Journal of Theoretical and Applied Information Technology
15th July 2023. Vol.101. No 13

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

ESTIMATION OF MINIMAL INITIAL SAMPLE SIZE IN PROGRESSIVE SAMPLING FOR BIG DATA ANALYTICS

YATHISH ARADHYA B C[1], DINESHA H A[2], LOKESH M R[3]

[1] Research Scholar, VTU-RRC, Department of Computer Science and Engineering, Belagavi, India

[2]Professor, SIET, Department of Computer Science and Engineering, Tumkur, India

[3]Professor, Vivekananda College of Engineering & Technology (VCET), Puttur, Department of Computer Science and Engineering, Visvesvaraya Technological University (VTU), Belgaum, Karnataka., India

E-mail: [1]docs.kit@gmail.com, [2]sridini@gmail.com, [3]lokeshmrmysore@gmail.com

ABSTRACT

Big data are vectored, high dimensional and voluminous. Sampling such data is daunting task. Progressive sampling is the solution for such data. However initial sample size of the progressive sampling technique plays a vital role in the overall computational time and convergence time of any classifier. In Progressive Sampling Algorithm (PSA) a number of times iterative computation runs will be accountable for the total time cost of the sampling and indirectly to the time required for convergence of the classifier to learn a hypothesis. All existing works on minimal sample size estimation are not appropriate to carry out in the Distributed File system like Hadoop. In this work we present a novel statistically optimal sample size technique and its analysis, to estimate the initial minimal sample size for big data in an HDFS environment. Heterogeneous big data datasets were experimentally used to estimate initial sample size in a Hadoop environment with the analysis of computational time and space complexity in all degrees of freedom along with the convergence of the learning algorithm. If the initial sample size were accurately estimated, then there will be a substantial reduction in PSA. Thus providing a proper initial sample size for PSA will ensure optimally fast learning of the classifier in Information Technology applications for substantial prediction and assessments thus leading to robust software performance.

Keywords: *Progressive Sampling Algorithm (PSA), PAC Framework, Big Data, Sample Size, Initial Sample Size*

1. INTRODUCTION

Sampling in Big data with various parameters and metrics is tedious task. Training a classifier for entire big data is a daunting task. The solution would be to train the classifier to samples selected out of big data. The quantity of samples selected will always be a problem that yields only optimal solutions and so is the task of training a classifier to samples of large data, with expected empirical results [8]. Whether the generated samples would be sufficient enough to train is a problem addressed in past times. The quality of samples selected via approximation techniques has various statistical approaches [7]. To provide efficient strategies to train a classifier with possibly fewer numbers of samples yet keep its classifying correctness within probably approximate correct framework progressive sampling methodology was resulted. A progressive sampling methodology is an iterative approach where each iterative step is designed to increment the sample size for the next iterative step depending on whether the convergence of the classifier is met for the given problem. [5][6]. All previous work related to PSA computation time was not tested upon a dataset with millions of instances and high dimensional. The initial sample size for the first iteration of PSA plays a significant role reduction the computational time for progressive sampling and also in turn overall run time [5][6][8][1]. The total number of iterations of PSA will be greatly reduced if the starting sample size or the initial sample size is very much minimal. Previous works concerned with minimal initial samples have not experimented on Big data sets.

Although there are few works on Statistical estimation of minimal starting sample size, none was concerned with the computational time of progressive sampling methodology for Big Data in a distributed file system environment.

To the best of our knowledge, our approach presented in this work efficiently determines a

minimal initial sample size for big data in a Hadoop environment with a concern for computational time and overall runtime of progressive sampling methodology. In recent advancements to progressive sampling methodology, Rademacher Averages are tested to provide the utmost bound for the sample size of a given problem and a given dataset [3][4][7][8]. Though the utmost bound proved to provide the maximum bound for the sample required to train a classifier and keep its performance within the PAC framework, there is no such profound work concerned overall run time of the algorithm.

The overall runtime of such a progressive sampling algorithm will be proportional to the number of iterations the PSA executes which incorporates time expenses in upgrading the sample size at each iteration and the amount of time training the classifier until the stopping criterion is met [9][10]. The stopping criterion in the context refers to the convergence of the classifier ensuring it is within the PAC framework. The total time required for progressive sampling computation can be expressed as Big-Oh (Sample size + computation time for sample schedule + run time of training classifier and testing). Thus, incorporating progressive sampling techniques for big data based information technology application, enhances fast and efficient learning and scalability, thereby leading to a new dimension in training and classification of big data.

In Section one defines the motivation of PS in Big data, initial sample size sampling problems in Progressive Sampling in big data, problem formulation, and outcome of PS in Big data. Section two studies the background. Section three covers the modeling of PS in the Big Data Hadoop Environment. Section Four provides the proposed methodology. Section five provides validation of the dataset with experimental setup and compares results obtained with the existing state.

2. RELATED WORK

In this section, the essentials of sampling in big data and data mining with the need for progressive sampling, optimized Convergence and divergence of learning algorithms for optimal training datasets, apply of discussed issues in big data parallel computing environment-related issues are discussed as follows:

The sampling of large data has proved versatile in solving many data mining tasks in addition to its primary usage in training classifiers. For example, in the paper [10] proved that sample set cardinality derived using Chernoff bounds can reduce the time consumed to mine association rules. However, such size bounds were found to be too conservative in practice. In the paper [12] [8] provided a VC-dimension-based sample bound on Big Data sets for the randomized algorithm PARMA which extracts frequent item sets and associations. In the paper [12] provided Rademacher Averages-based sample bound on Big Data sets for the randomized algorithm PARMA which extracts frequent item sets and association rules.

Sampling data Methodologies on progressive sampling discussed by Bradley et al proposed and proved that existing clustering algorithms can be scaled up to large databases by a repeated updating of the current model with randomly drawn samples [13]. Sampling is also widely adopted in database retrieval methodologies. For instance, according to the self-tuning samples presented by Ganti et al., the probability that a tuple will be chosen is proportional to how frequently it must respond to inquiries. (Exactly) [14]. However, in both [12] and [13], It is not indicated how to select the appropriate sample size. Although Ganti et al. also offered a metric to quantify the difference between two data sets in terms of the models constructed by a certain data mining technique [11], which might address the issue of generating models using random samples, it did not address how to choose the appropriate sample size.

Sampling of data size were discussed by Baohua Gu and Bing Liu [8] proposed a similar work to that of Ganti et al [11] [13] [15] in providing a metric to put a number on the variation between two data sets. However, the work discussed in [8] is based on statistical information divergence of data sets mostly from the UCI repository, and Sample sizes derived were independent of any data mining algorithm unlike in [3]. Alfonso Estrada and Eduardo F. Morales [27] proposed a Novel Progressive sampling technique aiming to reduce the overall run time to estimate the samples required for the convergence of any learning algorithm. However, the selection of the initial sample size by the work discussed in [28] is dependent on both types of data sets in use and the algorithm was chosen to act upon that data. Both [28] and [8] have not addressed the problem of huge data sets such as big data in Distributed Computational environments. For the concept of the most number of samples required for the convergence of learning algorithms use of statistically derived bounds by Matteo Riondato outdates all previous works [12]. However [12] [8], [13], [15] does not focus on improvising contemporary strategies followed in progressive sampling design. The fact that the initial sample size

chosen crucially determines the overall run time of any progressive sampling algorithm.

Rademacher Approximation is statistical tool to derive bounds on datasets required for the convergence of learning problem [18] [19] [20] [21]. To determine the sample numbers required to obtain the necessary accuracy for adaptive learning, localized Rademacher complexities are made use of. Such probabilistic bounds have been proved on the number of active examples and several applications to binary classification problems are considered [18]. Function classes can be created as combinations of functions from basis classes, and they can be shown to have Rademacher and Gaussian complexities that are constrained by the complexity of the underlying basis classes. [19] [22]. Biomedical data sets were efficiently used to estimate total sampling size using progressive sampling without deal with its initial sample size [23].

Work proposed in [12], [7] demonstrate the use statistical bound such as VC-Dimension and Rademacher Averages to derive ε-Approximation of given datasets and thus contributes to generate training samples in a one-shot sampling scheme. However, this may lead under sampling or oversampling depending on the type of dataset and classifiers used in learning. Even though [12] achieve parallel processing and onetime database scanning, it has not considered overall sampling time and heterogeneity scenarios of datasets and learning algorithm. On the other hand, work proposed in [8] proposes methodologies to compute minimal initial sample size for simple data sets which can fit into primary memory. In the phase of improvisation of iterative nature of progressive sampling scheme for big data, work contribution from all previous works aims little towards optimization of computation.

In this work, an enhanced methodology named Statistically Optimal Initial Sample size (SOISsize) for deriving to derive the initial minimal sample set size is provided. This work can be applied to carry out Progressive sampling on Big Data with size beyond the capacity of main memory, in a distributed computational environment such as HDFS.Since all computations in the proposed work are optimized in a parallel computation using Hadoop, the overall performance comparatively out rates all concerned previous works.

3. PROGRESSIVE SAMPLING MODEL ON BIGDATA ENVIRONMENT

In this section architecture of computation and data handling in a big-data Hadoop environment is modeled and discussed. The multi-Vectored and highly dimensional big data set can be loaded to secondary memory and preprocessed to apply Map Reduce function to generate key-value pairs. In the following section

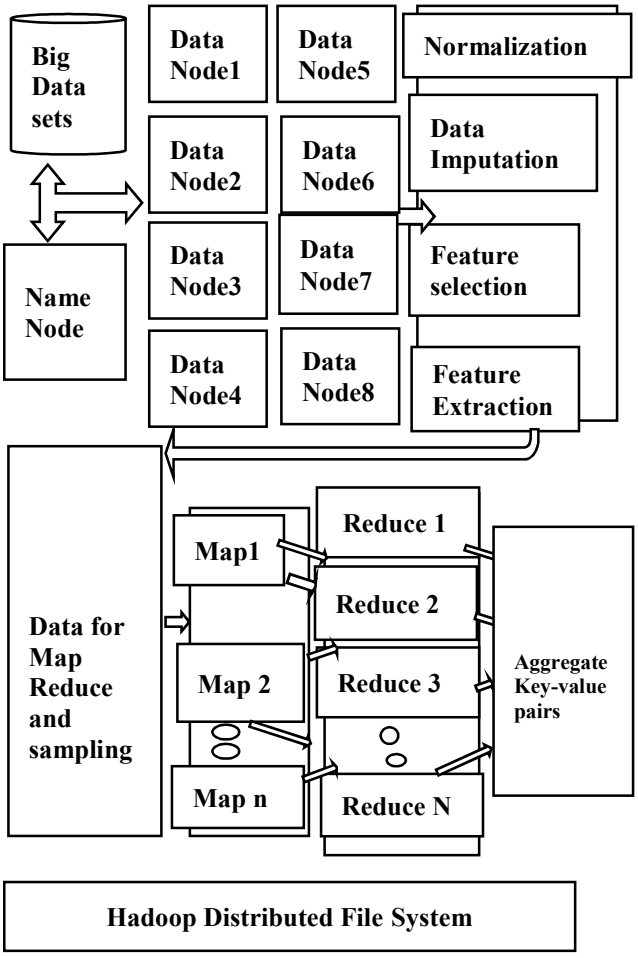

Figure 1. Architecture big data sets in the Hadoop environment

Figure 1 clearly depicts the architecture of big data in a distributed file system architecture. Data sets considered for the processing and analytics will be loaded into the HDFS system which names node. Entire data sets which require Giga bytes of space in memory will be loaded to n data nodes while all metadata required for controlling the data node will be kept created in the data node. HDFS analyses the data pattern and reports errors in the consistency of data. All reported error data will be resolved with basic preprocessing techniques [38]. Preprocessing techniques such as normalization, filling missing values, feature verification and selection, etc. are incorporated here. Post to preprocessing Map Reduce framework is applied to reduce the data to key-value pairs. Further preprocessed data will be subjected to parallel computation to estimate the

initial minimal sample size. Pre-Processing stages applied to Biomedical and UCI big data incorporate the following standard procedures. Data chosen for the estimation of the initial minimal sample size and sample set is an acquisition from various data sources.

3.1 Normalization

This preprocessing stage ensures that each row in the data set has a unit norm. Any row in the data set chosen, if found to mismatch in the expression of the unit for any quantity, such rows are normalized to maintain integrity.

3.2 Data Imputation

Data imputation is a technique where the missing value row will be retained by substituting it with a suitable value. The substation of value depends on the probable and classified reason for missing data. For the biomedical dataset chosen for estimating the initial sample size, the most occurring value is replaced.

3.3 Feature Selection and Extraction

The selection of a subset of features concerned with the setting of the experiment refers here to as feature selection. Most prominent features are just sufficient to make analytics hints feature selection, eliminating the most redundant ones. Extraction is a technique that can be applied to a subset of features to estimate a new feature.in the case of biomedical data set incorporated for progressive sampling, any additional features related to visibility or structure are introduced.

3.4 Hadoop Distributed File system

Hadoop Distributed File system implements a multi-node cluster mechanism to store data in a data nodes system called racks. Each data node has processing and storage capabilities. All data nodes are configured and controlled by a special node called the Name node. All data nodes under a single name node are enabled to run distributed applications including mining and analytics. In this context after preprocessing of big data sets, generated key-value pairs will be sued to estimate and determine a minimal initial sample set for a progressive sampling of big datasets using Rademacher Averages [39].

3.5 Map-Reduce Framework

Map–Reduce Algorithm generates key-value pairs out of the data provided to it as input. Mapping algorithms map data in different blocks and generate intermediate key-value pairs which will be provided as input to Reducer which reduces key-value pairs which will be then saved in HDFS. The Map Reduce algorithm is constructed based on two functions such as a map and reduce where the input is obtained from the key-value pairs which are denoted as (k,v).

The map function obtains a pair of inputs in a single iteration and provides a multiset of pairs i.e. $\{(k\_1,v\_1),(k\_2,v\_2),\ldots,(k\_n,v\_n)\}$. The multiset union is denoted as U which is comprised of the multisets of map function when it is applied to every pair of inputs. The set U is partitioned into $U\_\bar{k}$, where the specified key is denoted as $\bar{k}$ and the $U\_\bar{k}$ is comprised of the pair of values denoted as $(\bar{k},v)$. Similarly, the reduce function takes the input key as $\bar{k}$ and provides multiple sets of values [38][39].

The results of preprocessing of big datasets are tabulated as follows

Table 1. Analysis of preprocessing on datasets

	Parameters of Dataset a. Features selection, b. Features Extracted, c. No missing value rows and imputed, d. Size in GBs, e. Framework				
dataset	**a**	**b.**	**c.**	**d.**	**e**
ECBDL	985	3	35	23	HDFS
Medicare	123	3	56	25	HDFS
Melanoma	117513	0	23	103	HDFS
Splice	1000000	3	789	98	HDFS
adult	14	2	6	0.2	HDFS
census	14	2	5	0.2	HDFS
led	7	2	7	0.1	HDFS
covtype	54	2	7	0.5	HDFS

In the table 1 provides a complete analysis of each dataset loaded to Hadoop and the methodology applied on each data node to reduce the loaded data into key-value pairs

4. ESTIMATING A MINIMAL INITIAL SAMPLE SIZE MATHEMATICAL MODEL

In this section, complete methodology and associated concepts for estimating a minimal initial sample size to apply PSA in Big Data such that optimized learning of a concept can be achieved to be within the PAC framework, are proposed.

Figure 2 depicts the modeling of the proposed methodology to estimate the initial minimal sample size. Key-value pairs of dataset resulted after preprocessing and Map Reduce stages, the entire dataset will be taken to all data nodes in a distributed with the balance of instances loaded to each. Each part of the data set loaded specific data node is referred to as range space. For Data Set D, R is said

to range space where R⊂D. For each range space ri∈R as R= {r1. r2, r3…….}.

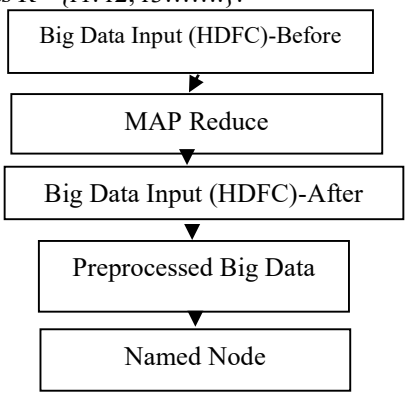

Figure 2. Modeling of Initial minimal sample size computation

4.1 Progressive sampling of Big Data

By fundamental definition of sampling, it refers to the selection of samples from the data pool. Here concerning big data, the selection of elements from the big data set is randomized. In all randomized selection probability of selecting any sample will always be 0.5 both with and without replacement strategies. A brief survey of the literature on sampling big data reveals a few sampling techniques such as Random sampling, Random Sampling with Bucketing, and Random sampling with Bernoulli's sampling [24] can be used in a Hadoop environment to handpick the samples from the data pool. A sampling of semi-structured data or unstructured data loaded to data nodes of HDFS can be sampled across all nodes and results can be aggregated to form a sample set. The sampling technique can be realized by a python package PyHive and Hive query language. Hive is a query language that can be used on any DFS or HDFS. PyHive is used to write scripts that run Hive queries on the data set loaded to HDFS.

Progressive sampling is controlled sampling where optimized training of any machine learning classifier is ensured to be within the probably approximately correct framework. It runs over iterations to arrive at the complete or total cardinality of samples optimally required to train the classifier. Use of Rademacher averages sets tight bounds for the most number of samples required. Computing Rademacher Averages locally at each data node and aggregating results in Rademacher Averages for the entire data set. The initial sample size or starting sample size required to start PSA on any given data set plays a vital role in determining the overall run time to approximate the optimal training sample set. In this work, a novel approach to estimating minimal initial sample size called Statistical Optimal size design is proposed with experimental design and results.

4.2 Measuring Quality of Samples

If a given dataset D, R is called to be Range space where R⊂D. Range space R_i∈R will be used to locally estimate sample size and later all of them are aggregated to form a Statistical Optimal Sample size. The primary criteria to estimate the sample size locally at each data node is to consider sample quality. The metric adopted here to measure the quality of a sample is Divergence or Deviation contributed by Kullback's Information Measure [12]. Although Divergence Concept is used in their work by Baohua Gu and Bing Liu [8] to estimate it did not experiment with Big data sets in a distributed environment.

i) Divergence

If z is a particular value of any generic variable Z and H_i is the concept that Z is from a statistical population with generalized probability densities (under a probability measure λ) $f_i(x)$, i = 1, 2, then the information divergence [8] is defined by

$$J(1, 2) = \int (f1(x)-f2(x)) (\log f1(x) - \log f2(x)\, d\lambda(x)).$$

According to [8] [23] J (1, 2) is referred to as a measure of the divergence between hypotheses H_1 and H_2 which indicates the measure of the difficulty of discrimination between them. Given two multinomial populations, if P_{ij} is the probability of occurrence of a j^{th} value in the ith population then the information divergence is

$$J(1, 2) = \Sigma c_j=1 (p1_j-p2_j) (\log p1_j - \log p2_j)$$

The information divergence has a limiting property which is described in theorem 1 whose proof can be obtained from [8]. Theorem 1 serves as a fundamental basis for the estimation of Sios.

Theorem 1: *If f(x) is probability density function and {fn(x)}, is a series of probability density functions where n → +∞, denote the information divergence from fn(x) to f(x) as J(fn(x),f(x)), then we have, if J(fn(x),f(x)) → 0, so that fn(x)/f(x) → 1[λ], uniformly. Here [λ] refers to the limitation and the fraction held in a probability measure λ. [8]*

Definition 1: *For any Range Space R (with n attributes) subset of a big data set i.e. R⊂D being processed in distributed environment such as HDFS and its sample S, then the quality of sample Q(S) =exp (-J (S, R)) where J_k(S, R) denotes information divergence between S and R on attribute K and it is given by J = (1 /n) $\sum n_k=1$ J_k(S, R) [8].*

To calculate J_k(S, R) at each data node of the Hadoop system all categorical attributes are treated as multinomial and continuous attributes such as numerical attributes are also treated as multinomial

by considering respective bin classes and bin class frequency through the construction of histograms. As per [8] the information divergence $J > 0$, hence $0 < Q \leq 1$, where $Q = 1$ means that no information divergence exists between S and R. Larger the information divergence, the smaller will be the sample quality; and vice versa [8] [24].

Upon a single scan of a specific Range Space R at each data node, both categorical values and frequencies of numerical values that fall in the bins can be incrementally gathered. Since scan and quality measure to estimate minimal sample size, runs parallel, the time complexity of calculating sample quality at any data node will be O (Nd) where Nd is the total number of instances at each data node. Since all data nodes in HDFS works the parallel same amount of time would require to process the entire data in HDFS and measure sample quality and hence to have an approximation of the quality of all samples in the entire data as a whole.

At each data node of Hadoop, occurrences of categorical values and Frequencies of numerical values can be incrementally collected. Hence Time complexity of calculating sample quality is O (N_d) (Nd is the total number of instances at any one data node of HDFS) [25][26]. Since computation runs massively parallel time taken by one data node to compute the sample quality will be the time for the entire big data set. However, data processing speed depends on the nature of the data under processing. Space complexity issues are trivial in impact as HDFS manages the entire data set on a secondary disk only. One scan of data at each node is sufficient enough to make processing and analytics.

ii) Statistically optimal Initial Sample size

The larger the sample size, the higher will be the sample quality. Therefore, the higher the divergence between elements of a sample, the better will be the sample for training any machine learning classifier. Statistically optimal initial sample need not be same as Optimal Sample size (OS_{size}). Hence statistically optimal initial sample size ($SOIS_{size}$) can be defined as follows

Definition 2*: For any dataset D, its $SOIS_{size}$ is the size at which its sample quality tends to become sufficiently close to 1.*

SOISsize depends only on dataset D, while OSS relies on both the data set and the learning algorithm. Therefore, the $SOIS_{size}$ is not necessarily the OSsize. Model Accuracies of both OSsize and $SOIS_{size}$ can be very close. For a given machine learning classifier LM and a sufficiently large dataset D with a probability density function FD(x), LM can be used as an operator to map FD(x) to any real number., i.e., LM: FD(x) → Acc∗ [8] where Acc* is the maximum accuracy obtained. Assume that a random sample of OS_{size} has the probability density function FOS(x) and that of SOISsize has p.d.f. Fsois(x) then there exists a theorem which is as follows;

Theorem 2: *If a random sample S of D has a probability density function FD(x) and L satisfies that FS(x)/ FD(x) → 1[λ] =⇒ |L(FS(x)) − L(FD(x)) | → 0, then Accsoissize → Accossize.*

Proof: Suppose we have a series of n random samples of Dataset D with an incrementally larger sample size. Denoting the i^{th} sample as {Si}, sample quality for Si can be designated as Q (Si) where i= 1,2,3,4 …. n.

According to the definition of SOISsize, Si →$SOIS_{size}$ as Q $\{S_i\}$ →1 i.e., the information divergence of S_i from D is J (Si, D) → 0.

Applying Theorem 1, we have,

$FS_i(x)/f(x) \rightarrow 1$, therefore, $|L(FS_i(x)) - L(f(x))| \rightarrow 0$.

In an asymptotic sense,

$L(fS_i(x)) \rightarrow$ Accsosssize

and

$L(f(x)) \rightarrow$ Accossize, that is, Accsosssize → Accossize.

However, in addition, $SOIS_{size}$, since all computations are massively parallel; incorporates the idea of estimating the mean population with a certain confidence level at each data node of HDFS, and later averaging both to arrive at the exact number of samples to start progressive sampling in the phase of training any classifier to sufficiently huge datasets. From statistics, the size of a sample to estimate the mean of a population with a certain confidence level is given by [27]:

$$N_{MIN} = \frac{N * \sigma 2}{(N-1) * D + \sigma.} \quad \text{------- (1)}$$

Where D is ((N∗C) 2/ 4) and C is the confidence level of this estimate. Estimating σ 2, is needed, which in the absence of any information it is approximated by σ ≈ N 4. Since N − 1 ≈ N for large N, and after some simple manipulation:

$$N_{MIN} = \frac{N}{4 \bullet N \bullet C2 + 1} \quad \text{----- (2)}$$

Possibly a good way is s to fix a confidence level (C) and apply it to all the databases. The Average of $SOIS_{SIZE}$ and N_{MIN} can yield the minimal initial sample size required to launch a progressive sampling on huge datasets managed by HDFS [27].

4.3 Algorithm for estimating minimal Initial sample size (SOISsize)

Input: a large Dataset (D) of Size N {

$S_{i|i}$=1, 2, 3……, n}

Output: Statistically Optimal Initial Sample Size ($SOIS_{size}$)

Step1: Load $\forall r_i \in R$ where R= {r1, r2……..rn} and R ⊂D to specific node of HDFS

Step2: Estimate the Probability Distribution Function of each $r_i \in R$ at all nodes And select some samples from each bin Compute N_{min} for each Range with a Predefined confidence level

Step 3: Compute Sample Quality for each sample Q (S_i)

Step 4: Plot the intermediate regression curve and Estimate its find mid-point called $SOIS_{size}$

Step 5: Find the Mean of individual data nodes $SOIS_{size}$

Step 6: Find the Mean of Nmin at each specific Node. And finally, find the average of $SOIS_{size}$ and N_{MIN} to yield the most Expected optimal SOISS

Step 7: Apply Random sampling at each node Select a sample set whose cardinality is $SOIS_{size}$

Step 8: Random sample the sample set resulted in Step 7 again to gather accurate SOIS

i) Sampling Schedule Design

Geometric scheduling with constant common factors can be heuristically adopted to improvise the sampling schedule to be used over subsequent iterations in PSA. Doubling the sample size at each new iteration can give rise oversample of the learning algorithm the at any time near its convergence. Therefore, the idea to incorporate here is not to use a multiplicative factor of 2 for the SOISsize. However, sample size can be increased with a fractional geometric ratio such as 1.5, which would result in a good approximation of the number of samples used in training. Thus in all experiments conducted, a fractional common ratio is adopted which has played a significant role in the overall computational time of PSA [27].

ii) Computing Rademacher Averages

Let Z be space and D be a fixed distribution D|Z. Let S = {$z_1 \ldots z_m$} be a set of examples drawn i.i.d. from D|Z. Furthermore, F will therefore be a class of functions f: Z → R.

Definition. The empirical Rademacher complexity of F is defined to be

$Rm(F) = E\sigma [\sup f \in F ((1/m) \sum m_i=1 \sigma_i f(z_i))]$

Where $\sigma_1 \ldots \sigma_m$ are independent random variables uniformly chosen from {−1, 1}. Such random variables as Rademacher variables. The upper bound to Rademacher Approximation to the given dataset provided in [5][7][8] is adopted here. The code given in [23] is used to compute Rademacher Averages of any given data set.

5. RESULTS AND DISCUSSION

In this section, experiments were conducted on datasets such as ECBDL, Medicare, Melanoma, Splice, adult, census, led, and covtype to estimate the initial minimal sample size and samples to initiate the PSA. Validation of chosen datasets is demonstrated to compute time complexity and space complexity

5.1 Experimental setup and computational procedure

The proposed sampling algorithm using the Rademacher average is implemented using python programming language to evaluate classification accuracy. The evaluation is carried out in a system with an Intel i5 2.7 GHz processor, 6GB Random Access Memory (RAM), and windows 10 operating system.

To compute $SOIS_{size}$ for a given big dataset D, R ⊂ D is loaded to a configured HDFS almost equally to each data node. Code written in Python runs in a distributed way across all nodes to make a single scan of each range space $r_i \in R$ at each data node and compute corresponding $Q(S_i)$ (i = 1,2,3, n). For the corresponding pair of (S_i, Q_i) a quality curve is plotted in intermediate stages and $SOIS_{size}$ can be estimated at any data node by finding the mid-point of the quality curve. $SOIS_{size}$ computed at all nodes can be aggregated by taking the mean value can be calculated to arrive at the most optimal $SOIS_{size.}$ Since computation is massively parallel processing and distributed through whole data, further N_{MIN} can be computed at all data nodes and later both $SOIS_{size}$ computed and mean of N_{MIN} calculated can be averaged to gather statistically optimal initial sample size to apply progressive sampling to ensure optimized learning of big data by machine learning algorithm [2].

To calculate ε-Approximation Rademacher Averages for the given dataset and hypothesis mapping function. For conducting progressive sampling to train any classifier to its convergence with and without SOIS samples Naïve Bayes classifier is used. The naive Bayes Classifier is trained explicitly until its convergence is met for the data set chosen and the corresponding time and number of iterations required are recorded. The accuracy of classification for each data is experimentally verified and results are plotted. Accuracy of classification with $SOIS_{size}$ is compared to that of training without initial minimal samples [26].

The Convergence of classifier with very little or

optimal learning and time consumed for estimating sample are mainly evaluated by conducting experiments. PAC (Probably Approximately correct frame work) provides a range of optimal value within which convergence of LA has to lie (0 to 1). Computational Time is another key value experimentally determined in distributed environment. Concerned to progressive sampling estimating its computational time in parallel environment is relatively novel by the methodology employed. Experiment setting focuses to multi-vectored big data sets in which we cover Volume, Veracity, and Vectored etc. Additionally, data selected for experiments diversified.

5.2 Input Data Anaysis

A total of 8 large data sets are used to estimate the initial minimal sample size and also the Rademacher bound of the dataset for the specific classification problem. Out of 8 datasets 4 datasets outlined in Table1 such as 'ECBDL',' Medicare', 'Melanoma', and 'Splice' are derived from publicly available data from several domains of healthcare and Biomedical Informatics and other 4 data sets such as adults, led census, COV type are selected from UCI repository. In the following subsequent sections, description of the data set such as source, formulation, and previous research work presented. All data sets outlined in table 2 are inherently class-imbalanced.

i) Big datasets of vectors Variety and volume

The follow charactertics of dataset are as follows,

ECBDL Evolutionary Computation for Data and Big Learning is the full form of ECDBL. A data set of large protein contact map prediction at ECDBL workshop. Aaron N Richter and Taghi M. Khosgoftar reformatted this dataset for their work in sample size estimation. This dataset comprises pair of amino acids in every instance and a binary class label indicating whether or not the pair is a protein structure. In this work we have utilized the dataset provided by ECBDL in our experiments, as it contains 7,998,231 instances and 985 features [26].

Table 2. Biomedical Datasets of high volume and veracity

	Big data datasets: a. **Proteomics, b. Insurance Claims, c. Clinical records, d. Genomics**			
	ECBDL	**Medicare**	**Melanoma**	**Splice**
Domain	**a.**	**b.**	**c.**	**d.**
Native instances	7998231	3692555	9531408	4627840
Positive Instances	171,933	1409	17246	14549
Class ratio	2.15%	0.03815%	0.181%	0.341%
Features	985	123	117513	1000000

Medicare dataset contains attributes of medical insurance and medical claim of all US citizens aged 65 and above. Center for Medicare and medicated services releases this data every year. This data is highly dimensional and voluminous. Previous work used this data to make categorical encoding and reformat for big data experiments [26].

Melanoma is a dermatological dataset. This dataset includes data from common dermatology patients visits between 2011 and 2016. [26]. each instance in the dataset has attributes which describe Melanoma characteristics of recent years. A binary class label is adopted to identify whether a patient with certain dermatological features could be classified as melanoma or not. Features incorporated in the model include patient name, family history, allergies etc.

Splice a dataset called Splice is used to locate human acceptor splice sites in DNA sequences. This complex, large data set was acquired from the LIBSVM dataset archive. A significant prediction issue is splice detection. Splice dataset is gathered from LIBSVM repository and used to perform sampling and derive samples.

ii) Big datasets of vectors Variety and volume

These datasets are adopted from UCI and exhibit veracity and velocity

Table 3. UCI datasets

	Datasets			
	Adult	**Led**	**census**	**covtype**
Instances	48000	100K	199.5k	581012
Positive Instances	36000	NA	48842	360000
Class ratio	51.23%	45.76%	2.84%	5.98%
feature	14	7	14	54

The adult dataset comprises more than 45k instances with 14 different features such as age, work, education, relationship. Etc.

The led dataset comprises more than 100k instances with 7 attributes about LED display domain with 10 class labels. etc.

In the Census, adult dataset comprises more than 190k instances with 14 different features such as age, work, education, relationship. Etc.

IN the Covtype, adult dataset comprises more than 58k instances with 54 different features such as soil type, hillslope, forest, etc.

We evaluate measures on 4 huge and voluminous biomedical big data sets such as ECBDL, Medicare, Melanoma, and Splice, and 4 UCI datasets such as adult, led, census, and covtype. Table 3 shows the statistically optimal initial sample size ($SOIS_{size}$) for all data sets used in our experiments, Computation time (T_{sois}) to gather SOISsize. Progressive sampling time with the incorporation of the minimal initial sample set, to converge naïve Bayes classifier is denoted by TPS_{sois} and that without SOIS is by TPS_{ws}. The number of Iteration with SOIS is

indicated by NoI_{sois} and that without SOIS is denoted by NoI_{ws}. Plots show the sample size, computation time, and progressive sampling time with and without the initial sample size. The accuracy of the classifier trained to values generated by PSA is evaluated against the so far best methodologies in the literature to estimate the initial sample size.

5.3 Validation of Results

Table 4 outlines the initial sample size $SOIS_{size}$ along with its computation time T_{sois}, Time for PSA TPS_{sois}, Time for PSA without Initial samples size T_{ws} and Number of iterations. The values shown in the columns of table 4 describe the results of the Progressive Sampling Algorithm with an Initial minimal sample size. The column with the Attribute

Table 4. Initial Sample size and time complexities

	Progressive sampling results		
Datasets	**Total Data**	**$SOIS_{size}$**	**$T_{sois(in\ Sec)}$**
ECBDL	7998231	1279716	3.8
Medicare	3692555	590809	2.8
Melanoma	9531408	1,525,025	5.34
Splice	4627840	740454	4.4
adult	48000	8000	2.2
led	100K	10000	3.1
census	199.5k	35000	1.23
covtype	581012	70000	1.56

'Total data' refers to the total number of instances the data set has. SOISsize refers to a number of instances that can be selected from the data set as initial minimal samples to apply a progressive sampling algorithm. Values estimated as optimal sample size depend on the dataset under consideration. Tsois refers to time expensed in the computation of SOISsize. Table 4 outlines the Time for PSA TPS_{sois}, Time for PSA without Initial samples size T_{ws} Number of iterations

The values shown in the columns of table 4 describe time complexities associated with the computation of PSA with and without $SOIS_{size}$. T_{sois} refers to the Time required for the estimation of $SOIS_{size}$.

Table 5: Initial Sample size and time complexities

	Progressive sampling time complexity results		
Datasets	**$T_{sois\ (in\ Sec)}$**	**$TPS_{sois\ (in\ Sec)}$**	**$TPS_{ws\ (in\ Sec)}$**
ECBDL	3.8	12.234	48.23
Medicare	2.8	13.432	50.43
Melanoma	5.34	13.234	55.12
Splice	4.4	15.456	56.23
adult	2.2	2.123	18.12
led	3.1	3.13	14.36
census	1.23	5.21	32.36
covtype	1.56	5.1	46.47

TPS_{sois} refers to time expensed in the execution of PSA including $SOIS_{size}$ sample in the first iteration. TPS_{ws} refers to the time required to execution of PSA without including SOISsize. Table 6 outlines a number of iterations for PSA with and without the initial sample size Table 6 shows the number of iterations required to complete PSA on datasets shown in the first column. NoI_{sois} refers to the number of iterations required to complete PSA with incorporating of initial minimal sample size and NoI_{ws} refers to the number of iterations required to complete PSA without incorporating initial minimal samples in the first iteration. Thus by the value shown, it can be inferred that the initial sample size improves the efficiency of PSA.

Table 6. Number of Iterations

	No. of Iterations in PSA	
Datasets	**NoI_{sois}**	**NoI_{ws}**
ECBDL	4	32
Medicare	3	35
Melanoma	4	65
Splice	3	34
adult	2	18
led	2	25
census	2	26
covtype	2	31

Table 7 outlines space complexities for PSA with and without the initial sample size Table 7 shows the value of memory space occupied by or made use of, at each respective data node of HDFS. S_{data_node} refers to space in computation for PS without SOISsize. The quantity S_{data} node sois refers to t space in computation for PS with SOISsize.

Table 7. Space Complexity in PSA

Datasets	**$S_{data\_node(PS)}$**	**$S_{data_node_sois}$sois**
ECBDL	18	8
Medicare	16	6
Melanoma	25	15
Splice	28	8
adult	2	0.5
led	3	1
census	3	1
covtype	3	1

Journal of Theoretical and Applied Information Technology
15th July 2023. Vol.101. No 13

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

5.4 Validation of data sizes and minimal initial sample set size

The graph below shows the plot of the comparison of Dataset sizes and the corresponding initial minimal sample size estimated.

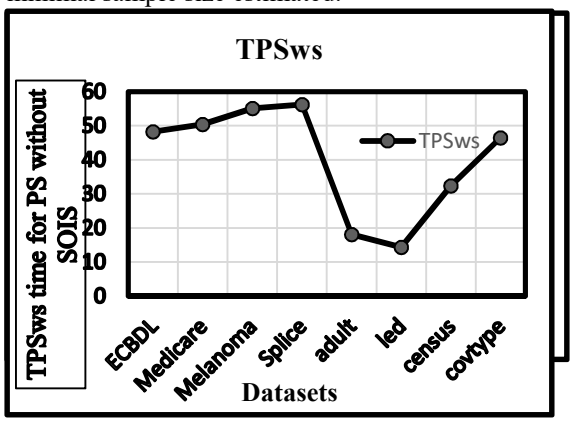

Figure 3: Results Of Dataset Size And Corresponding SOIS

Along the x-axis, dataset instances are plotted and with the y-axis, the initial sample size SOISsize is plotted. Unit along both axes is fixed to no of instances. It can be observed that there is a substantial decrease in the initial sample size required. Concerning biomedical big datasets, it can be observed that there is a substantial decrease in $SOIS_{size}$. and for datasets chosen from the UCI repository, there is a significant decrease in sample size.

Time Complexity of each data set for the initial sample size

It can be observed that variation of computation time almost varies linearly with respect to the volume of the dataset. The curve in the graph can be identified by Tsois.

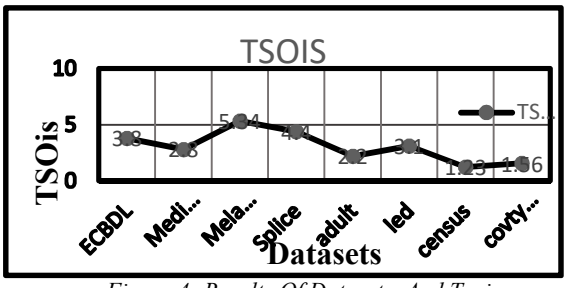

Figure 4: Results Of Datasets And Tsois

Time Complexity of each data set for progressive sampling without a minimal initial sample size

The above graph shows the plot of the Time taken to arrive at the initial minimal sample size. Along the x-axis, data sets are mentioned, and via the y-axis time computational time of the initial minimal sample size (TPSws)) are plotted.

Figure 5: Results of TPSws for each dataset

It can be observed that variation of computation time almost varies linearly w r t to the volume of the dataset. The curve in the graph can be identified by Tsois.

Time Complexity of each data set for progressive sampling

The graph below shows the time complexity of progressive sampling with the incorporation initial minimal sample size. Along the x-axis, data sets are marked and that of the y-axis time consumed for overall progressive sampling is plotted.

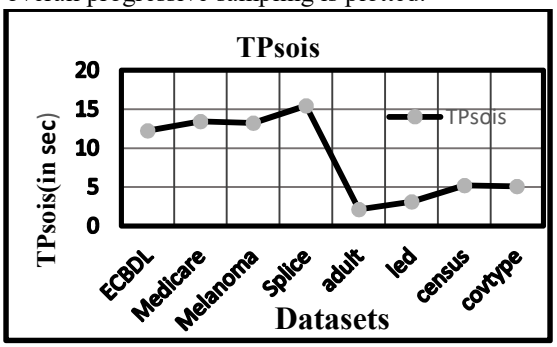

Figure 6: Results Of Tp_{sois} For Each Dataset

5.5 Comparison of Time Complexities for each data set for progressive sampling

The figure7 shows a plot of comparison of time complexities of PSA with and without SOISsize. Variation of time complexities with and without initial minimal sample sizes shows a linear relationship. Along the x-axis, TPSws are plotted. The unit of TPSws and TPSsois is considered to be in seconds. The curve which indicates progressive sampling on big datasets is identified by TPSsois in the graph. The curve which indicates progressive sampling without an initial minimal sample size is identified by TPS_{ws}.

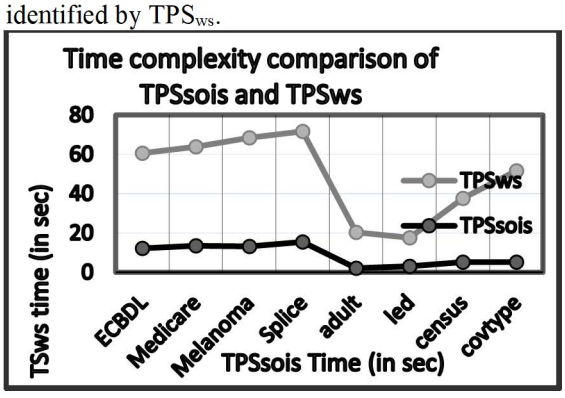

Figure 7: Results Of Tpssois And Tps_{ws}

It can be observed in the graph that there is a substantial reduction in the total time taken for the

conduction of progressive sampling in big data and arriving at an accurate classification within the PAC framework. Concerning the voluminous dataset from the biomedical category there is a significant difference in time required for progressive sampling due to the veracity of the datasets Thus the variation of time complexities of progressive sampling varies in (Big-Oh) O(Size of datasets+ training cost). However, with initial minimal sample size effectively reduces overall PSA runtime cost.

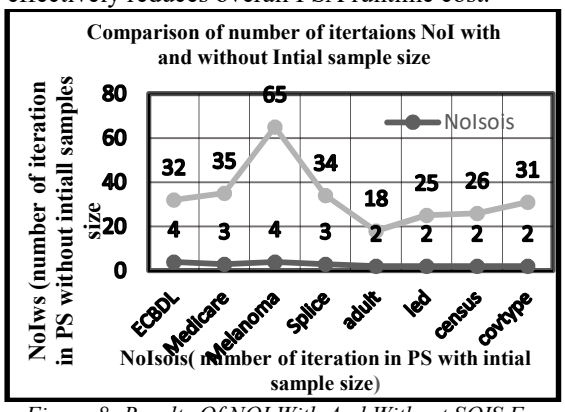

Figure 8: Results Of NOI With And Without SOIS For Each Dataset

The figure 8 shows a plot of comparison of the number of iterations of PSA with and without $SOIS_{size.}$ Along the x-axis, datasets are marked and that of the y-axis quantify the number of iteration incurred in a total runtime of PSA with the incorporation of Initial minimal sample size ($SOIS_{size}$) and without it. The curves shown in the graph indicates the number of iteration PSA runs with and without the incorporation of $SOIS_{size}$

Space Complexity of each data set for the initial sample size

The Figure 9 shows a plot of the Space complexities of initial sample sizes for the corresponding dataset. Space complexity refers to memory occupation by certain data under processing. In this sum, the total data in all data nodes of HDFS during the computation of PSA are plotted for each dataset. Along the X-axis datasets are marked and the y-axis memory space in Giga Bytes is quantified. The curve shown graph indicates the memory consumed for the process of estimating the initial sample.

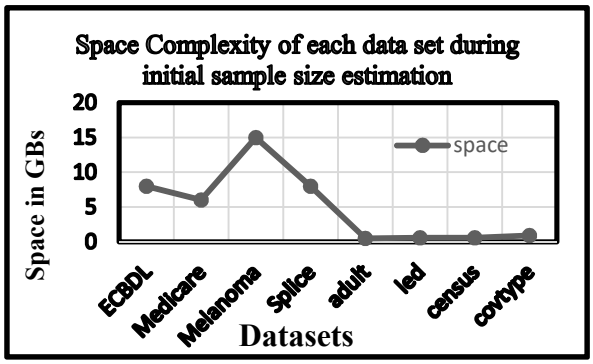

Figure 9: Results Of Dataset Size And Corresponding $Sois_{size}$

Space Complexity of each data set for the Progressive sampling process

The figure10 shows the plot of Space complexities of the PS process for each dataset along the x-axis, datasets are marked, and which sum of total data occupied at each node of HDFS is plotted. Size of the data occupied at each node in gigabytes. It is evident in the graph that incorporating an initial minimal sample size in applying PSA in Big Data Analytics would exhibit a substantial reduction in memory space usage. Additionally, employing parallel processing also assures possibly less computational space and time.

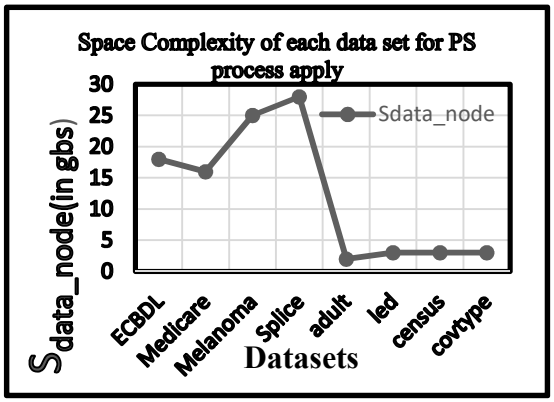

Figure 10: Results Of Dataset Size And During PS Corresponding $Sois_{size}$

The Figure 11 shows, a plot of space complexities in the computation process of PSA with and without $SOIS_{sois}$. Along the X-axis datasets are marked and through the y-axis, space complexities are calibrated with the unit in terms of gigabyte. The curves Sdata_node_sois in the graph indicate the usage of memory space while the computation of PSA is running state, with the incorporation of $SOIS_{size}$.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

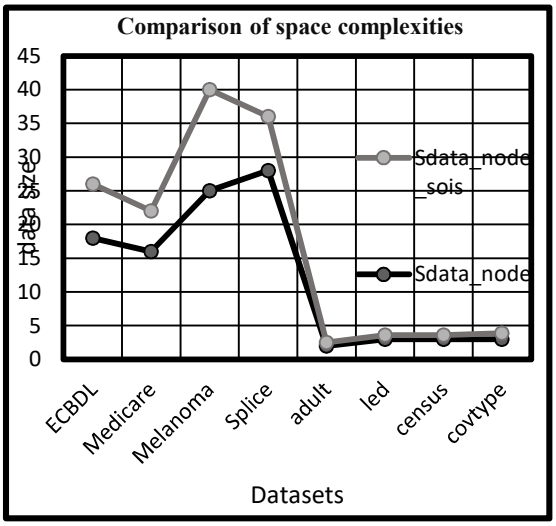

Figure 11: Results Of Space Complexities In PS Computation

The curve Sdata_node indicates memory occupation for computing PSA without SOIS. It is clearly visible that with SOISsize, lesser and more efficient occupation of memory for PSA computation.

Accuracy of classification.

The figure 12 shows a plot of the accuracy results of classification obtained by training a learning algorithm with SOISsize and without SOISsize Along the x-axis, datasets are marked and with the Y-axis Accuracies of classification by the Naïve Bayes algorithm are plotted. The curve "PSA with SOIS Accuracy of classification" indicates the accuracy demonstrated by the classifier for the training data set approximated by PSA with $SOIS_{size}$ obtained through the proposed work.

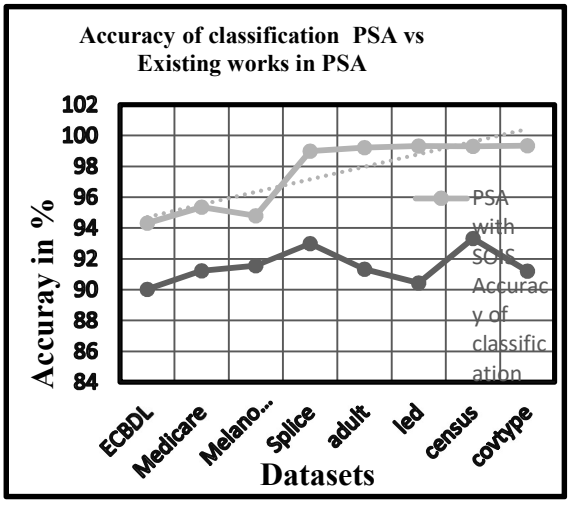

Figure 12: Results Of Accuracy Of Classification

The curve "Existing work Accuracy" demonstrate the accuracy obtained out of training the classifier by existing methodologies to derive an initial minimal sample size. It is evident from the graph that PSA with $SOIS_{size}$ would enable the classifier to provide better accuracy than earlier models of PSA with the initial sample size estimation technique. For the comparison of accuracy, earlier work proved in [4], [36], and [37] were considered.

6. COMPARATIVE STUDY

This section presents a comparative study of the different proposed models for estimating the initial minimal sample size for a progressive sampling of big data. To analyze the impact of the estimated initial minimal sample size on progressive sampling efficiency, the accuracy of classifiers for each iteration output is studied.

In the Bangera, Nandita, and Kayarvizhya [36] showed that progressive sampling provides effective sampling for future sensitive machine learning algorithm. Richter, Aaron N, and Taghi M Khoftar [26] provided a methodology to determine the sample size of biomedical big data with a complete focus on learning from generated samples and achieved good accuracy in learning. Our work presents reasonably better accuracy in learning when compared to learning accuracies in [26],[36],[37].Since there was no discussion on the overall execution time of PS, the proposed work provides comparatively good accuracy and optimal computational time and convergence of the learning algorithm.

In the Gu B, Liu B, and Hu F [8] worked on estimating the initial sample size for PSA on data sets of UCI and derived $SOIS_{size}$ using the statistical information measure and also obtained classifier accuracy in the range of 90-93 percent for the data set and classifier used. Also, the sampling schedule adopted in their work was a geometric ratio equal to 2 which would double the sample size for each subsequent iteration. In the Provost F, Janse D Oates [3] proposed a methodology of progressive sampling drawing the initial sample was made completely random. In the Matteo Riondato and Eli Upfal [14] have presented an estimation of ε-Approximation of frequent item set via progressive sampling approach with Rademacher averages with an assumed random initial sample set size.

In the Provest F, Jensen, and D. Oates T [3] proposed a method for progressive sampling where the initial sample size was assumed to random value and

geometric progression for sampling schedule design. Bangera, Nandita, and Kayarvizhya [36] achieved accuracy in an average of 91% for feature-based and progressive sampling. However initial sample selection was made random and not guaranteed to be minimum. Bangera, Nandita, and Kayarvizhya [37] proposed a reduction in the time taken to mine frequent item sets using progressive sampling with Rademacher averages. The Runtime of PSA is compared with that of static sampling without concern for the initial minimal sample size. [37] achieved significantly good accuracy.

Richter, Aaron N, and Taghi M Khoftar [26] provided a methodology to determine the sample size of biomedical big data with a complete focus on learning from generated samples and achieved good accuracy in learning. Since there was no discussion on the overall execution time of PS, the proposed work provides comparatively good accuracy and optimal computational time and convergence of the learning algorithm. Convergence of classifier is ensured to be within PAC framework accuracy limits.

Table 7: Comparative Studies

Metrics of Performance a. **Type of Big data, b. Methodology, c. $T_{sois,\ d.}$ Avg $T_{psa,\ e.}$ Accuracy of classification**					
P-ID	a.	b.	c.	d.	e
[8]	Veracity and velocity	Statistical divergence	27.5		91.3%
[10]	Volume and veracity	ε-approx.	-	-	95.4%
[3] [4]	Volume and veracity	PSA	-	-	89.2%
[14]	Volume and veracity	Rademacher Averages	-	-	91.8%
[36]	Volume and veracity				91.8%
[37]	Volume and veracity	Rademacher Averages	-	12.3	91.8%
[26]	Volume and veracity	Inverse power law	-	-	91.8%
Proposed work	Velocity, veracity, and volume	PSA with Initial sample size	3.5	8.5	95.3%

7. CONCLUSIONS

Progressive sampling ensures the estimation of an optimum number of the sample set for training a learning algorithm iteratively. The initial sample set size in the very first iteration of PSA plays a key role in computational time and the number of samples used in the training phase. The proposed work efficiently estimates a minimal initial sample set size in a big-data Hadoop environment. Statistical measures used in the approach discussed are verified in a parallel computation environment and prove to be comparatively improved in all such related work. The proposed methodology statistically optimal sample size ($SOIS_{size)}$ experimentally shows a better result in computational time and cardinality of the training set. Space Complexities are transcendentally reduced due to distributed and parallel computing in the Hadoop environment.

Sampling on big data to extract a learning or training set in a distributed environment for highly dimensional datasets is in focus since very recent times. However computational time estimation and space complexity optimization is very novel to this area of the research work which serves out to be the strength of the paper. Results proposed were verified using one learning algorithm on multiple datasets which need to address in sequential papers. Also datasets used have values in binary, a different mathematical tool like pseudo dimension has to be used in future enhancements. Thus, this works significantly contributes to reducing training time and cost of learning algorithm while also keeping the result of a classifier to be in the probably approximately correct framework as the misclassification results.

In our opinion this work proposed, proves reasonably well on heterogeneous big data sets and shows best accuracy of learning. However, for real valued datasets Psuedo-Dimesion mathematical bounds are expected to provide near accurate sampling bounds. Hence adopting Pseudo Dimension to progressive sampling would be a future research works. For real valued datasets Pseudo dimension mathematical tool must be applied on datasets to determine bounds in distributed environment. Multiple learning algorithm performance for the bounds estimated can be used analysis as an extended future work.

REFERENCES:

[1]. Eite, Rui, and Pavel Brazdil. "Improving progressive sampling via meta-learning." EPIA. 2003.

[2]. Zhao, Jia, et al. "A novel clustering-based sampling approach for a minimum sample set in the big data environment." International Journal of Pattern Recognition and Artificial Intelligence 32.02 (2018): 1850003.

[3]. Pixar, Per Christensen. “Progressive Sampling Strategies for Disk Light Sources.” (2018).

[4]. Provost F., Jensen D., Oates T. (2001) Progressive Sampling. In: The Springer International Series in Engineering and Computer Science, vol 608. Springer, Boston, MA

[5]. Yathish Aradhya B C. and Y P Gowramma. Progressive Sampling Algorithm with Rademacher Averages for Optimized Learning of Big Data: A Novel Approach. International Journal of Computer Applications 175(15):37-40, August 2020.

[6]. Yathish Aradhya B C. and Y P Gowramma. Rademacher Averages bounded progressive sampling algorithm for big data analytics: A novel Approach International Journal of Scientific & Engineering Research Volume 11, Issue 7, July-2020 1374 ISSN 2229-5518.

[7]. Elomaa, Tapio, and Matti Kääriäinen. "Progressive Rademacher sampling." AAAI/IAAI. 2002.

[8]. Gu B., Liu B., Hu F., Liu H. (2001) Efficiently Determining the Starting Sample Size for Progressive Sampling. In: De Raedt L., Flach P. (eds) Machine Learning: ECML 2001. ECML 2001. Lecture Notes in Computer Science, vol 2167. Springer, Berlin, Heidelberg.

[9]. Static and Dynamic sampling by GH John - 1996 Aug 2, 1996 - Publication: KDD'96: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining August 1996.

[10]. Zaki, Mohammed Javeed, et al. "Evaluation of sampling for data mining of association rules." Proceedings Seventh International Workshop on Research Issues in Data Engineering. High Performance Database Management for Large-Scale Applications. IEEE, 1997.

[11]. "Sampling-based Randomized Algorithms for Big Data Analytics" by Matteo Riondato, Ph.D., Brown University, May 2014

[12]. V. Ganti, J. Gehrke, R. Ramakrishnan, and W.Y. Loh. A framework for measuring changes in data characteristics. In Proceedings of PODS'99, 1999.

[13]. An Overview of Statistical Learning Theory. Vladimir N. Vapnik. Abstract—Statistical learning theory was introduced in the late. 1960's. Until the 1990's it was a by VN Vapnik - - 1999Cited by 5582 articles.

[14]. Mining frequent item through Progressive Sampling with Rademacher Averages Publication: KDD '15: Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data MiningAugust 2015 Pages 1005–1014https://doi.org/10.1145/2783258.2783265

[15].Zeng, Xueqiang, and Gang Luo. "Progressive sampling-based Bayesian optimization for efficient and automatic machine learning model selection." Health information science and systems 5 (2017): 1-21.

[16]. Lozano, Fernando. "Model selection using Rademacher penalization." Proceedings of the 2nd ICSC Symp. on Neural Computation (NC2000). Berlin, Germany. ICSC Academic Press. 2000.

[17]. Koltchinskii, Vladimir. "Rademacher complexities and bounding the excess risk in active learning." The Journal of Machine Learning Research 11 (2010): 2457-2485.

[18]. Koltchinskii, Vladimir, and Dmitriy Panchenko. "Rademacher processes and bounding the risk of function learning." High dimensional probability II. Birkhäuser Boston, 2000.

[19]. Pellegrini, Leonardo, et al. "MCRapper: Monte-Carlo Rademacher averages for poset families and approximate pattern mining." ACM Transactions on Knowledge Discovery from Data (TKDD) 16.6 (2022): 1-29.

[20]. Ma, Shuhua, et al. "SC-PROSAC: An Improved Progressive Sample Consensus Algorithm Based on Spectral Clustering." 2021 3rd International Conference on Robotics and Computer Vision (ICRCV) (2021): 73-77.

[21]. Bartlett, Peter L., and Shahar Mendelson. "Rademacher and Gaussian Complexities: Risk Bounds and Structural Results." Journal of machine learning research (2003).

[22]. Oneto, L. et al. "An improved analysis of the Rademacher data-dependent bound using its self-bounding property." Neural networks: the official journal of the International Neural Network Society 44 (2013): 107-11.

[23].weightedrademacher/test_independent_c_vecs. py at master • ermongroup/weighted-rademacher • GitHub

[24]. hao, Jing, et al. "Efficiency-enhanced Progressive Sampling Method on One-shot Person Re-Identification." 2020 IEEE International Conference on Real-time Computing and Robotics (RCAR). IEEE, 2020.

[25]. Li, Hui, et al. "Progressive sample mining and representation learning for one-shot person re-identification." Pattern Recognition 110 (2021): 107614.

[26]. Richter, Aaron N. and Taghi M. Khoshgoftaar. "Sample size determination for biomedical big data with limited labels." Network Modeling Analysis in Health Informatics and Bioinformatics 9 (2020): 1-13.

[27]. Estrada, Alfonso and Eduardo F. Morales. "NSC: A New Progressive Sampling Algorithm."

[28]. Oneto, Luca, et al. "Fast convergence of extended Rademacher complexity bounds." 2015 International Joint Conference on Neural Networks (IJCNN). IEEE, 2015.

[29]. Ng, Willie, and Manoranjan Dash. "An evaluation of progressive sampling for imbalanced data sets." Sixth IEEE International Conference on Data Mining-Workshops (ICDMW'06). IEEE, 2006.

[30]. Umarani, Venkatapathy, and Muthusamy Punithavalli. "Analysis of the progressive sampling-based approach using real-life datasets." Open Computer Science 1.2 (2011): 221-242.

[31]. Lokesh M R, Swathi M, Sahana K S, Chandan Babu, Sushmitha K S" Ambient Air Quality Monitoring in Dumping Ground Using Cyber-Physical System "Journal of Emerging Technology and Innovative research @2022 JETIR July 2022, Volume 9, Issue 7 ISSN 2349-5162 https://www.jetir.org/view?paper=JETIR2207663

[32]. Lokesh M R, B K Amruth Gowda, Prajwal B L, Vaibhav M P, Haseeb Ahmed Danish" Review on Iris Recognition Research Directions- A Brief Study" International Journal for Research in Applied Science & Engineering Technology (IJRASET) ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538 Volume 10 Issue VII July 2022- Available at www.ijraset.com https://doi.org/10.22214/ijraset.2022.45856

[33]. Lokesh M R, Jahnavi R, Puneeth B U, Sthirya R, "A Device to Detect Dairy Product Freshness" International Journal of Scientific & Engineering Research, Volume 8, Issue 1, July-2022 ISSN 2229-5518 https://drive.google.com/file/d/1_g4QODtwAxIRZnEmhsU7Y6hrBAb7gO7T/view?usp=drivesdk

[34]. Lokesh, M. R., and Y. S. Kumaraswamy. "Next State Prediction algorithm for the avionic systems using the hidden Markov models" Green Computing and Internet of Things (ICGCIoT), 2015 International Conference on. IEEE, 2015.

[35]. Lokesh M R, Lavanya S Nair, Manish Kumar Nagar, Jyothi Lakshmi, Dilsha S Unni "Challenges in Accident Collision and Prevention in Cyber-Road Traffic Control System" International Journal for Scientific Research and Development IJSRD, Volume 3, Issue 12, Pages 384-386, 2016.

[36]. Bangera, Nandita and Kayarvizhya N. "Machine Learning Driven Feature Sensitive Progressive Sampling Model for Big Data Analytics." International Journal of Advanced Computer Science and Applications (2021):

[37]. Bangera, Nadita and Kayarvizhya N. "A Progressive Sampling-based Approach to Reduce Sampling Time." 2019 4th International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT) (2019): 74-78.

[38]. Mohammad, Atif, Hamid Mcheick, and Emanuel Grant. "Big data architecture evolution: 2014 and beyond." Proceedings of the fourth ACM international symposium on Development and analysis of intelligent vehicular networks and applications. 2014.

[39]. Pandagale, Ashwini A.. "A review on HDFS Architecture in Hadoop and Hadoop Components." (2017).