%	2.1	0.2	2.3
XG Boost	81.55%	81.55%	74.64%	77.59%	1.8	0.0	1.8
Cat Boost	85.33%	85.33%	72.87%	78.61%	1.0	3.0	4.0
Distributed machine learning Algorithms in Standalone Environment	**Accuracy**	**Recall**	**Precision**	**F1-Score**	**Time to train in sec**	**Time to predict in sec**	**Total time in sec**
KNN	81.92%	81.44%	81.45%	87.13%	25.0	20.0	45.0
Random Forest	95.92%	95.44%	95.45%	99.71%	23.0	21.0	44.0
Naïve Bayes	65.92%	65.44%	69.54%	69.97%	16.0	19.0	35.0
Ada Boost	91.73%	91.65%	95.45%	99.71%	32.0	17.0	49.0
XG Boost	93.73%	93.17%	96.45%	99.71%	16.0	11.0	27.0
Cat Boost	97.37%	97.17%	98.45%	99.13%	28.0	19.0	47.0

6.3 Data Set: Medical Recommendations System

In prediction, some patients are satisfied whoever having the personal clinic and good, the specialization of doctors, and some patients are not satisfied due to Personal reasons. Every year in India Maharashtra and Gujarat states people suffer from main diseases like Diabetes and long term problems like cancer.

In Distributed environment Cat boost Algorithm is performing best accurate with 97.51% with time of 8 seconds .whereas in the personal computing environment is KNN Algorithm had best accuracy with 96.06 % with time of 29 seconds.

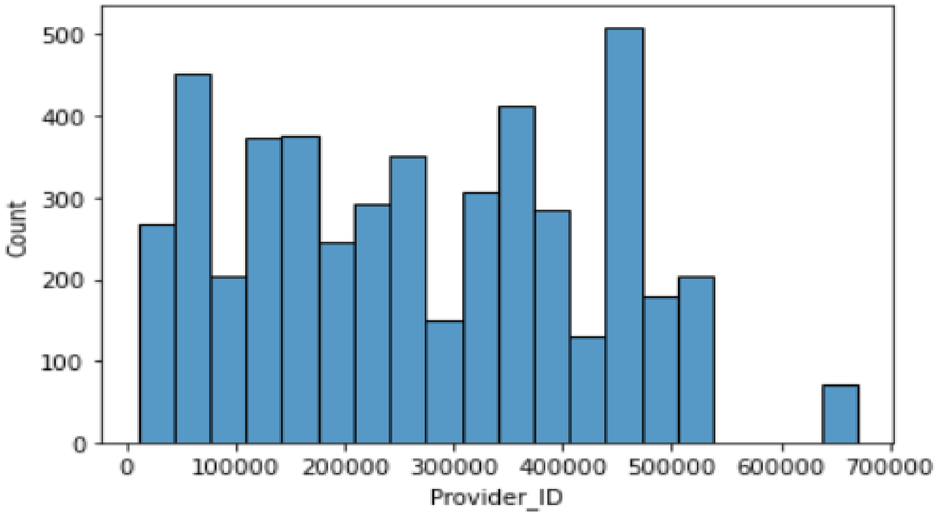


Figure 9: Medical services providers in India

By analysing the above figure 9 the x axis represents Provider_ID, y axis represents count of various medical vendors.

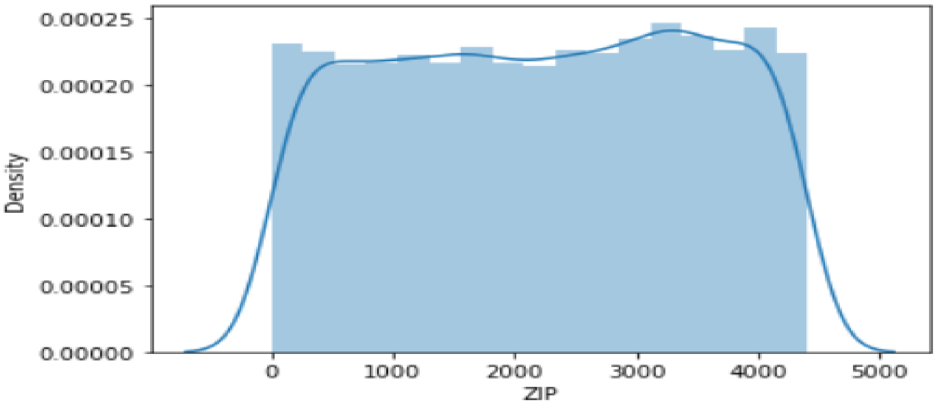


Figure 10: Area wise clinics in India

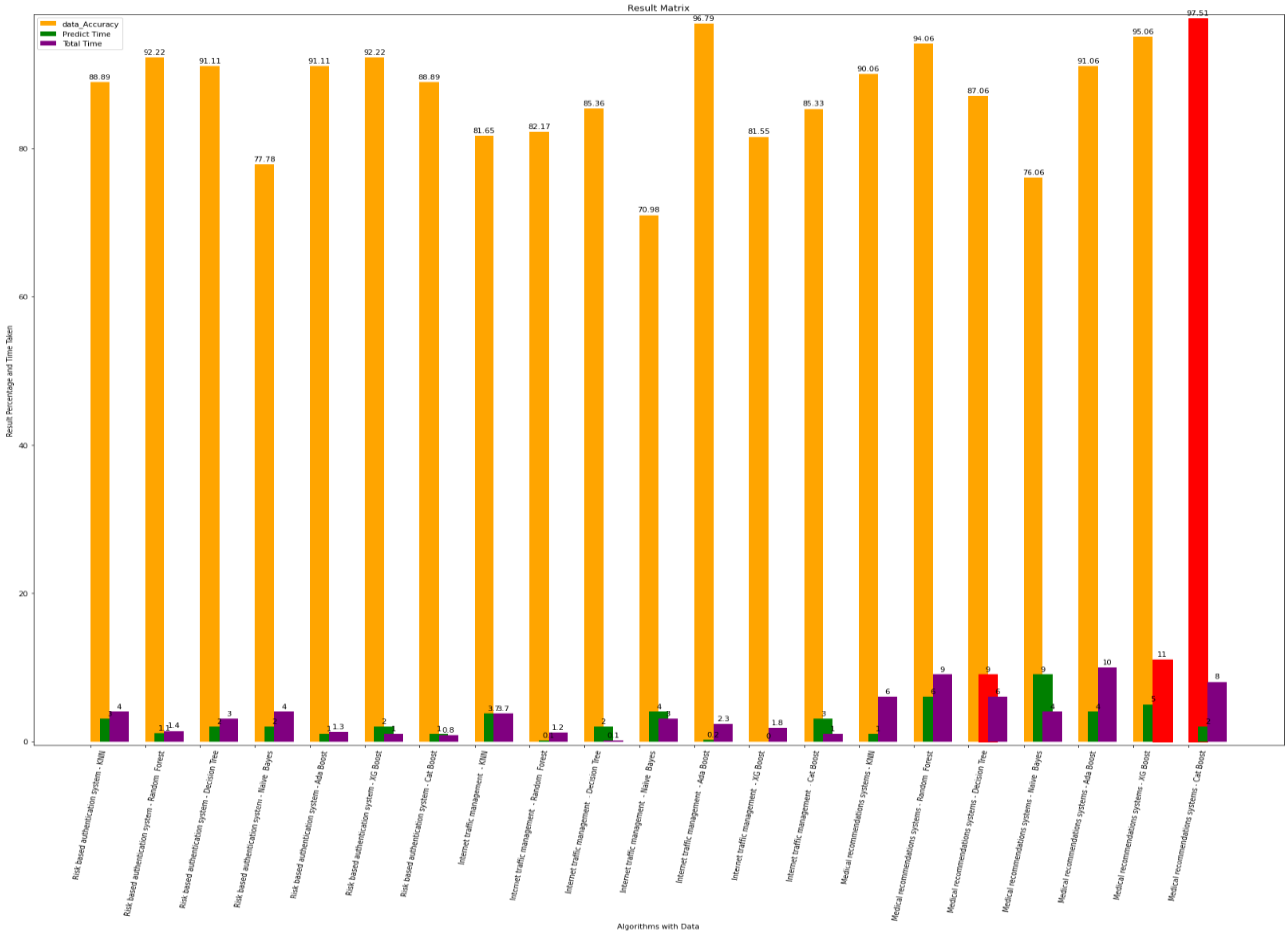
By analysing the above figure 10 the x axis represents ZIP, y axis represents Density of various vendors which are providing medical solutions in n no of cities.

Table 11: The Performance analysis of Machine Learning Algorithms in Distributed Environment and Standalone Environment are shown in table 1 and 2 of Medical Recommendation system data set.

Machine learning Algorithms in distributed environment – spark	**Accuracy**	**Recall**	**Precision**	**F1-Score**	**Time to train in sec**	**Time to predict in sec**	**Total time in sec**
KNN	90.06%	90.06 %	90.86%	91.06%	5.0	1.0	6.0

Random Forest	94.06%	94.01%	94.09%	96.06%	3.0	6.0	9.0
Decision Tree	87.06%	87.06%	87.06%	96.06%	3.0	9.0	6.0
Naïve Bayes	76.06%	74.06%	76.86%	79.61%	5.0	9.0	4.0
Ada Boost	91.06%	91.06%	91.69%	96.96%	6.0	4.0	10.0
XG Boost	95.06%	95.56%	95.56%	98.96%	6.0	5.0	11.0
Cat Boost	97.51%	97.56%	97.56%	97.90%	6.0	2.0	8.0
Machine learning Algorithms in Standalone Environment	**Accuracy**	**Recall**	**Precision**	**F1-Score**	**Time to train in sec**	**Time to predict in sec**	**Total time in sec**
KNN	96.06%	94.06%	96.86%	95.06%	13.0	16.0	29.0
Random Forest	93.06%	93.06%	93.86%	90.56%	48.0	11.0	59.0
Decision Tree	94.26%	94.41%	94.45%	91.26%	9.0	10.0	19.0
Naïve Bayes	77.26%	79.44%	77.44%	79.13%	29.0	20.0	49.0
Ada Boost	95.73%	95.44%	94.44%	98.13%	21.0	18.0	39.0
XG Boost	97.73%	97.44%	96.44%	10.01%	31.0	15.0	46.0
Cat Boost	91.73%	91.44%	87.44%	99.13%	45.0	11.0	56.0

Time



Accuracy, Prediction

Figure 12: Visualization of Machine learning algorithms in Distributed Computing Environment.

By observing figure 12, X axis represents Accuracy, Prediction, and y axis represents Time to compare and know the best algorithm in machine learning. The Ada boost algorithm performs with high accuracy of 97.51%, total time is 8 seconds. Whereas Naïve Bayes Algorithm performs is low with an accuracy of 70.98%, and total time taken is 7 seconds.

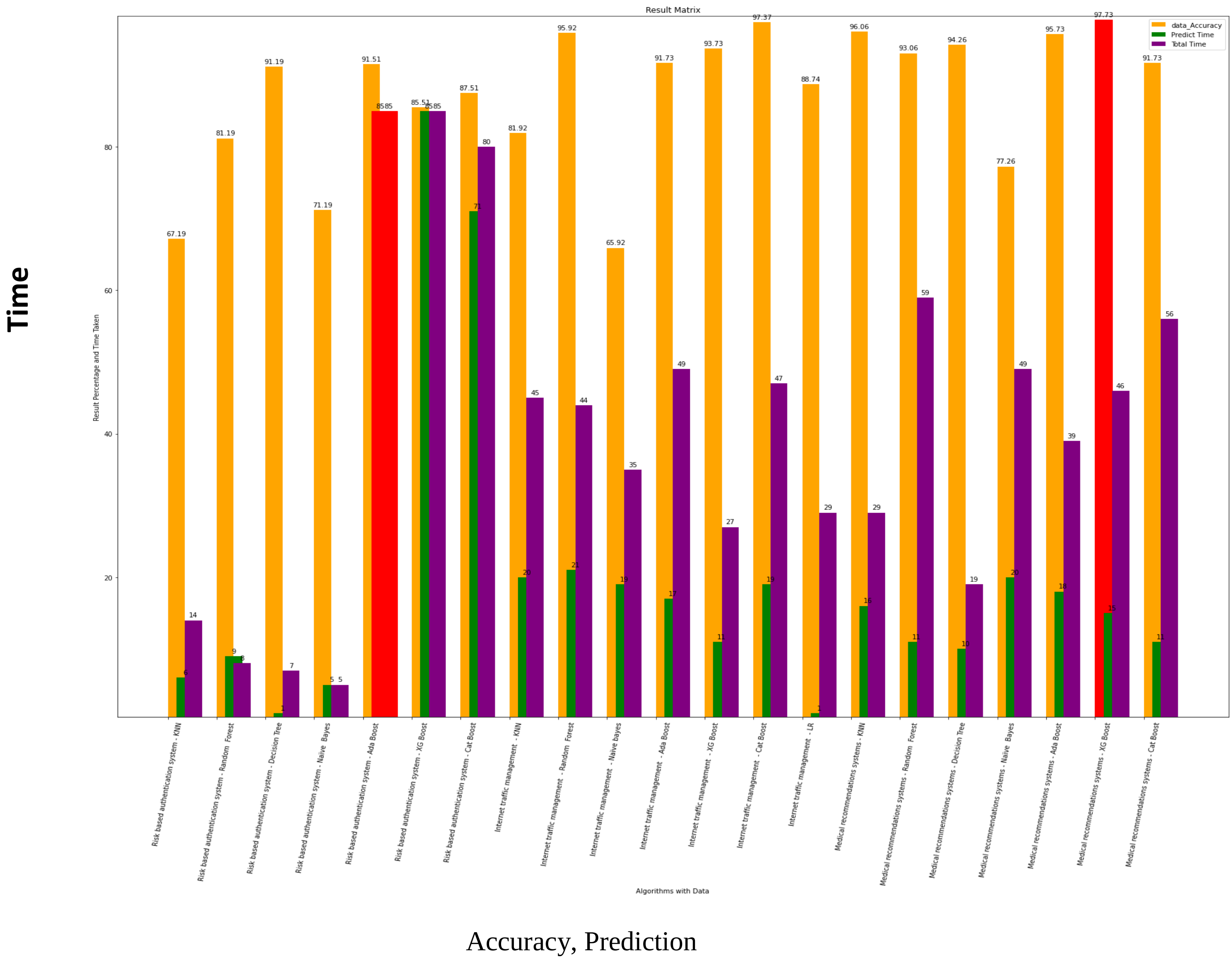


Figure 13: Visualization of machine learning algorithms in Standalone Environment.

By observing figure 13, X-axis represents Accuracy, Prediction, and y-axis represents Time to compare and know the best algorithm in machine learning. The Ada boost algorithm performs with high accuracy of 97.37%, total time is 47 Seconds. Whereas Naïve Bayes Algorithm performs low with an accuracy of 65.92% and the total time taken is 35 seconds.

7. CONCLUSION

A complete overview of Machine Learning algorithms based on distributed machine learning Environment and Standalone Environment are discussed and used for analysing data. K-Nearest Neighbour, Random Forest, Naïve Bayes, AdaBoost, XgBoost, CatBoost, and Decision Tree algorithms are compared with seven evaluated measures for access like Accuracy, Precision, Recall, F1 score time to train, time to predict, and total time considered. It defines the concept of algorithms related to the data sets used for seven measures to demonstrate both measures influence the accuracy outcomes in a way that equivalent to the pervasive understanding of the concept. By observing the accuracies of all three data sets in applied Machine Learning Algorithms Cat Boost algorithm with an accuracy of 97.51%, a total time of 8 seconds, and an accuracy of 97.37%, a total time of 47 seconds is considered best for prediction in both distributed environment and Standalone Environment.

REFERENCES

1. J. Magdum, R. Ghorse, C. Chaku, R. Barhate and S. Deshmukh, "A Computational Evaluation of Distributed Machine Learning Algorithms," 2019 IEEE 5th International Conference for Convergence in Technology (I2CT), 2019, pp. 1-6, doi: 10.1109/I2CT45611.2019.9033733.
2. M. Khoshlessan, B. Fahimi and M. Kiani, "A comparison between Machine learning algorithms for the application of micro-grids Energy management," 2020 IEEE International Conference on Industrial Technology (ICIT), 2020, pp. 805-809, doi: 10.1109/ICIT45562.2020.9067203.
3. Y. Li, "Research on big data analysis and processing system based on Spark platform," 2022 International Conference on Machine Learning and Intelligent Systems Engineering (MLISE), 2022, pp. 263-266, doi: 10.1109/MLISE57402.2022.00059.
4. S. Ramírez-Gallego, B. Krawczyk, S. García, M. Woźniak, J. M. Benítez and F. Herrera, "Nearest Neighbor Classification for High-Speed Big Data Streams Using Spark," in IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 47, no. 10, pp. 2727-2739, Oct. 2017, doi: 10.1109/TSMC.2017.2700889.
5. W. Ai, K. Li, C. Chen, J. Peng and K. Li, "DHCRF: A Distributed Conditional Random Field Algorithm on a Heterogeneous CPU-GPU Cluster for Big Data," *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, 2017, pp. 2372-2379, doi: 10.1109/ICDCS.2017.66
6. J. Chen, T. Wang, R. Abbey and J. Pingenot, "A Distributed Decision Tree Algorithm and Its Implementation on Big Data Platforms," *2016 IEEE* International Conference on Data Science and Advanced Analytics (DSAA), 2016, pp. 752-761, doi: 10.1109/DSAA.2016.64.
7. M. Drakoulelis, G. Filios, V. Georgopoulos Ninos, I. Katsidimas and S. Nikoletseas, "Virtual Light Sensors in Industrial Environment Based on Machine Learning Algorithms," *2019 15th* International Conference on Distributed Computing in Sensor Systems (DCOSS*)*, 2019, pp. 709-716, doi: 10.1109/DCOSS.2019.00126.
8. F. Yuan, F. Lian, X. Xu and Z. Ji, "Decision tree algorithm optimization research based on MapReduce," *2015 6th* IEEE International Conference on Software Engineering and Service Science (ICSESS), 2015, pp. 1010-1013, doi: 10.1109/ICSESS.2015.7339225.
9. B. Jia, M. Wu, B. Li, Y. Yu, N. Zhang and G. Ma, "Perceptual Forecasting Model of Power Big Data Based on Improved Random Forest Algorithm," *2022* International Conference on Machine Learning and Intelligent Systems Engineering *(MLISE)*, 2022, pp. 267-270, doi: 10.1109/MLISE57402.2022.00060.
10. W. S. Albaldawi, R. M. Almuttairi and M. E. Manaa, "Big Data Analysis for Healthcare Application using Minhash and Machine Learning in Apache Spark Framework," *2022* International Congress on Human-Computer Interaction, Optimization and Robotic Applications *(HORA)*, 2022, pp. 1-7, doi: 10.1109/HORA55278.2022.9799934.

11. Z. Huang *et al*., "A Distributed Computing Framework Based on Variance Reduction Method to Accelerate Training Machine Learning Models," *2020 IEEE International Conference on Joint Cloud Computing*, 2020, pp. 30-37, doi: 10.1109/JCC49151.2020.00014.
12. U. Dixit, S. Bhatia and P. Bhatia, "Comparison of Different Machine Learning Algorithms Based on Intrusion Detection System," *2022 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COM-IT-CON)*, 2022, pp. 667-672, doi: 10.1109/COM-IT-CON54601.2022.9850515.

A. Sheshasaayee and J. V. N. Lakshmi, "An insight into tree based machine learning techniques for big data analytics using Apache Spark," *2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*, 2017, pp. 1740-1743, doi: 10.1109/ICICICT1.2017.8342833.

13. ZheHuang Huang and Xiaodong Shi, "A distributed parallel AdaBoost algorithm for face detection," *2010 IEEE International Conference on Intelligent Computing and Intelligent Systems*, 2010, pp. 147-150, doi: 10.1109/ICICISYS.2010.5658736.
14. W. Y. Al-Rashdan and A. Tahat, "A Comparative Performance Evaluation of Machine Learning Algorithms for Fingerprinting Based Localization in DM-MIMO Wireless Systems Relying on Big Data Techniques," in IEEE Access, vol. 8, pp. 109522-109534, 2020, doi: 10.1109/ACCESS.2020.3001912.
15. J. Chandrasekaran, H. Feng, Y. Lei, R. Kacker and D. R. Kuhn, "Effectiveness of dataset reduction in testing machine learning algorithms," *2020 IEEE International Conference On Artificial Intelligence Testing (AITest)*, 2020, pp. 133-140, doi: 10.1109/AITEST49225.2020.00027.
16. M. Klymash, M. Kyryk, I. Demydov, O. Hordiichuk-Bublivska, H. Kopets and N. Pleskanka, "Research on Distributed Machine Learning Methods in Databases," 2021 IEEE 4th International Conference on Advanced Information and Communication Technologies (AICT), 2021, pp. 128-131, doi: 10.1109/AICT52120.2021.9628949.
17. G. D. Kalyankar, S. R. Poojara and N. V. Dharwadkar, "Predictive analysis of diabetic patient data using machine learning and Hadoop," 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), 2017, pp. 619-624, doi: 10.1109/I-SMAC.2017.8058253.
18. Y. Xie, D. Feng, Y. Hu, Y. Li, S. Sample and D. Long, "Pagoda: A Hybrid Approach to Enable Efficient Real-Time Provenance Based Intrusion Detection in Big Data Environments," in IEEE Transactions on Dependable and Secure Computing, vol. 17, no. 6, pp. 1283-1296, 1 Nov.-Dec. 2020, doi: 10.1109/TDSC.2018.2867595.
19. W. Zhang, X. Chen, Y. Liu and Q. Xi, "A Distributed Storage and Computation k-Nearest Neighbor Algorithm Based Cloud-Edge Computing for Cyber-Physical-Social Systems," in IEEE Access, vol. 8, pp. 50118-50130, 2020, doi: 10.1109/ACCESS.2020.2974764.
20. R. Gu, S. Fan, Q. Hu, C. Yuan and Y. Huang, "Parallelizing Machine Learning Optimization Algorithms on Distributed Data-Parallel Platforms with Parameter Server," 2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS), 2018, pp. 126-133, doi: 10.1109/PADSW.2018.8644533.

21. K. Rajeshkumar, S. Dhanasekaran and V. Vasudevan, "Applications of Machine Learning Algorithms for HDFS Big Data Security," 2022 International Conference on Computer Communication and Informatics (ICCCI), 2022, pp. 1-5, doi: 10.1109/ICCCI54379.2022.9740908.
22. J. Yang, X. Meng and M. W. Mahoney, "Implementing Randomized Matrix Algorithms in Parallel and Distributed Environments," in Proceedings of the IEEE, vol. 104, no. 1, pp. 58-92, Jan. 2016, doi: 10.1109/JPROC.2015.2494219.
23. M. Zhou and W. Ai, "Distributed Reduced Kernel Extreme Learning Machine," *2021* China Automation Congress (CAC*)*, 2021, pp. 3384-3387, doi: 10.1109/CAC53003.2021.9728238.
24. D. Tian, "Simulation of Distributed Big Data Intelligent Fusion Algorithm Based on Machine Learning," *2022* International Conference on Artificial Intelligence and Autonomous Robot Systems (AIARS*)*, 2022, pp. 421-424, doi: 10.1109/AIARS57204.2022.00101.