International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 7, Issue 2, ISSN No. 2455-2143, Pages 356-363
Published Online June 2022 in IJEAST (http://www.ijeast.com)

LOG ANOMALY DETECTION USING MACHINE LEARNING

Keerthan S, Tejass Sanker J, U R Narasimha Rangan, Vinayak G mulugund, Mrs. Vani K S
Department of Information Science and Engineering
Nitte Meenakshi Institute of Technology Bengaluru, Karnataka

***Abstract*—Anomaly detection in log record manages finding text which can give hints to the reasons and the anatomy offailure of a run and most normally, area and instrument explicit customary articulation from recently seen error messages is utilized to dig the applicable message from the log file**
The proposed model is assessed utilizing the BGL and network log message Data sets. The outcomes acquired show that the quantity of negative examples anticipated to be positive is low. Log messages are presently extensively utilized in cloud and programming frameworks. They are significant for classification and anomaly detection as a great many logs are created every day. In this paper, a model for log message anomaly detection is proposed

***Index Terms*—log anomaly, BGL logs, network logs, Machine learning, random forest, Decison Tree, Multinominal Naive Bayes**

I. INTRODUCTION

Software frameworks and a few applications accessible over the world are supposed to run appropriately 24 hours on the clock and failures in these applications can create a few issues for the clients.
These failures are kept in log documents which are produced at whatever point there is an application running on the framework. Logs are utilized for some errands like observing the framework, execution of an application, identify the disappointments and a few different purposes.
As step by step there is expansion in progression in innovation and more innovation is becoming valuable to people there is additionally ascend in the security worries for the innovation and many dangers are making the innovation come up short. These minor or significant dangers can be a gigantic variable for a devastating mechanical disappointment
To defeat these issues and give a superior User Interface and client framework and reduction how much frailty these dangers are giving, a superior framework execution is required, and a superior framework that can do log oddity location at higher effectiveness and quicker to execute can be utilized to track down these dangers or interruptions in the framework to forestall further disappointments in the framework.
By distinguishing these irregularities or anomaliesthat might happen whenever, we can investigate them, yet recognizing these abnormalities physically is almost incomprehensible as it takes gigantic measure of assets, thus these oddities can be identified utilizing a few Machine Learning calculations with great frequencies.
Consequently our principal objective is to identify these anomalies in log records involving Machine learning algorithms which can find the peculiarities in the framework logs with the best effectiveness and exactness, which will help us in making a device which will be skilled to make main driver examination on bombing applications and caution a similar in the User Interface which will assist developers with finding the high recurrence oddities in log documents produced by the framework. Then, at that point, the framework specialists can deal with fixes to the anomalous application or framework undertaking to correct it and give better client experience in the innovation world
Our application is fit for taking framework logs as info, pre-process the log, distinguish the ways of behaving of the record and framework interference. The model we made is right now utilizing 3 different Machine learning algorithms which is trained on the data set we are utilizing. the model chooses the best working ML model out of the three and give the outcome out in our front end.
Depicting our front end, it gives the client to enter single log or transfer a log document to send the input to back end where it is pre-processed and broke down as an anomaly ornot and afterward print the analysis toward the front page for the client.
Involving this application designers can find the anomalies in the log documents and diminish the security concerns, system failures and increment the framework execution whichwill help in improved efficiency.
Identifying anomalies in system behaviour, or anomaly detection, is a crucial step in creating a reliable computer system. Systems are more vulnerable to faults and vulnerabilities as they become more complicated than ever before, which results in lost revenue. As a result, anomaly detection has grown increasingly difficult and crucial. System logs are a great source of data for online monitoring and anomaly detection because they capture notable events as they happen from currently operating processes. Both in academia and business, log-based anomaly detection has

emerged as a research area of practical significance. Anomaly detection techniques that currently exist use logs. These methods now in use only use one machine learning model. The merits of each model, however, vary depending on the target system. It can be challenging for developers to choose which approach will best solve the current practical issue.

Log collecting, log parsing, feature extraction, and anomaly identification are the four key phases in anomaly detection. There are two types of ways for detecting anomalies: supervised methods and unsupervised methods. Despite their great accuracy, supervised algorithms are not always appropri- ate in a real-world situation since data labels are frequently not accessible. To solve this issue, unsupervised anomaly detectiontechniques are offered.

II. BACKGROUND

Log Anomaly detection is not a new topic, huge amount of time is already being invested along with human resource. Researchers have proposed many techniques for anomaly detection and published numerous papers in different journals. Even today the research is being carried out to develop better algorithms to find anomalies in a system log file. Below we discuss some recent published research papers that deal with the usage of machine learning to develop better algorithms for anomaly detection.

Anomaly Detection of System Logs Based on Natural Language Processing and Deep Learning

In this paper, they have proposed a model that combines natural language processing with machine learning algorithmsand deep learning algorithms, in contrast to other traditional approaches that use system logs for anomaly detection, such as the rule-based approach, the method based on association analysis, or the classification-based approach. The process involves converting the raw logs to vectors using the Word2vec and TFIDF vectorizers to extract thedataset's features. These have been used with the Thunder Bird super computer's dataset. In order to discover anomalies, the transformed logs are then used to Long Short-Term Memory (LSTM), which is compared to other algorithms like Gradient Boosting Decision Tree and Nave Bayes. When logs were transformed to vectors, it was shown that the skip-gram model captured more semantic information than TFIDF [1][2][3]

Ensemble Methods for Anomaly Detection Based on System Log

In this research, two machine learning ensemble techniques-based anomaly detection algorithms are proposed. The proposed methodology was implemented by the authors using the HDFS dataset. Log collecting, log parsing, feature extraction, and anomaly identification are the four key phases in anomaly detection. The first technique combines the weighted prediction of ensemble methods using a mixture of experts to determine the outcome. The second approach simply divides the sample into n pieces and applies n models on the same to extract features. The outcomes of the suggested two-method assembly learning approach appear to validatethe veracity and accuracy of the suggested procedures. When compared to single same proposed model, the accuracy of two ensemble learning methods was higher. [4][5][6]

An Integrated Method for Anomaly Detection From Massive System Logs

In order to perform anomaly detection from huge logs, the approach described in the study integrates the K-prototype clustering and k-NN classification algorithms. This method also makes use of a unique clustering-filtering-refinement framework. To effectively characterise behaviours, log characteristics are first examined, then features based on session data are extracted. Second, the K-prototype clustering algorithm is used to divide the data set into various clusters depending on the retrieved features. The remaining events are then considered anomaly possibilities for further study after the evident normal events that are highly cohesive clusters are sorted away. In order to determine the local and global degrees of abnormality for these anomaly candidates, two distance-based features are constructed. K-NN has been used to generate accurate findings based on the new features. A platform for anomaly detection is used, and a log collection is built, to verify this procedure. [7]

Incremental Analysis of Large-Scale System Logs for Anomaly Detection

Traditional machine learning algorithms that are implemented centrally cannot handle the enormous volume of logs thatare produced every day. As a result, the paper presents a distributed log analysis method for finding anomalies. It is an expansion of the framework that uses incremental analysis to find abnormalities early. The log data that hasaccumulated as of that point is periodically processed in the extended version. Analyze the approach's trade-offs between accuracy and average anomaly detection time using an influential dataset with different time periods. On the basis of an existing benchmark dataset, controlled experiments were carried out to calculate the approach's accuracy and efficacy. The proposed approach can reduce time consumption while maintaining the same level of accuracy as the offline approach, according to the results. [8]

Unsupervised log message anomaly detection

Two deep auto-encoder networks and an isolation forest are used in this paper's suggested model. The training and feature extraction for one auto encoder takes place while the

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 7, Issue 2, ISSN No. 2455-2143, Pages 356-363
Published Online June 2022 in IJEAST (http://www.ijeast.com)

anomaly detection for the other takes place. For the prediction of positive samples, the isolation forest is used. On datasets of log messages from Open stack, Thunderbird, and BGL, the suggested model is put into practise. Results indicate that, particularly with isolation forest and auto encoder, the proportion of negative samples projected to be positive that are wrongly classified is minimal. It has been found that an isolation forest alone does not effectively solve the issue. In spite of the fact that only a little quantity of data was used for training, it is clear from the final observation that precision for positive logs is significantly better than that of algorithms. [9]

III. SYSTEM DESIGN

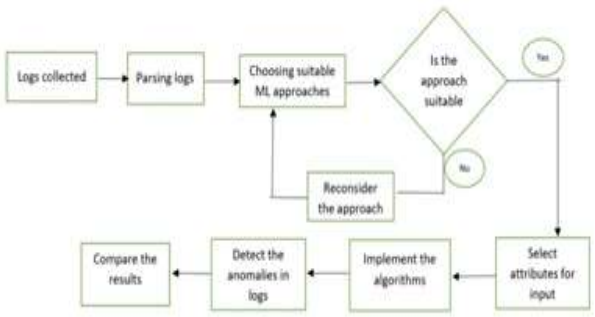

Fig. 1. System Design

As the above flow chart depicts, we have gathered a few logs from Logpai. These include BGL, HDFS. We have also collected network logs which would work similar to system logs so that we can predict any type of attacks in the network logs. Then we parsed the raw logs into structured format. This was the main step as we need to structure the logs beforewe feed it into the model. During parsing we also added a few preprocessing steps such as Tokenizing and stemming to increase the accuracy of the model.

Then we selected the most powerful algorithms, tweaking the parameters to achieve the best accuracy. If the approach was suitable we considered those models. In this project the models we used are Gaussian Naive Bayes, Random Forest, and Decision Tree. Once the trained model is ready, we have tested the model which has given good accuracy. Then the model is used to predict the Anomalies. We implemented this by creating a webpage where we take the input from the user and our model predicts the output and displays it to the user. The three models will be used to predict the output and the highest frequent output will be displayed to the user

IV. FRAMEWORK

A. White space Tokenizer

Text preparation is the first step in every NLP project. Simply speaking, preprocessing input text involves transforming the data into a format that can be analysed and predicted. It's an essential stage in creating a fantastic NLP application. There are different ways to pre - process text stop word removal, tokenization and stemming. The tokenization process is crucial. It is the process of segmenting a stream of textual data into tokens, which can be words, terms, sentences, symbols, or other significant components. There are numerous open-source programmes that can be used to tokenize data.

A tokenizer known as a "Whitespace Tokenizer" splits on and eliminates solely whitespace characters. Word, CoreLabel, and other Lexed Token objects may be returned by this implementation. It has a parameter that determines whether to interpret EOL characters as whitespace or as a token. The class returns an EOL as a Word with the String value "n" ifit is a token.

B. Snowball Stemmer

By reducing word inflection to its root forms, the natural language processing approach known as stemming helps to prepare text, words, and documents for text normalisation. Inflection is the process through which a word is changed to transmit a variety of grammatical categories, including tense, case, voice, aspect, person, number, gender, and mood, according to Wikipedia. Therefore, even if a word may have various inflected forms, the NLP process becomes more complicated when multiple inflected forms appear in the sametext. In order to reduce words to their most basic form, or stem which may or may not be a valid word in the language we use stemming.

Snowball is a small string processing language for creating stemming algorithms for use in Information Retrieval, plus a collection of stemming algorithms implemented using it. A Snowball programme can be converted into source code in another language using the Snowball compiler; at the moment, Ada, ISO C, C, Go, Java, Javascript, Object Pascal, Python, and Rust are supported.

C. Count Vectorizer

By using the vectorization technique, you can speed up the execution of your code. When you are implementing an algorithm from scratch, it is a really fascinating and significant technique to optimise algorithms. We can now make our code run quickly with the aid of highly optimised numerical linear algebra libraries in C/C++, Octave/Matlab, Python, etc. In machine learning, there is the idea of an optimization method that calculates to obtain the optimal parameters for the machine learning model while attempting to lower the error. Therefore, compared to an unvectorized implementation, a vectorized implementation of an optimization algorithm can significantly speed up processing.

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 7, Issue 2, ISSN No. 2455-2143, Pages 356-363
Published Online June 2022 in IJEAST (http://www.ijeast.com)

A collection of text documents is transformed into a vector of term/token counts using Scikit-Count Vectorizer. learn's Ad- ditionally, it makes it possible to pre-process text data before creating the vector representation. It is a very flexible feature representation module for text because of this functionality.

D. Encoders

In real life, category string values make up the majority of the data, whereas most machine learning models only deal with integer values and some with other, potentially intelligible values. In essence, all models carry out mathematical operations that can be done with a variety of instruments and methods. But the unpalatable fact is that mathematics depends entirely on numbers. Consequently, we can sum up by saying that the majority of models demand numbers asthe data, not strings or anything else, and these numbers can be float or integers. In order to provide the models with the data with transformed categorical values and to help them make better predictions, categorical data must first be converted into integer format, a process known as encoding.

Label encoder: Python's label encoding technique substitutes a numeric value between 0 and the number of classes minus one for the category value. If there are 5 distinct classes represented by the categorical variable value, we use (0, 1, 2, 3, and 4).

One Hot Encoder: Encode categorical features as a one-hot numeric array. This transformer should have as its input an array-like structure of integers or strings that represent the values assigned to categorical (discrete) attributes. A one-hot (also known as "one-of-K" or "dummy") encoding approach is used to encode the characteristics. For each category, a binary column is created, and a sparse matrix or dense array is produced (depending on the sparse parameter). The categories are by default derived by the encoder from the distinct values in each feature. As an alternative, you may also manually de-fine the categories. Numerous scikit-learn estimators, includinglinear models and SVMs using the default kernels, require categorical data to be fed, and this encoding is necessary for this.

V. ALGORITHMS

A. Random forest

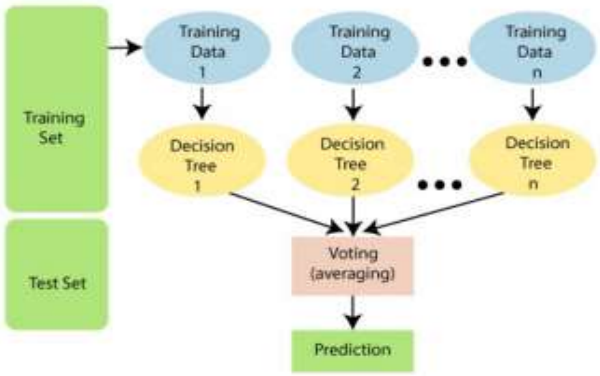

Fig. 2. Random Forest

Random Forest is a very famous ML algorithm that belongs to the supervised learning technique. It is innovated on the idea of ensemble learning. The ensemble learning method integrates various classifiers to address difficult issues and enhance model performance. Random Forest is a classifier that uses more than one decision trees on various subsets of the input dataset and averages the output or results to increase the dataset's predicted accuracy. Random forests are bagged decision tree models that split on a subset of features on each split. It uses Bootstrapping where we randomly sample with replacement. Bagging, or bootstrap aggregating, is where we create bagged trees by creating X number of decision trees that is trained on X bootstrapped training sets. Then we predict the output by the mean value. Considering a single Decision Tree, there may be high varience, hence aggregation techniques makes the variance tweak, hence as the majority vote. So weather classification or regression analysis, Random forest can be used as it can handle binary features, categorical and numerical. Its best with high dimensional data. It is also much faster as we train only on a subset of the training data

B. Decision Tree

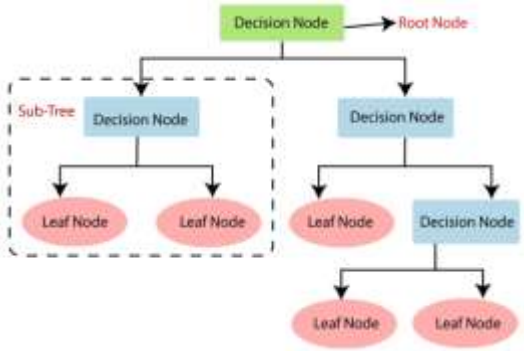

Fig. 3. Decision tree

It is a tree like structured classifier, where the internal nodes stand in for a data set's features, branches for the decision making process, and each leaf node for the classification

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 7, Issue 2, ISSN No. 2455-2143, Pages 356-363
Published Online June 2022 in IJEAST (http://www.ijeast.com)

result. It is a graphical depiction for obtaining all feasible answers to a choice or problem based on predetermined conditions. The Classification and Regression Tree algorithm (CART), is used to construct a tree. A decision tree only poses a question and divides the tree into sub-trees according to the response (Yes/No).The decision of making strategic splits extremely affects a tree's accuracy. The decision criteria are different for classification and regression trees. The purity of each node increases with target variable. There are many tweaks in the attribute selection such as entropy, Gini Index, Chi square, Information Gain and so on. Also we can do Pruning where we trim of the branches of the tree so the structure is minimal without affecting the accuracy.

C. Multinominal Naive Bayes

Popular in Natural Language Processing(NLP) is the Bayesian learning method known as the Multinomial Naive Bayes algorithm . It determines the likelihood of each tag for a particular sample and outputs the tag with the highest likelihood. The Naive Bayes method is an effective technique for examining text input and resolving issues involving severalclasses. Because the Bayes theorem, which calculates the likelihood of occurrence based on prior knowledge of theevent's conditions, is the foundation upon which the Naive Bayes theorem is based. We determine the chance of class A when predictor B is available. Based on the aforementioned formula. The model is fast and easy to implement but requires the predictors to be independent. In real life senarios, it isn't difficult to find such cases as maximum real life cases predic- tors are independent.[10]

$$P\left(\frac{B}{A}\right) = \left(\frac{P(A \cap B)}{P(A)}\right)$$

Fig. 4. Multinominal NB

VI. IMPLEMENTATION

A. Datasets

The unstructured raw log file that was collected was converted into a structured format using various Natural Language Processing methods as described below.

- 1117838570 2005.06.03 R02-M1-N0-C:J12-U11 2005-06-03-15.42.50.675872 R02-M1-N0-C:J12-U11 RAS KERNEL
INFO instruction cache parity error corrected

The log file was split into each line and further each line was tokenized using whitespace tokenizer. Few tokens were merged to form a proper meaningful sentence. Stemming was performed on the tokenized words to prevent any sort of errors occurring due to it. Each list was then given a header and then converted to structured format.

Once the structured dataset is ready we remove all the unnec- essary columns. The content column are then vectorised using count vectorizer and is appended to the dataset. The values in the label column are then used converted to normal if there is a '-' else it is considered to be an anomaly. The data is then preprocessed using Label Encoder and one hot encoder. Once the preprocessing is done the dataset is split into testing and taining dataset where 80 percent of the logs are considered fortraining.

For the network log the dataset was already in csv format and hence only the headers was added to the dataset.

0,tcp,ftp-data,SF,491,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00, 0.00,0.000.00,1.00,0.00,0.00,150,25,0.17,0.03,0.17,0.00,0.00, 0.00,0.05,0.00,normal,20

The different type of attacks were clubbed into few specific types of attack like R2L Attack, DOS attack, Probe Attack and U2R Attack. The data was prepocessed using label encoder. The dataset was then trained into test and train dataset where 90 percent of the data was used for training

B. Converting unstrcutured log to structured log file

First the raw data logs are taken and split based on new lines. Then we used popular Natural Language Processing tech- niques such as Tokenizing and stemming. Before tokenizing we converted every log to lower case. We have used white space tokenization where the tokens are split according to the space. Next we extracted the content message from these tokens and formed into a single meaningful sentence. Then the words and sentences are stemmed using Snowball stemmer which removes and changes the stop words or tenses of a word. After the preprocessing we convert it into a structured csv file with appropriate headers given. Once the structured csvwas ready it was ready to preprocess the structured file so that we can train the model.

C. Preprocessing the data

In this preprocessing step first we drop a few columns which are unique for every line. As these are unique such as id, date, node, time these attributes can be removed as they dont have any impact in training.

Then we have used Count vectorizer to vectorize the text messages. Once the text message column was vectorized, we then concatenated these vectors with the original dataset removing the text messages. We then split it into x and y label where x is the features and y is the target. The Target variable y was labeled using label encoder. Then a few features with categorical labels are converted to numericals using One hot Encoder. Then we used our models to train our data. We used Decision Trees, Random Forest and Multinomial Naive Bayes. The model was trained on the training data. Training data was acquired from the dataset which was around 80percent of it.

We then predicted our result on the test data and

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 7, Issue 2, ISSN No. 2455-2143, Pages 356-363
Published Online June 2022 in IJEAST (http://www.ijeast.com)

calculatedthe accuracy of the model which showed accuracy around 96 percent.

D. Implementing different algorithms

In the previous section all the labels were classified which included types of anomalous logs. So we considered to remove it and label all types of anomalous logs to single category Anomaly. So now we had two categorical values named normalor anomaly. We then used the same model and the accuracy wasincreased by 2

Then the same strategy was applied on other models and we were successful in achieving higher accuracy in all the models. Our aim was to create all the models with higher accuracy so that for a given new log if we apply all the models, we would consider the output which will be very frequent.

VII. TESTING

The dataset was test on three different algorithms for both the datasets namely Decision Tree, Random Forest and Na¨ıve Bayes. The data was trained and tested for each algorithm. Our project was based on Network logs and system logs by Blue Genie (BGL).We tested our model with the test data available and recorded the highest achievable accuracy. Below are the accuracy score, f1 score, recall and confusion matrix of all the models on network and system logs.[11]

A. BGL logs

Random forest

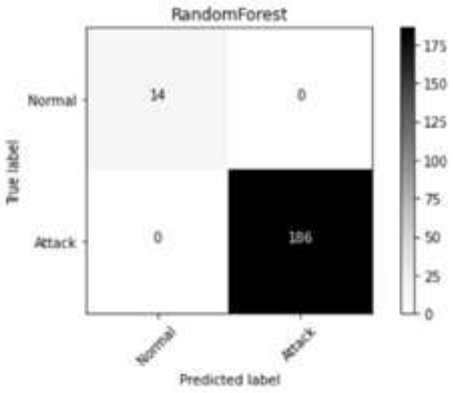

Fig. 5. RF confusion matrix (BGL)

Decision Tree

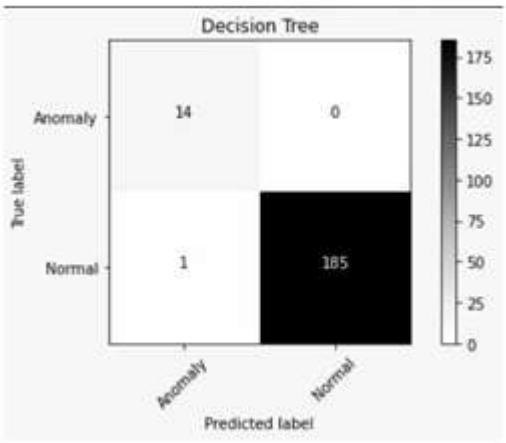

Fig. 6. DT confusion matrix (BGL)

Multinominal naive bayes

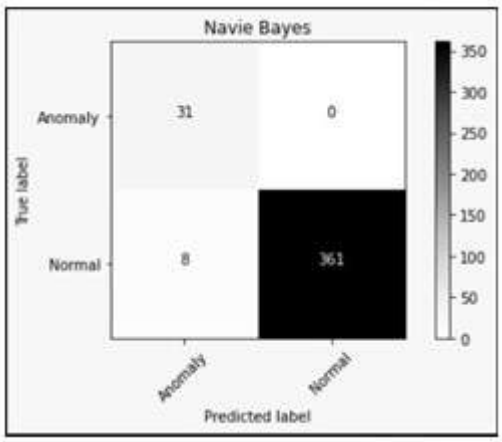

Fig. 7. MNB confusion matrix (BGL)

B. Network logs

Random forest

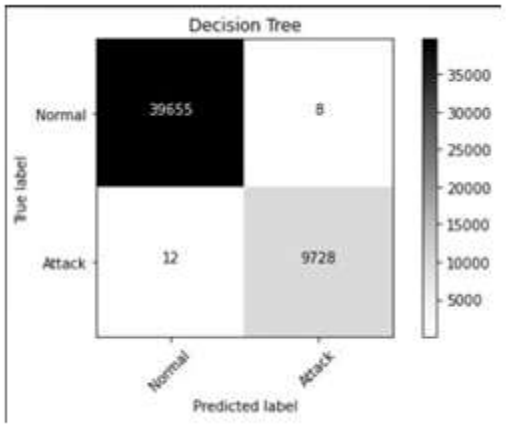

Fig. 8. RF confusion matrix (Network)

Decision Tree

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 7, Issue 2, ISSN No. 2455-2143, Pages 356-363
Published Online June 2022 in IJEAST (http://www.ijeast.com)

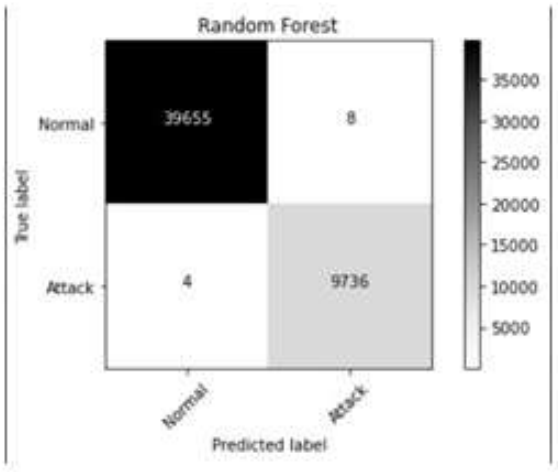

Fig. 9. DT confusion matrix (Network)

Multinominal naive bayes

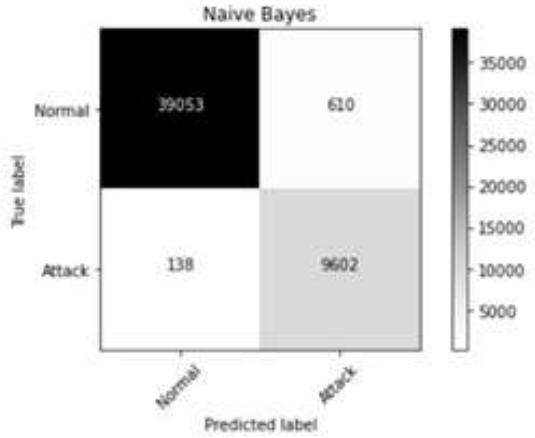

Fig. 10. MNB confusion matrix (Network)

VIII. RESULTS

As a result, machine learning algorithms are well-trained on the datasets to prevent over fitting and to deliver the best accuracy and efficiency. The Frontend webpage takes a log and reports whether the log is anomalous or not. We can also upload log files to the webpage, which alerts us of the amount of anomalies in the uploaded log and provides us option to print the anomalous log.

IX. CONCLUSION

Cloud systems or any system in general are capable of generating millions of text log messages every day. Thus, the detection of anomalies in these logs is very important, the huge amount of logs makes this a difficult task and hard to overcome. In this paper, a model was proposed for anomaly detection using supervised learning technique. The model proposed was evaluated using two log message data sets, namely Network and Blue Genie supercomputer logs. The results attained shows that this proposed model is better than various other models that have been used in the literature survey. In the future, the performance of other models such as Gaussian Mixture Model or LSTM and better methods of preprocessing can be investigated

X. REFERENCES

[1] M. Wang, L. Xu, and L. Guo, "Anomaly detection of system logs based on natural language processing and deep learning," in 2018 4th International Conference on Frontiers of Signal Processing (ICFSP), 2018, pp. 140–144.

[2] R. B. Yadav, P. S. Kumar, and S. V. Dhavale, "A survey on log anomaly detection using deep learning," in 2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), 2020, pp. 1215–1220.

[3] D. A. Bhanage, A. V. Pawar, and K. Kotecha, "It infrastructure anomaly detection and failure handling: A systematic literature review focusing on datasets, log preprocessing, machine deep learning approaches and automated tool," IEEE Access, vol. 9, pp. 156 392–156 421, 2021.

[4] X. Xia, W. Zhang, and J. Jiang, "Ensemble methods for anomaly detec- tion based on system log," in 2019 IEEE 24th Pacific Rim International Symposium on Dependable Computing (PRDC), 2019, pp. 93–931.

[5] P. Sun et al., "Context-aware learning for anomaly detection with imbal- anced log data," in 2020 IEEE 22nd International Conference on High Performance Computing and Communications; IEEE 18th International Conference on Smart City; IEEE 6th International Conference on Data Science and Systems (HPCC/SmartCity/DSS), 2020, pp. 449–456.

[6] L.-P. Yuan et al., "Time-window based group-behavior supported method for accurate detection of anomalous users," in 2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Net- works (DSN), 2021, pp. 250–262.

[7] Z. Liu et al., "An integrated method for anomaly detection from massive system logs," IEEE Access, vol. 6, pp. 30 602–30 611, 2018.

[8] M. Astekin, S. Özcan, and H. Sözer, "Incremental analysis of large- scale system logs for anomaly detection," in 2019 IEEE International Conference on Big Data (Big Data), 2019, pp. 2119–2127.

[9] A. Farzad and T. A. Gulliver, "Unsupervised log message anomaly detection," ICT Express, vol. 6, no. 3, pp. 229–237, 2020.

[10] "A novel machine learning-based approach for the detection of ssh botnet infection," Future Generation Computer Systems, vol. 115, pp. 387–396,2021.

[11] J. Paul et al., "Prediction of road accident and severity of bangladesh applying machine learning techniques," in 2020 IEEE 8th R10 Human- itarian Technology Conference (R10-HTC), 2020, pp. 1–6.