ENHANCING WEB MINING THROUGH ENSEMBLE TREE-BASED LEARNING FOR WEB USAGE PATTERN ANALYSIS

[1]Divya, [2]Dr. Shrwan Ram

[1]M.E. Scholar, Department of Computer Science and Engineering,M.B.M. University, Jodhpur

[2]Professor & HOD, Department of Computer Science and Engineering,M.B.M. University, Jodhpur

Email-: Divyavarde1901@gmail.com

* Corresponding Author: Divya

Abstract: *This research paper delves into the realm of web mining, specifically focusing on web usage pattern analysis using ensemble tree-based machine learning techniques. The study explores the challenges and advantages of web log pattern analysis, shedding light on its applications in personalized marketing, user behavior categorization, and business analytics. Privacy concerns associated with web usage mining are also addressed. The paper proposes a methodology employing tree-based classifiers, such as Decision Trees, Random Forests, and Gradient Boosting, to analyze web logs and extract meaningful insights. The research objectives include investigating machine learning techniques for web usage pattern analysis and evaluating the efficacy of an ensemble tree-based approach. The proposed methodology is applied to a dataset derived from Apache access logs, showcasing high accuracy, precision, recall, and F1-scores. Comparative analysis with other data mining techniques highlights the superiority of the proposed ensemble tree-based learning model in terms of overall performance.*

Keywords: *Web Mining, Web Usage Mining, Ensemble Tree-Based Learning, Decision Trees, Random Forests, Gradient Boosting, Apache Access Logs, Pattern Analysis, Machine Learning, Data Mining.*

I. INTRODUCTION

Web mining leverages the methods of data mining to derive meaningful insights from online data. This encompasses web pages, the connections among them, and the logs of user interactions on websites. By analyzing extensive online datasets, web mining reveals significant trends and patterns. This extracted data can be divided into details, documents, architectures, and user profiles. The foundation of web mining is built on both process-based and data-driven elements. It encompasses multiple stages, such as data collection, selection prior to processing, pattern discovery, and subsequent analysis. Given the pervasive role of the internet in today's society, the methodologies for extracting information from the web have become significant areas of research.

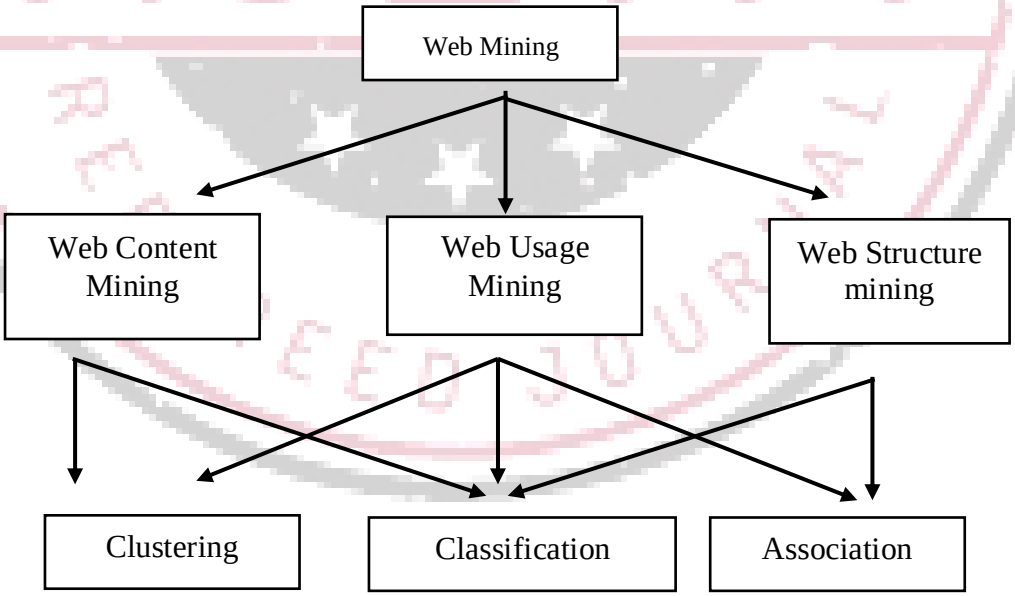

Figure 1. Taxonomy of Web Data Mining

A. Web Usage Mining

This domain emphasizes techniques that aim to forecast user behaviors when they engage with the internet. Web usage mining involves discerning user navigation trends from online data and aims to draw meaningful insights from the auxiliary data obtained during users' online browsing sessions. This kind of mining extracts data from web logs to recognize patterns in how users access particular sites. Current academic research and commercial tools analyze these patterns for a variety

of purposes. Insights derived from this analysis can be utilized for personalization, system upgrades, modifications to websites, business analytics, and categorizing user behavior. The act of monitoring user activities on websites and collecting this information through web logs is termed web usage mining, also commonly referred to as log mining [1]-[4].

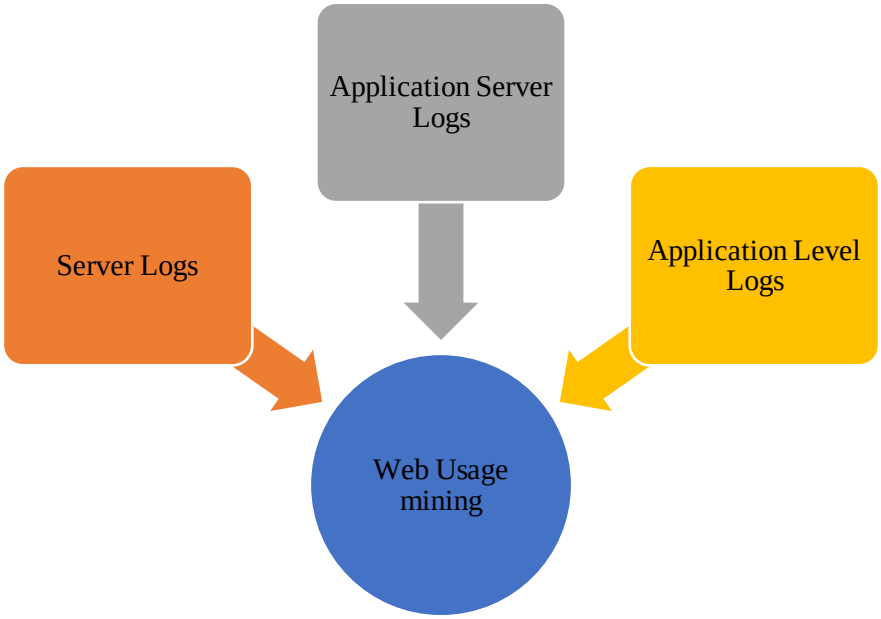

Figure 2 Web usage Mining data source

However, a significant challenge with Web Usage Mining is the nature of the data it relies upon. As the internet has expanded, there's been an exponential increase in web data, with many transactions occurring in mere seconds. This vast amount of data is often semi-structured, making it distinct from traditional structured data [5]. Therefore, it demands extensive preprocessing to extract meaningful information from this semi-structured content.

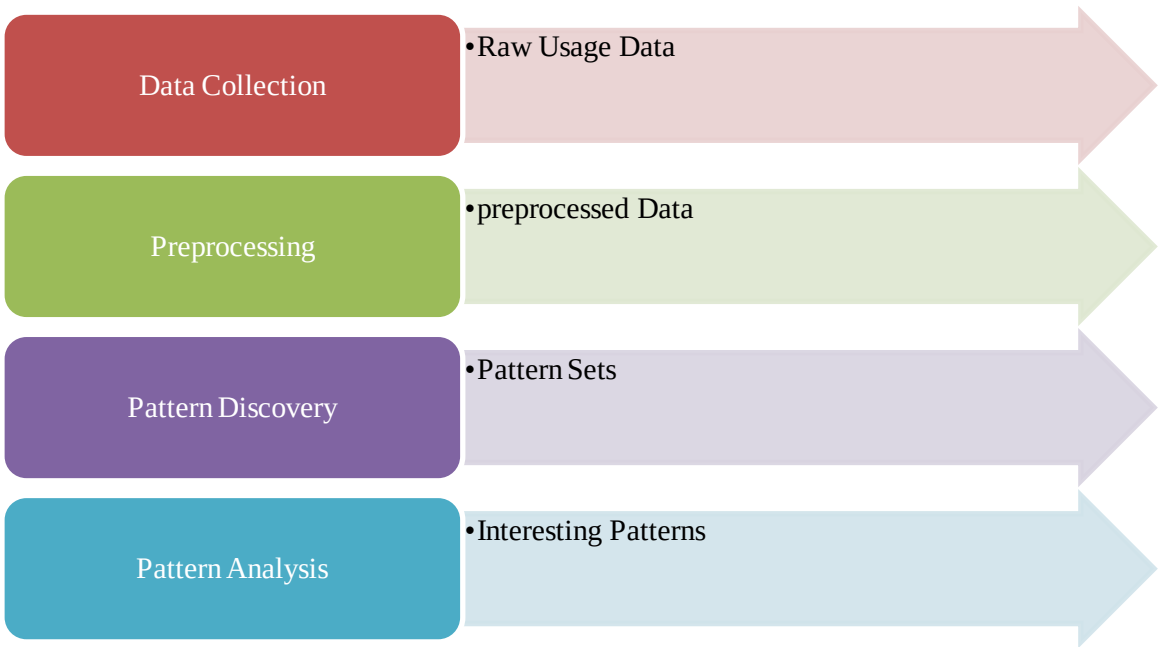

Figure 3 Web Usage Mining Phases

A. Advantages And Disadvantages Of Web Log Pattern Analysis

Advantages of Web Log Pattern Analysis

- Because of its various benefits, web usage mining is a popular technique among businesses and governmental organizations. Finding patterns in web usage data may be used to predict user attitudes in areas like industrialization and other medical industries.
- Personalized marketing via e-commerce is now partially due to this technology, increasing trade volumes.

- By better understanding client demands and responding to those needs more quickly, businesses may build stronger connections with their customers. By adjusting prices depending on the profiles they have generated, they may boost profitability. They can even locate clients who may otherwise choose a competitor. To reduce the probability of losing a client or consumers, the business will make special offers to the particular customer in question.
- Specific models, such as the probabilistic latent semantic analysis, offer additional dimensions to user behavior and access trends. One of the significant benefits of web usage mining, including personalization, is exemplified by this. Through this method's collaborative recommendations, users can access content that is more pertinent to them.
- The advantages of the method are also shown by factors specific to web usage mining. The use of semantic information in understanding, analyzing, and deducing usage patterns during the mining phase is one of them.

Disadvantages of Web Log Pattern Analysis

- While web usage mining is not inherently problematic, leveraging it on personal data can raise potential concerns.
- The invasion of privacy is the ethical concern with web usage mining that receives the greatest attention. When information about a person is acquired, utilized, or shared without the person's knowledge or permission, privacy is regarded to have been lost. To create anonymous profiles, the collected data will be examined, rendered anonymous, and then grouped.
- Rather than assessing users based on personally identifiable information, these programs de-individualize individuals by analyzing their mouse actions. De-individualization is generally understood as the propensity to evaluate and treat individuals based on traits shared by the group rather than individual traits and qualities.
- Companies that gather data for a specific purpose might repurpose it for completely different goals, potentially conflicting with the user's interests.

B. Challenges In In Web Mining

Web content mining, while valuable, faces certain challenges and issues. However, for each of these challenges, there are potential solutions. Some of these include.

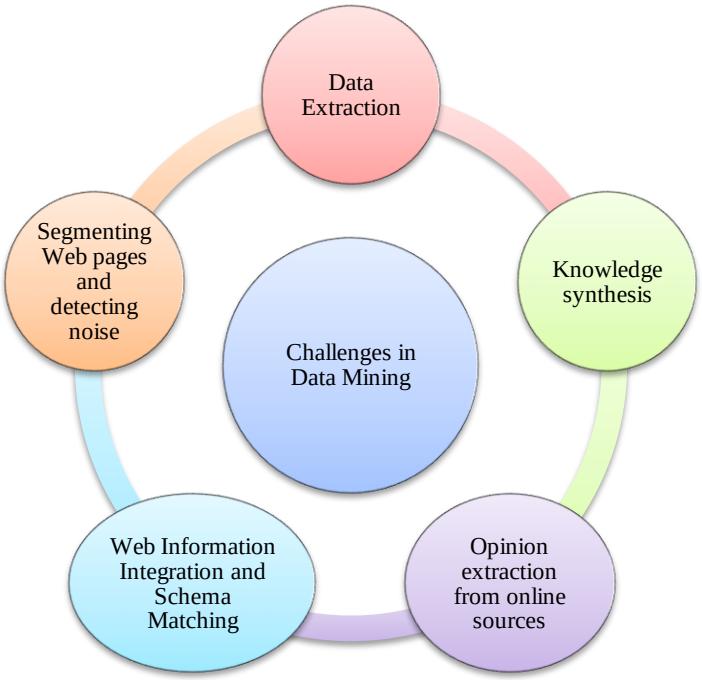

Figure 4 Challenges in Pattern and prediction analysis in web usage data

- Data Extraction: Extracting structured data, such as products and search results, out of web pages. Such data may be extracted to enable service provision. To overcome this issue, machine learning and automated extraction are the two primary kinds of methodologies employed.
- Web Information Integration and Schema Matching: The Web is filled with data, yet each website (or even page) displays the same data differently. Finding or matching data that is semantically similar is a significant issue with several practical applications.

- Extracting opinions from internet sources: There are various places to find opinions online, including forums, blogs, chat rooms, and consumer reviews of goods. For marketing intelligence and product benchmarking, mining opinions is crucial.
- Synthesis of knowledge: Concept hierarchies or ontologies are helpful in a variety of applications. However, manually producing them takes a lot of time. To provide the user with a comprehensive understanding of the issue area, the key application is to organize and synthesize the bits of information found on the web. We'll outline a few current techniques for investigating information redundancy on the web.
- Segmenting Web sites and identifying noise: In many Web applications, one just needs the primary content of the Web page without adverts, navigation links, and copyright notices. It's an intriguing challenge to automatically partition Web pages and extract the key material.

C. Research Objectives

The main objective of this research can be broken down into the following points:

- Investigate tree-based machine learning techniques for analyzing web usage patterns derived from weblogs.
- Evaluate the efficacy of an ensemble tree-based learning approach optimized for features in enhancing the performance of weblog behavior analysis.
- To improve the performace level.

II. LITERATURE REVIEW

Mittal et al. [6] proposed a machine learning based approach for web usage mining and also emphasizes its role in detecting patterns and extracting information from web server logs. As a component of web mining, web usage mining zeroes in on web data extraction. The researchers utilized tree-based classification techniques, including treeJ48, RandomTree, RandomForest, and REP Tree, to categorize server log data and subsequently scrutinized the outcomes.

Valeja et al. [7] employed the K-means method to enhance the speed of DBSCAN. They subsequently clustered each group using DBSCAN. When necessary, border clusters were merged. According to their findings, the proposed method reduced execution time by roughly 98% without substantially affecting the quality of clustering results.

Gholizadeh et al. [8] examined the relationship between production rules in manufacturing lines using association rules. They applied logical reasoning to address the interplay of rules between automotive production lines and products. The findings revealed that the application of data mining association rules achieved an accuracy rate of over 87%. This approach is valuable for formulating manufacturing guidelines, enhancing management decisions in the realm of IoT, and streamlining the production process.

Asadianfam et al. [9] proposed webpage recommendations using c-means clustering. A random selection of users was isolated, and clustering was performed based on the collected data. The study's results, using this model, yielded a list of top-recommended pages for users, achieving the highest accuracy of 89.56% and the utmost precision of 96.45% in comparison to other methods.

Mohanty et al. [10] employed the conventional fuzzy clustering method called fuzzy c-means to cluster fuzzy membership matrices and presented their clustering outcomes. A novel equation for formulating fuzzy membership vectors was used in conjunction with a feature weight calculation technique during the clustering process. The results indicate a 10% enhancement in clustering accuracy for multivariate time series datasets compared to prior clustering techniques.

Hailin and Miao [11] emphasized the development of continuous mining algorithms, especially for density-based clustering. Such methods demonstrate resilience in eliminating outliers from varying density regions or identifying clusters of diverse granularities. Their accuracy rate was 60.375% on selected datasets.

Bhattacharjee et al. [12] underscored research in artificial intelligence, illustrating how traditional clustering methods, when paired with innovative techniques, can achieve enhanced efficiency. Their focus was on clustering techniques such as CLARA, CURE, and K-means. Their method's efficacy was significantly higher when compared to its raw data output. They also incorporated a Rand index-based comparison with results from other methodologies using the same dataset. Notably, the CLARA method achieved 100% efficiency on two datasets and surpassed the DBSCAN method on one of them.

Kotyrba et al. [13] proposed clustering methods based on CLARANS. Their approach's novelty is its consideration of only users exhibiting consistent behavior during clustering. Results demonstrate notable improvements in clustering quality and speed. Metrics include a silhouette coefficient of 0.7830 and significant reductions in various computational requirements.

Bhuvaneswari et al. [14] pinpointed trends in inspection reports using rule-based text mining. They classified proposed renewal events based on gathered data and other pivotal parameters, such as age, into four levels of criticality using unsupervised clustering. According to their framework, merely 8.8% of damaged roofs are of utmost significance and are

within the budgetary constraints. This adaptable method can be applied to various assets and can significantly enhance fund allocation processes for large owner organizations like municipalities and school boards.

III. TREE BASED LEARNING CLASSIFIERS

Tree-based classifiers are a class of machine learning algorithms that leverage decision tree models to predict or classify data. Decision trees are hierarchical structures where each node represents a specific feature or attribute, and each branch signifies a decision based on that feature. These classifiers are particularly valuable in solving classification problems where the objective is to assign input data points to predefined classes or categories. Tree-based machine learning classifiers are widely utilized across various domains, including data mining, pattern recognition, and predictive modeling.

Tree-based classifiers employ a methodical approach to data organization and analysis, breaking it down into subsets based on attribute values. These classifiers construct trees iteratively, continually dividing data until a set criterion is achieved. Building such a classifier entails selecting optimal attributes at each node using criteria like maximizing information gain. The process refines the tree, enhancing its prediction accuracy.

A. Tree-Based Machine Learning Classification Methods

Tree-based machine learning algorithms are popular and powerful techniques for classification (and regression) tasks. These methods segment the predictor space into a number of simple regions. For a given observation falling into one of these regions, the prediction is a constant value. Here are some of the most widely used tree-based classification methods are presented.

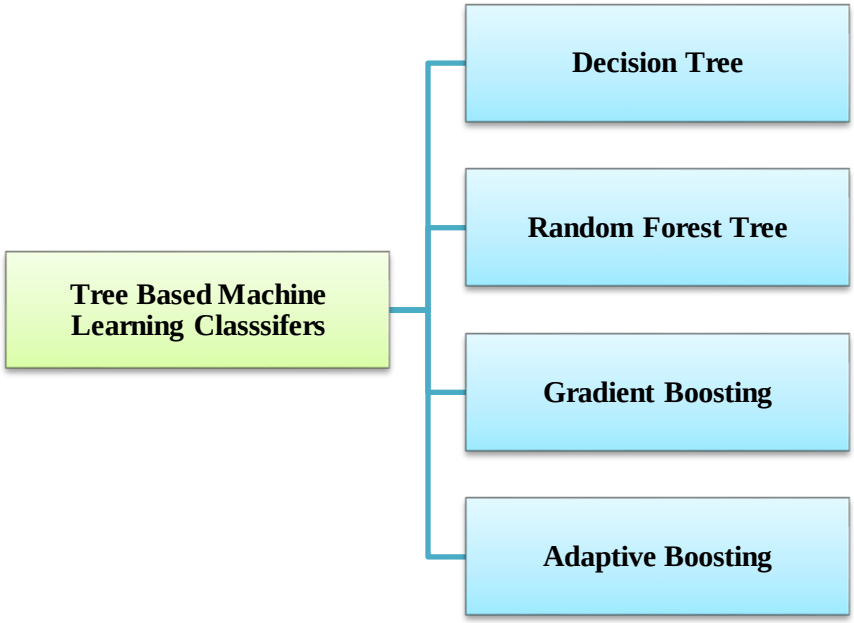

Figure 5 Types of Tree-Based Classifiers

Decision Tree

Decision trees operate by breaking down a complex decision-making process into a series of simpler decisions, effectively creating a "tree" of choices. In the context of supervised learning, decision trees perform this breakdown by recursively partitioning the data space into smaller regions based on feature conditions, as presented in figure 3.2. Beginning at the root node, the algorithm evaluates conditions to decide which subsequent branch or node to follow. This process continues until a leaf node is reached, which contains the final decision or classification. One drawback of conventional decision trees is their tendency to overfit the data, especially as the tree becomes more complex. Overfitting occurs when the model captures noise in the data, making it less effective for new, unseen data. Another limitation is that the partitioning is typically aligned with the coordinate axes, which might not always capture the underlying patterns in the data effectively.

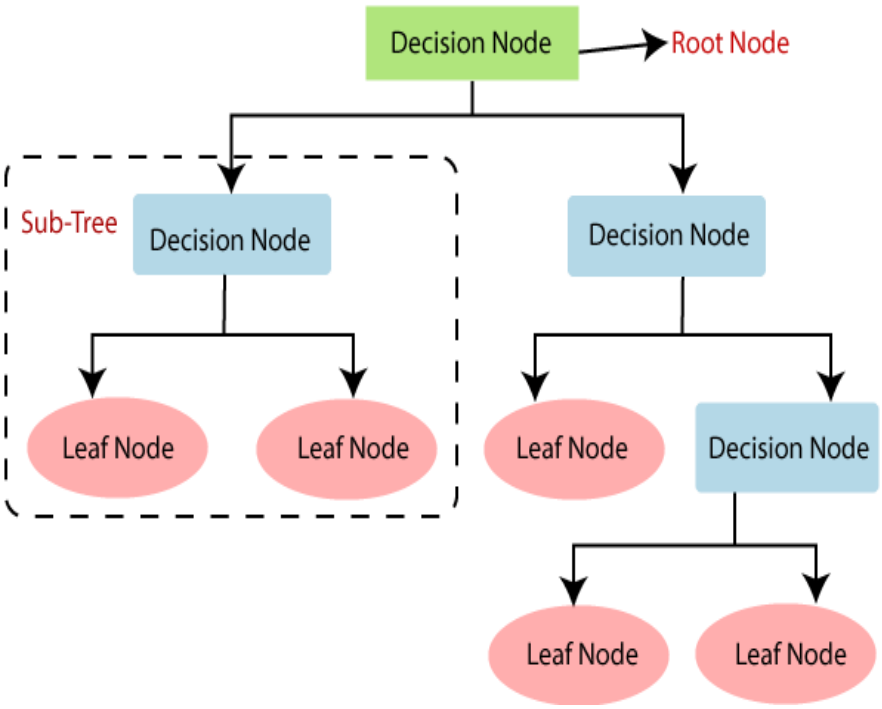

Figure 6 Architecture of Decision Tree

Random Forest

The random forest technique is a refined version of the bagging approach, encompassing both bagging and feature selection randomness to produce a collection of distinct decision trees. The method of randomly selecting features, often referred to as "feature bagging" or the "random subspace approach," results in a set of features that minimizes correlation among the trees.

One distinguishing factor between decision trees and random forests is that while the former evaluates all potential feature divisions, the latter only examines a chosen subset. Three primary hyperparameters need specification before initiating the random forest's training: node size, tree count, and the sampled feature count. Once configured, the random forest can address both regression and classification tasks.

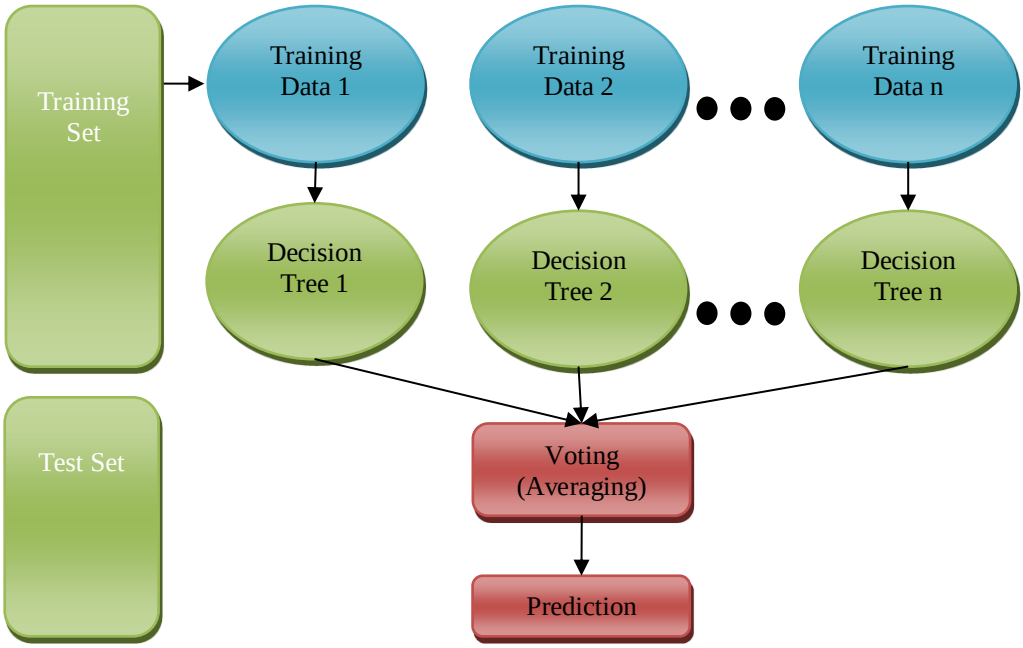

Figure 7 Architecture of Random Forest

The random forest method consists of multiple decision trees. Each tree is based on a data sample taken from the training dataset with the possibility of repetition. This sampling method is often referred to as a bootstrap sample. From this sample, a third is reserved as test data, termed the out-of-bag (oob) sample, which will be discussed further.

Gradient Boosting

Gradient Boosting is a powerful machine learning technique used for regression and classification tasks. It combines the principles of boosting and gradient descent to enhance the performance of weak learners, typically decision trees. Starting with an initial model, usually a basic prediction like the mean of the target, the algorithm iteratively refines this model. In each iteration, it computes the residuals (the differences between predictions and actual values) and trains a new weak learner to predict these residuals.

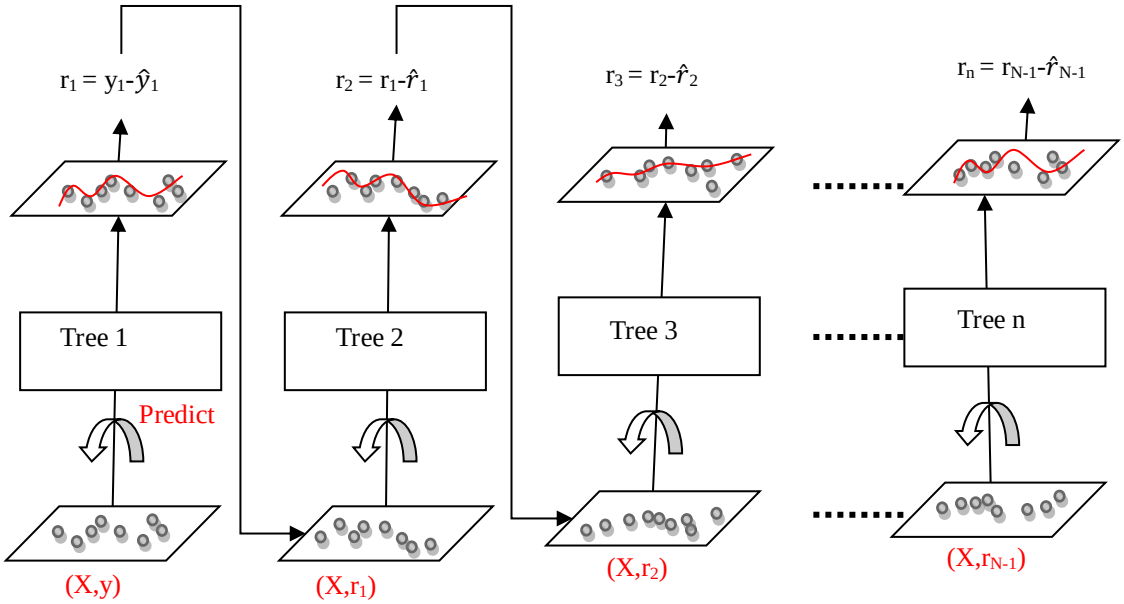

Figure 8 Architecture of Gradient Boosting

Table 1 Differentiation Among Tree-Based Classifiers

Algorithm	Key Features	Variants	Advantages	Disadvantages
Decision Trees	Top-down, recursive split. Uses Gini or entropy.	CART, ID3, C4.5	Intuitive Handles mix of features.	Overfitting with deep trees. Biased towards features with more levels.
Random Forest	Ensemble of trees. Majority voting.	-	Reduces overfitting. Handles missing values.	Less interpretable. Computationally intensive.
Gradient Boosted Trees (GBT)	Builds models iteratively. Corrects errors of last.	XGBoost , LightGBM, CatBoost	Higher accuracy. Different predictor variables.	Overfitting if not tuned. Computationally intensive.

IV. RESEARCH METHODOLOGY

During the functioning of the system, extensive information and runtime statistics are recorded in logs. Logs are crucial for analyzing usage patterns in large-scale computer systems. However, automatically parsing these logs can be challenging due to their unstructured nature. Many studies focus on log collection, templating, vectorization, and classification to analyze online usage patterns. Based on these studies, a model has been proposed that uses clustering and an ensemble tree-based machine learning classification for analyzing weblog behavior.

Proposed Methodology

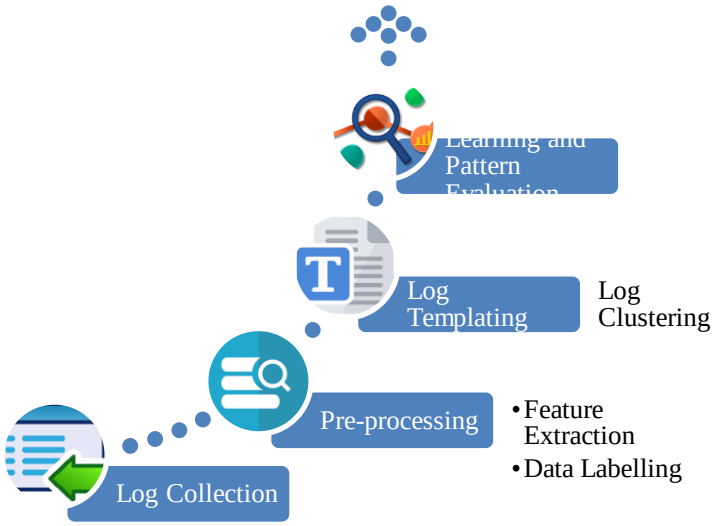

Figure 9 Working Steps for Web Server Access Pattern Analysis

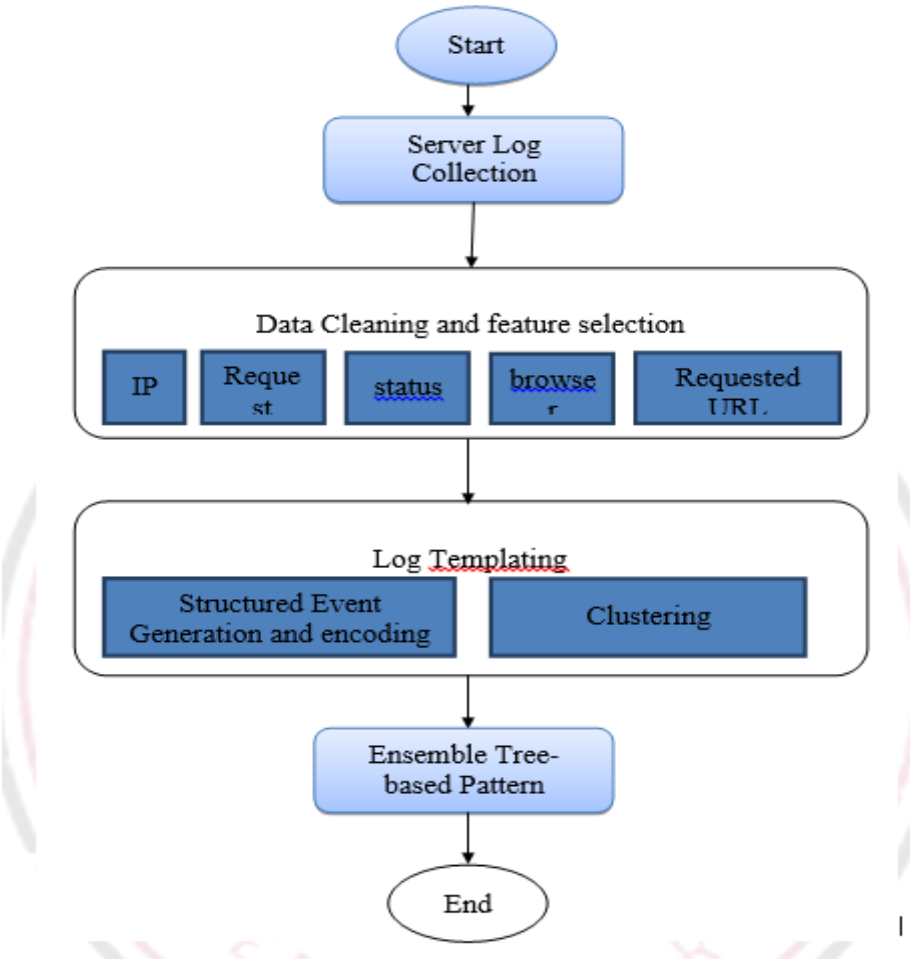

Figure 10 Working Flowchart of Web Server Access Pattern Analysis

Log Collection

When it comes to the monitoring of computer systems, one of the most crucial duties for software developers and operation engineers is log gathering. A computer system or network device can send its logs to a log file, a syslog, a trap, a snmp, or even a programme API. These are some of the most common methods for receiving logs. The research activity will typically involve the utilization of certain open log files as raw data sources. The HDFS log file is a dataset that was compiled from more than 200 EC2 machines owned by Amazon. As the primary resource for log collecting, this work makes use of the HDFS log file.

Data Pre-Processing

The initial phase of data handling is termed "pre-processing," which encompasses feature extraction and data labeling. Specifically, it involves gleaning the relevant features from raw server logs and assigning them one of two labels: a value of 1 signifies anomalous data, whereas a value of 0 denotes typical behavior.

Log Templating

Logs, which consist of free-text information, represent a form of unstructured data. Log parsing aims to transform the content of these raw messages into structured event templates. Log parsers can be broken down into three distinct groups. Methods that are based on clustering make up the first category of this list. The distance between each pair of logs is used as the primary factor in the clustering process. Each cluster is responsible for the generation of event templates. The second group consists of methods that are based on heuristics. The log templates can be directly extracted using these methods, which are based on heuristic criteria. Algorithm used for log templating is illustrated as below:

Algorithm: Log Templating

Input: Weblogs W_s

Output: Cluster sets C_s

1: Begin

2: For i in W_s

3: $EW_{s_i} \xleftarrow{encoding} W_{s_i}$

4: $CS_i \xleftarrow{GWO} EW_{s_i}$

5: End

6: Return C_s

7: End

By using one-hot encoding, we enhance the richness of our training data, offering scalability as required. This method of assigning numerical values facilitates a more direct computation of probabilities for our values. Given its ability to offer more nuanced predictions compared to singular labels, we've chosen one-hot encoding for values produced by proposed system.

To numerically define a data clustering problem, consider a set of M entities represented as $O = \{O_1, O_2, \ldots . O_M\}$. The vector $O_p = O_p^1, O_p^2, O_p^3 \ldots \ldots O_p^M$ signifies the pth data item, where o_p^j represents the jth characteristic of o_p^j. The clustering concept involves assigning each object O to one of the K clusters, represented as $D = D_1, D_2, D_3, D_4, \ldots \ldots \ldots D_K,$ ensuring every cluster has a minimum of one object.

Exploring the search domain involves a methodical assessment. In the initial stages, an algorithm sifts through the search landscape, aiming to identify superior solutions. During this exploration phase, search agents may bypass local peaks to ensure a comprehensive assessment of the entire search domain. As the algorithm progresses, the emphasis shifts from exploration to honing in on the best possible solution, known as the exploitation phase of an algorithm. Striking the right balance between these two phases is crucial for the algorithm's efficacy, thus introducing a fresh approach is often advantageous. In the GWO, the balance between these phases is orchestrated by the variable $\vec{b}$. As previously mentioned, this value decreases linearly over time.

Advantages

- Fewer parameters.
- Simple principles.
- Easy to implement.

Disadvantages

- Slow convergence speed.
- Low solution accuracy.

V. RESULT AND DISCUSSION

This section presents the outcomes from the simulation. This chapter presented an discription about simulation scenario along with result analysis with comparative state-of-art. To gauge the efficacy of proposed algorithmic approach, it was tested under the subsequent setup:

- Pentium Core I5-2430M CPU @ 2.40 GHz
- 4GB RAM
- 64-bit Operating System
- Python Platform

A. TOOLS USED

In this section, we will discuss the many tools that were utilized during the process. Python was chosen for the study's experimental programming language implementation. Python is a high-level programming language that is dynamic, object-oriented, and powerful. It also supports collaborative programming. Between the years 1985 and 1990, it was developed by Guido van Rossum. Similar to Perl, Python code is distributed under a public license known as the GPL Public License (GPL). You will leave this class with a solid comprehension of the Python programming language. Python is a computer language that is intended to be simple and straightforward to learn. It makes use of English terms rather than symbols and has fewer syntactical elements than a number of other computer languages.

- The Python programming language is an interpreted language. Python is under the control of its interpreters during the execution process. It is not necessary for us to construct your program before processing it. This is analogous to the language abilities required for PERL and PHP.
- Python is a language that supports dynamic content. You are able to write your program in Python by sitting at a Python line and interacting with the interpreter in an interactive manner.
- Python is a computer language that takes an object-oriented approach. Python is capable of supporting the Object-Oriented programming strategy or method, which encloses programming instructions within objects.

Python programming boasts several notable characteristics, including:

It accommodates multiple paradigms such as Object-Oriented Programming, Structured Programming, and Functional Programming.

Python can function both as a scripting language and can be compiled to byte code for a wide range of applications.

The language offers advanced dynamic data types and can execute dynamic type checking.

It possesses an automatic garbage collection feature.

Integration with other languages like C, C++, as well as technologies like COM, ActiveX, and CORBA, is straightforward.

Numpy: It's available for free as an open-source project. Python's numerical module is known as NumPy. The NumPy library was written in Python. It is an abbreviation for "Numerical Python," to provide its full name. This library provides a repository for array processing algorithms and multidimensional array objects. NumPy's intended benefit is a 50x speedup for array objects over regular Python lists. nd-array is the name of NumPy's array object, and it comes equipped with a number of helpful features that make working with arrays a breeze. In the world of computers, here is where references are stored.

(Scientific) Python bundles NumPy, SciPy, and Matplotlib are widely used together (plotting library). This duo is frequently substituted for MatLab, a widely-used and powerful computer environment. Comparatively, Python, MatLab's alternative, is currently considered to be a more cutting-edge and all-encompassing programming language. NumPy's open-source nature is an added bonus. NumPy's primary feature is the nd-array class of N-dimensional arrays. The term is used to describe a collection of identical products. Items in the collection can be located using a zero-based index. In a nd-array, the size of each element matches the size of the underlying memory. Data-type objects make up each element of the nd-array (called d-type).

Keras: Keras is a well-liked application programming interface (API) for complex NN tasks. It is written in Python and supports several back-end neural network computation engines. Rather than doing low-level operations like tensor products and convolutions on its own, Keras relies on a back-end engine. Although Keras is compatible with a wide variety of back-end engines, Google's TensorFlow is the primary back-end and the one used by default. As was previously noted, the Keras API is already a part of TensorFlow as tf.keras, and it will take centre stage as the primary TensorFlow API in version 2.0. With Keras, you have the flexibility to change between several backend implementations. Only one of these five frameworks, TensorFlow, has adopted Keras as its primary high-level application programming interface. Keras is an open-source, TensorFlow-based deep learning framework with pre-assembled modules for all neural network computations. Keras's high-level Python interface and flexibility to employ a variety of compute backends make it easy to learn and

implement. Keras is more sluggish than some other deep learning systems, but it's incredibly intuitive and easy to use for this purpose.

Pandas is a library in Python that provides efficient, flexible, and expressive data structures for working with "relational" or "labelled" data. Its intention is to be the basis for genuine and accurate data analysis in Python. Its ultimate goal is to be the go-to free software for statistical modelling and modification across all languages. It has made significant progress toward its goal. Pandas were developed in 2008 by AQR Capital Management employee Wes McKinney. He successfully lobbied the AQR for permission to release the Pandas into the public domain. Chang In 2012, she became the library's second major donor among AQR staff. There have been numerous publications of pandas. The most up-to-date version of Pandas is 1.4.1. Some of the traits of pandas are:

DataFrame objects can be easily created from previously existing Python and NumPy data structures, even if they contain irregularly indexed data, by employing effective, flexible group by capabilities to conduct split operations. Data Aggregation and Transformation with Pandas

- Data Handling: With Pandas, users can efficiently aggregate and transform data sets, merging them as necessary.
- Smart Slicing: Pandas offers intelligent label-based slicing, indexing, and subsetting of large data sets.
- Reshaping: Users can flexibly reshape and pivot data sets.
- Merging: The library has built-in capabilities for combining data sets.

Pandas primarily offers two data structures for manipulation:

- Series: Analogous to a column in an Excel sheet. While labels don't necessarily need to be hashable, they must be unique.
- DataFrame: A two-dimensional, tabular data structure. It supports both integer and label-based indexing. A DataFrame consists of rows, columns, and the data contained within.

Matplotlib, designed for Python and its numerical extension NumPy, enables data visualization and plotting across various platforms. It stands as an open-source counterpart to MATLAB. Matplotlib's API integrates seamlessly with GUIs to embed graphs. The matplotlib.pyplot module offers MATLAB-like functionalities. Each method in pyplot modifies or creates figures. Typically, a few lines of Python code using Matplotlib can produce visual representations of data.

B. Parameters Used

Accuracy: It is one of the most important parameters for determining the classifier's performance. The mathematical expression of accuracy is given in eqn (5.1):

$$\text{Accuracy} = \frac{(\text{TP} + \text{TN})}{(\text{TP} + \text{TN} + \text{FP} + \text{FN})} \tag{5.1}$$

Precision: It's calculated as the ratio of properly identified pattern to the total number of access logs, and it's given by:

$$\text{Precision} = \frac{\text{TP}}{(\text{TP} + \text{FP})} \tag{5.2}$$

Recall: Recall is the ratio of true positive (TP) to them sum of false negative and true positive.

$$\text{Recall} = \frac{\text{TP}}{(\text{TP} + \text{FN})} \tag{5.3}$$

F1-score: The F1-score is the result of harmonic mean of precision and recall rates.

$$\text{F1-score} = 2 * [(\text{precision} * \text{recall}) / (\text{precision} + \text{recall})] \tag{5.4}$$

Here, TP stands for True Positive

TN stands for True Negative

FP stands for False Positive and FN stands for False Negative.

C. Dataset Description

For this experimental analysis, we utilize a dataset sourced from the Apache access log server. The Apache access log is an integral part of web server management, offering insights into various user activities and the server's response to them.

The insights garnered from these logs can be invaluable for security, optimization, and understanding user behavior. The dataset derived from the Apache access log server contains the following attributes:

- IP Address: Represents the internet protocol address of the user or system making a request to the server.
- Datetime: Indicates the exact date and time when a request was made.
- GMT: Stands for Greenwich Mean Time, providing a standardized reference for time zones.
- Request: Details the specific action or resource being requested by the user, such as accessing a particular webpage or downloading a file.
- Status: Provides the HTTP status code in response to the request, indicating if the request was successful, had an error, etc.
- Size: Represents the size of the data that was transferred in response to the request.
- User Agent: This refers to the browser or tool the user utilized to make the request, offering insights into user preferences.
- Country: Points out the geographical origin of the request, allowing for regional analysis of user activity.
- Label: This might be an attribute used for classification, such as categorizing requests as 'normal' or 'malicious'.

Provides a visual example of the APACHE Web log file for a specific site server. This visualization is essential for a quick overview and understanding of the dataset's structure and the kind of information available.

ip	datetime	gmt	request	status	size	referer	browser	country
114.125.221.132	2019-07-01 10:54:15	+0700]	GET /bkd_baru/assets/images/scan_sertifikat/D0...	200.0	12133	http://universitas.com/bkd_baru/assets/images/...	Mozilla/5.0 (Linux; Android 5.1.1; SM-J111F Bu...	Indonesia
114.125.221.132	2019-07-01 10:54:23	+0700]	GET /bkd_baru/assets/images/scan_sertifikat/D0...	200.0	15491	http://universitas.com/bkd_baru/assets/images/...	Mozilla/5.0 (Linux; Android 5.1.1; SM-J111F Bu...	Indonesia
114.125.221.132	2019-07-01	+0700]	POST /bkd_baru/assets/images/scan_sertifikat/D...	200.0	16305	http://universitas.com/bkd_baru/assets/images/...	Mozilla/5.0 (Linux; Android 5.1.1; SM-	Indonesia

Figure 5.1: Server Log Data Visualization

Table 5.1: Performance Analysis of Classifiers

Classifiers	Accuracy	Precision	Recall	F1-score
LinearSVM	73.67%	73.67%	100.00%	84.84%
RandomForest	98.93%	98.57%	100.00%	99.28%
DecisionTree	98.93%	98.57%	100.00%	99.28%
KNeighbors	96.20%	95.49%	99.55%	97.47%
GradientBoosting	94.67%	93.25%	100.00%	96.51%
Ensemble Tree Based Learning	99.00%	99.00%	98.00%	99.00%

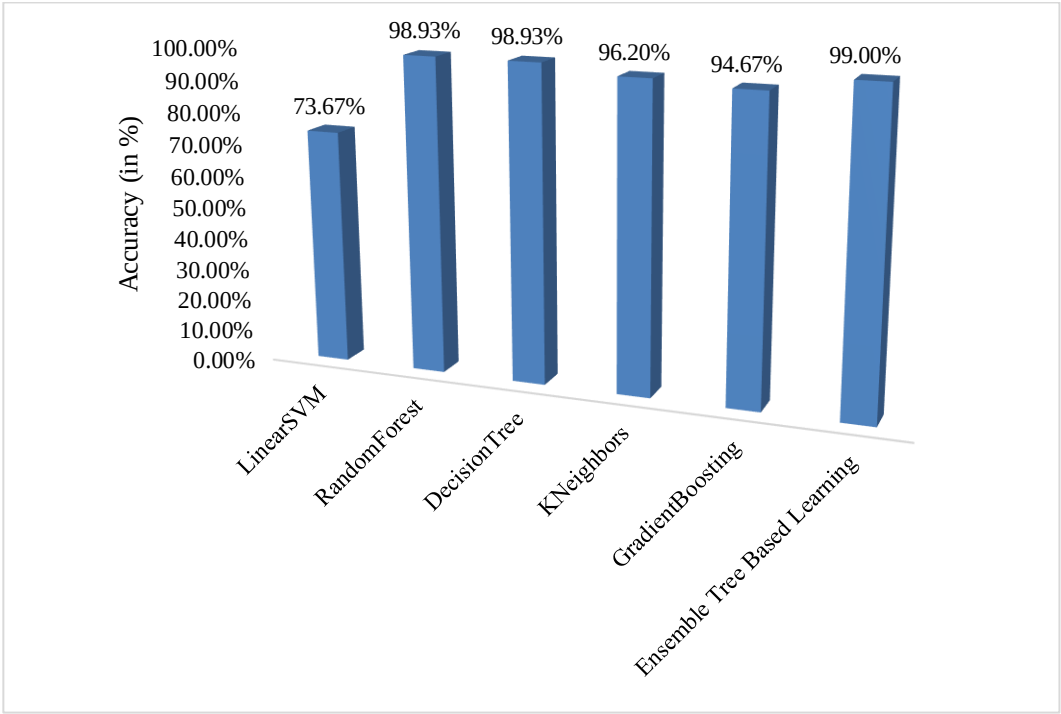

Figure 5.2: Accuracy Comparison of Classifiers

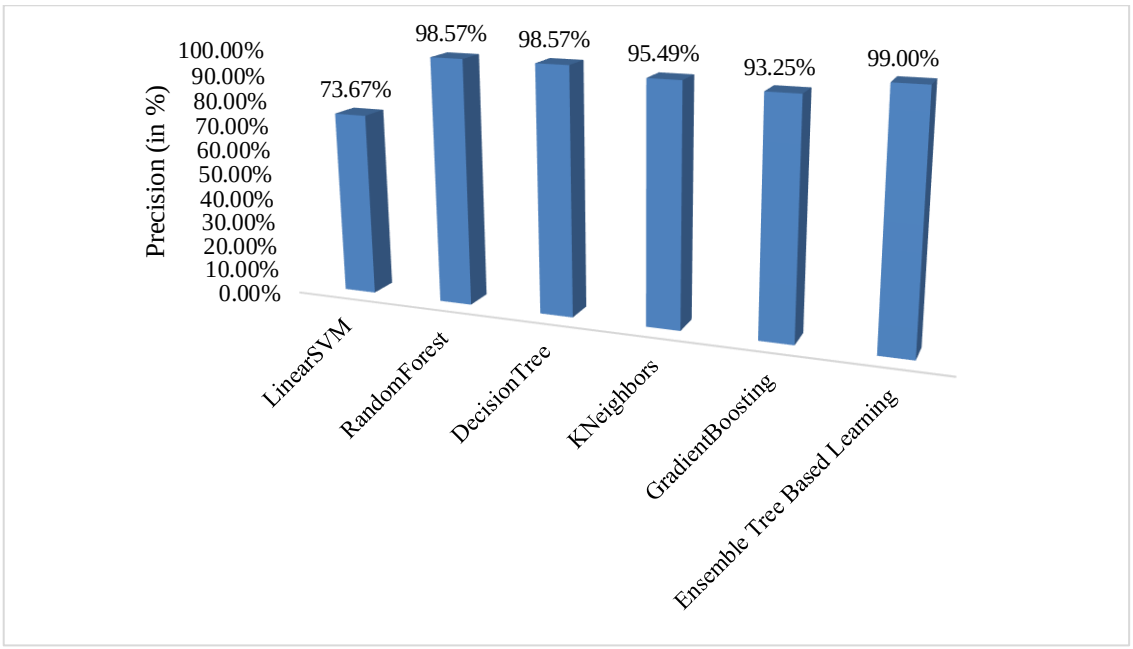

Figure 5.3: Precision Comparison of Classifiers

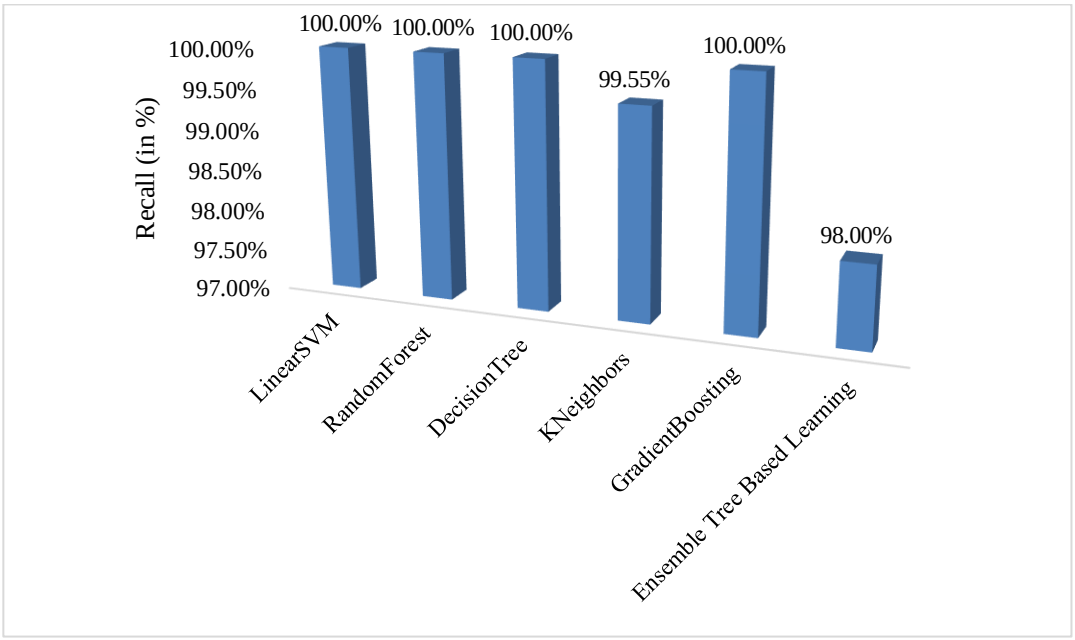

Figure 5.4: Recall Comparison of Classifiers

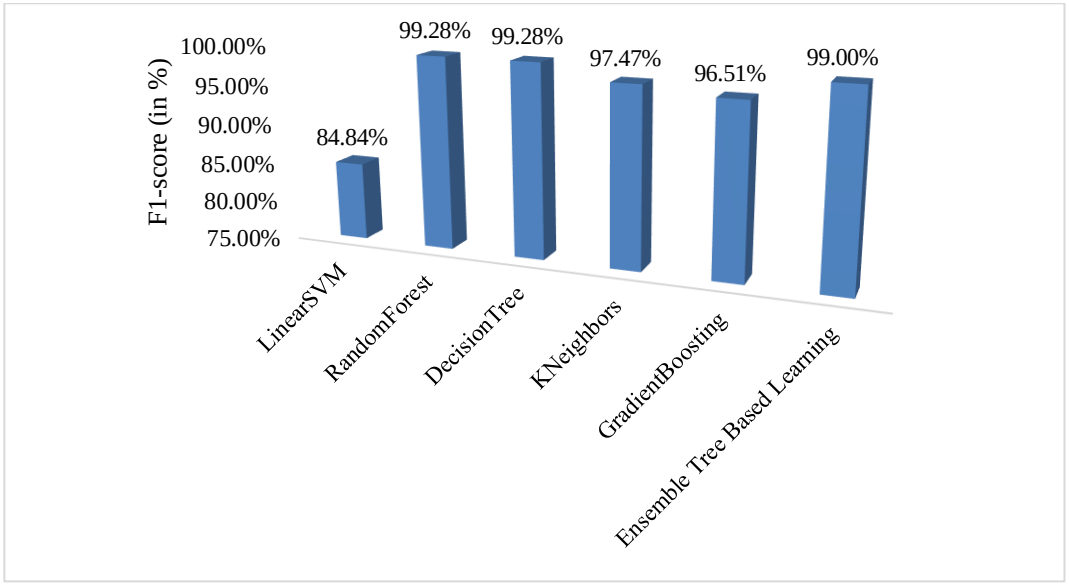

Figure 5.5: F1-score Comparison of Classifiers

ROC curve is presented to show the area of curve between true positive rate and false positive rate which is 98%.

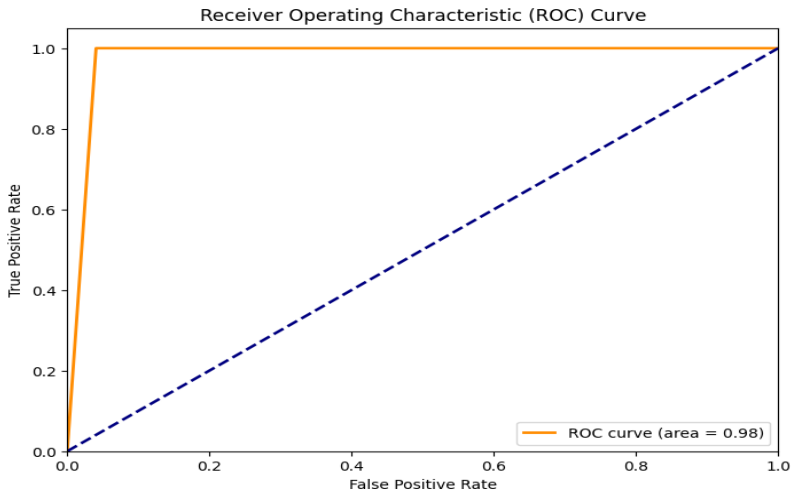

Figure 5.6: ROC Curve for Ensemble Tree Based Learning

Table 5.2: Result Evaluation with Different Training and Testing Ratios

Training/Testing Ratio	Accuracy	Precision	Recall	F1-score
60:40	99%	99%	98%	98%
70:30	99%	99%	98%	99%
80:20	99%	99%	97%	98%

Table 5.2 provides the results of a classification model evaluated with different training and testing ratios. The table shows the performance metrics (Accuracy, Precision, Recall, and F1-score) for three different ratios: 60:40, 70:30, and 80:20. Across all three training/testing ratios, the model demonstrates very high accuracy, with a consistent score of 99%. This suggests that the model performs extremely well in terms of overall correctness of predictions. The precision values are also consistently high at 99% for all ratios. This indicates that the model is effective at making positive predictions, and when it predicts a positive result, it is correct 99% of the time. Recall values are also quite high, ranging from 97% to 98%. This indicates that the model can correctly identify a high percentage of the actual positive instances in the dataset. The F1-scores, which consider both precision and recall, are also very high, ranging from 98% to 99%. These F1-scores suggest that the model achieves a good balance between precision and recall, making it suitable for tasks where both false positives and false negatives are important to minimize.

D. Comparative Analysis

Here, in this section, the comparative state of art models is presented. For comparison, different data mining techniques are selected and their results are presented below in table 5.3. The proposed Ensemble Tree based Learning model seems to be the best among the ones listed in terms of overall performance across all metrics. Logistic Regression appears to be the least effective model based on the metrics provided, with particular weaknesses in F1-score. Models like XGBoost and Decision Tree provide a stable and reliable performance, making them good benchmarks to compare other models against. It's important to consider the context and the problem being tackled when choosing a model. While Ensemble Tree based Learning performs best here, it might not always be the case for every dataset or problem.

Table 5.3: Comparative State-of-Art

Model	Accuracy	Precision	Recall	F1-Score
RepTree [1]	98%	-	-	-
Logistic Regression[39]	57.52%	63.00%	58.00%	44.00%
Decision Tree [39]	78.80%	79.00%	79.00%	79.00%
XGBoost [39]	87.09%	87.00%	87.00%	87.00%
Isolation Forest [40]	83.00%	-	-	-
Ensemble Tree based Learning (Proposed)	99%	99%	98%	99%

V. CONCLUSION

This research paper contributes to the field of web mining by proposing an effective ensemble tree-based learning approach for web usage pattern analysis. The study emphasizes the advantages of web log pattern analysis in diverse applications and addresses associated challenges, including privacy concerns. The proposed methodology, leveraging tree-based classifiers, demonstrates high performance on a real-world dataset, highlighting its potential for practical implementation. The comparative analysis underscores the superiority of the ensemble tree-based learning model over other data mining techniques. As web data continues to grow exponentially, the methodologies presented in this research offer valuable insights for extracting meaningful information, enhancing user experience, and supporting business decision-making in the dynamic online environment.

REFERENCES

[1] B. Gao, "The Use of Machine Learning Combined with Data Mining Technology in Financial Risk Prevention," Comput. Econ., vol. 59, no. 4, pp. 1385–1405, Apr. 2022, doi: 10.1007/S10614-021-10101-0/METRICS.

[2] H. Khatter and A. K. Ahlawat, "An intelligent personalized web blog searching technique using fuzzy-based feedback recurrent neural network," Soft Comput., vol. 24, no. 12, pp. 9321–9333, Jun. 2020, doi: 10.1007/S00500-020-04891-Y/METRICS.

[3] W. Song, N. Jiang, H. Wang, and J. Vincent, "Use of smartphone videos and pattern recognition for food authentication," Sensors Actuators B Chem., vol. 304, p. 127247, Feb. 2020, doi: 10.1016/J.SNB.2019.127247.

[4] P. Verma and N. Kesswani, "FEDUS : A comprehensive algorithm for web usage mining," https://doi.org/10.1080/02522667.2019.1616912, vol. 41, no. 3, pp. 835–854, Apr. 2019, doi: 10.1080/02522667.2019.1616912.

[5] Karthikeyan, T., Karthik Sekaran, D. Ranjith, and J. M. Balajee. "Personalized content extraction and text classification using effective web scraping techniques." International Journal of Web Portals (IJWP) 11, no. 2 (2019): 41-52.

[6] Mittal, Ruchi, Varun Malik, Vikas Rattan, and Deepika Jhamb. "Performance comparison of tree-based machine learning classifiers for web usage mining." In Proceedings of International Conference on Communication, Circuits, and Systems: IC3S 2020, pp. 379-387. Springer Singapore, 2021.

[7] Gholizadeh, Nahid, Hamid Saadatfar, and Nooshin Hanafi. "K-DBSCAN: An improved DBSCAN algorithm for big data." The Journal of Supercomputing 77 (2021): 6214-6235.

[8] S. Asadianfam, H. Kolivand, and S. Asadianfam, "A new approach for web usage mining using case based reasoning," SN Appl. Sci., vol. 2, no. 7, pp. 1–11, Jul. 2020, doi: 10.1007/S42452-020-3046-Z/TABLES/3.

[9] S. N. Mohanty, J. Rejina Parvin, K. Vinoth Kumar, K. C. Ramya, S. Sheeba Rani, and S. K. Lakshmanaprabu, "Optimal rough fuzzy clustering for user profile ontology based web page recommendation analysis," J. Intell. Fuzzy Syst., vol. 37, no. 1, pp. 205–216, Jan. 2019, doi: 10.3233/JIFS-179078.

[10] H. Li and M. Wei, "Fuzzy clustering based on feature weights for multivariate time series," Knowledge-Based Syst., vol. 197, p. 105907, Jun. 2020, doi: 10.1016/J.KNOSYS.2020.105907.

[11] P. Bhattacharjee and P. Mitra, "Density-Based Mining Algorithms for Dynamic Data: An Incremental Approach," Stud. Comput. Intell., vol. 1028, pp. 313–335, 2022, doi: 10.1007/978-981-19-1021-0_13/COVER.

[12] M. Kotyrba, E. Volna, R. Jarusek, and P. Smolka, "The use of conventional clustering methods combined with SOM to increase the efficiency," Neural Comput. Appl., vol. 33, no. 23, pp. 16519–16531, Dec. 2021, doi: 10.1007/S00521-021-06251-9.

[13] M. S. Bhuvaneswari and K. Muneeswaran, "User Community Detection From Web Server Log Using Between User Similarity Metric," Int. J. Comput. Intell. Syst., vol. 14, no. 1, pp. 266–281, 2020, doi: 10.2991/IJCIS.D.201126.002.

[14] K. Mostafa, A. Attalla, and T. Hegazy, "Data mining of school inspection reports to identify the assets with top renewal priority," J. Build. Eng., vol. 41, Sep. 2021, doi: 10.1016/J.JOBE.2021.102404

online-pdf-no-copy.com